“Electric Vehicle Charging Management System(EVCMS)”

Project Report

of

Software Engineering

(CS330)

By

Pathan Fiza Faiyazkhan (22000986)

Third Year, Semester 5

Bachelor of Technology (CSE)

Course In-charge: Prof. Darshan Parmar

Department of Computer Science and Engineering

Navrachana University, Vadodara

Autumn Semester

(Jul-Dec 2024)

Table of Contents

List of Figures

List of Tables

I Project Description

1 Project Overview

The Electric Vehicle Charging Management System (EVCMS) Application is designed to enhance the adoption of electric vehicles (EVs) and support the transition to a sustainable, eco-friendly transportation ecosystem. This application offers features that streamline the Searching nearest Charging Stations, Dynamic Scheduling and Reservation, Predictive Maintenance and Charging Completion time. By IoT Integration, the application monitors and analyzes charging patterns, predicts peak usage times and alleviate grid load. Supports On-Route and Roadmap Charging Points. Users can favorite frequently used charging stations for quick access and easy rebooking.

Abstract

The EVCMS Application aims to contribute to a greener future by making EV charging more convenient, efficient, and accessible, ultimately supporting the global shift towards sustainable transportation.

The primary goal of the EVCMS Application is to facilitate the widespread adoption of electric vehicles by providing a user-friendly, sustainable, eco-friendly transportation system and contribute to a greener future that connects EV owners with a network of charging stations allowing to Search Nearest Available Charging Stations , dynamic scheduling and real-time reservation of charging slots , Notification for maintenance and estimated completion times. By integrating IoT sensors the system can monitor and analyze charging patterns, predict peak usage times, and optimize charging schedules to minimize energy costs and reduce grid load.

Furthermore, the application supports multiple payment options and gives Improved Travel Experience which supports EN route or Roadmap charging points (Source and destination point) and charging points along the way, making long-distance travel more manageable, Provides adding of favorite charging station like frequently used station for rebooking.

2 The Purpose of the Project

2a The User Business or Background of the Project Effort

1. **Search Nearest Available Charging Stations**:
    - Locates nearby EV charging stations based on the user's current location.
2. **Dynamic Scheduling and Real-Time Reservation**:
    - User Registration and Login will be done here. Schedules charging sessions dynamically and reserve charging slots in real time.
3. **Notification and Feedback System**:
    - Receive notifications for maintenance updates and estimated completion times of charging sessions. Feedback forms will be supplied after user picks up their Ev's.
4. **Integration with IoT Sensors**:
    - Monitor and analyze charging patterns using IoT sensors.
    - Predict peak usage times and optimize charging schedules to minimize energy costs and reduce grid load.
5. **Multiple Payment Options**:
    - Support various payment methods for convenience.
6. **Improved Travel Experience**:
    - Provide information on en route charging points (from source to destination).
    - Identify on-route charging stations along the travel route to facilitate long-distance travel.

7. **Favorite Charging Stations**:

    - Allow users to add frequently used charging stations to a list for easy rebooking and quick access.

This solves problems of EV owners as:

- It is easily Accessible to search for the nearest available charging stations
- Efficient Charging Scheduling especially during peak periods reducing waiting time.
- Energy Cost and Grid Load Management as charging pattern is analyzed.
- Payment Flexibility as multiple payment is available. Long-Distance Travel Challenges are solved as provides en-route and on-route facility.
- Efficient for rebooking as Charging Stations are marked Favorite.

2b Goals of the Project

The main goal of the EVCMS (Electric Vehicle Charging Management System) Application is to facilitate the widespread adoption of electric vehicles (EVs) by making the charging process more convenient, efficient, and accessible. This overarching goal is pursued through several key objectives:

- Enhance Accessibility
- Optimize Charging Efficiency
- Support Sustainable Energy Use
- Improve Travel Experience
- Offer Flexible Payment Options
- Increase Reliability

By achieving these objectives, the EVCMS Application aims to support the global shift towards sustainable transportation and contribute to a greener future.

Examples

We want to notify Ev Owners for maintenance updates and estimated completion times of their Charging sessions.

We want to manage Electric Vehicle Charging Stations and also our main aim is to analyze charging patterns using IoT sensors to minimize energy costs and reduce grid load.

2c Measurement

1. **Ease of Use and Increased Customer Engagement:**
    - User-friendly interface for searching, reserving, and managing charging sessions.
    - Real-time data and insights on charging station usage and patterns by reviews.
2. **Real-Time Information and Reservations:**
    - Real-time reservation of charging slots and dynamic scheduling. Reduces waiting times and allows for better planning and flexibility.
3. **Enhanced Customer Satisfaction and Loyalty:**
    - Improved reliability and user experience as users are informed with notifications about maintenance and estimated completion times to avoid disruptions and ensure a reliable charging experience.
4. **Optimized Travel Planning:**
    - En route and destination charging points, along with favorite station lists.
5. **Cost Savings , Energy Efficiency and Data-Driven Decision Making**:
    - IoT-based monitoring and optimization of charging schedules. Helps reduce energy costs and grid load, which can be reflected in lower charging costs for users.
    - Analytics from IoT sensors on charging patterns and peak times. Provides valuable insights for making strategic decisions about expanding or upgrading charging facilities.

3 The Scope of the Work

The Electric Vehicle Charging Management System (EVCMS) will be used by firstly who own Electric vehicles and in various scenarios, including:

- **Public Charging Stations**:
  - **Urban Areas**: In cities and towns where public charging stations are widely available, EVCMS will help EV owners locate, reserve, and manage their charging sessions.
  - **Highways and Rest Stops**: Along major highways, the system will assist drivers in finding and using charging stations during long-distance travel, ensuring that they can recharge en route.
  - **Public Transportation**: Public transport authorities managing electric buses or other EVs can utilize the system for efficient charging station management and scheduling.
- **Hotels and Resorts**:
  - EVCMS can be integrated into hotel services, allowing guests to charge their vehicles during their stay, with the convenience of booking slots in advance.
- **Renewable Energy Integration**:
  - **Solar-Powered Charging Stations**: EVCMS can be used at charging stations powered by renewable energy sources like solar panels, optimizing the use of green energy for EV charging.

3a The Current Situation

Process	Before EVCMS	After EVCMS
Searching for Charging Stations	Manual search using maps or physical signs with no real-time data. Unaware of availability of Charging Stations.	Automated search with real-time information on Nearest availability and compatibility
Booking Charging Slots	Manual booking via phone or in-person, This method was inefficient, prone to human error, and didn't allow for dynamic adjustments	Dynamic, centralized booking system with real-time adjustments.
Predictive Maintenance	Fixed-interval or reactive maintenance.No notifications are provided.	predictive maintenance with notifications and reducing costs.
Energy Usage Management	Unoptimized energy use, leading to higher costs and grid strain.	Optimized energy management minimizing costs and reducing grid load using IOT Sensors.
Route Planning for Long Trips	Manual planning using separate tools with no integration into navigation.	Integrated route planning of automatic charging station with en route on route.
User Experience Consistency	Inconsistent interfaces and processes across different stations.	Standardized, intuitive user experience across all supported charging stations.

Table 1 – Table of Current Situation-Before and After EVCMS:

3b Context of the Work

1. **EVCMS (System)**
    - **Central System**: The EVCMS is the core of the diagram, responsible for managing all functionalities related to electric vehicle charging, including searching, booking, payment processing, and maintenance management.
2. **External Entities**

    - **EV Owners/Users**: EV owners use the system to search for charging stations, book slots, make payments, and receive maintenance, Booked slots alerts.

    - **Charging Stations**: Charging stations receives booking and payment information of EV Owners from EVCMS. Charging Stations updates Availability of Charging Slots, Booked Slots information and Pricing on EVCMS.

    - **Payment Gateways**: The payment gateways handle transactions initiated by users for charging services.

    - **IoT Devices/Sensors**: IoT devices at charging stations and in vehicles send data to the EVCMS, which uses it for predictive maintenance and energy management. For example, during peak demand times, the system can balance the load across multiple stations or adjust charging speeds to prevent overloading the grid. By analyzing energy usage patterns, the system can suggest off-peak charging times to users.

    - **Route Planning/Navigation Systems**: Provides EN route and ON route Planning. The EVCMS integrates with navigation systems to provide route planning and optimal charging stops along the route with suggested charging stations.

    - **Support/Maintenance Staff**: Staff receive alerts and data for maintaining Electric Vehicles and Scheduling next maintenance and addresses any issues.

Diagram Representation:

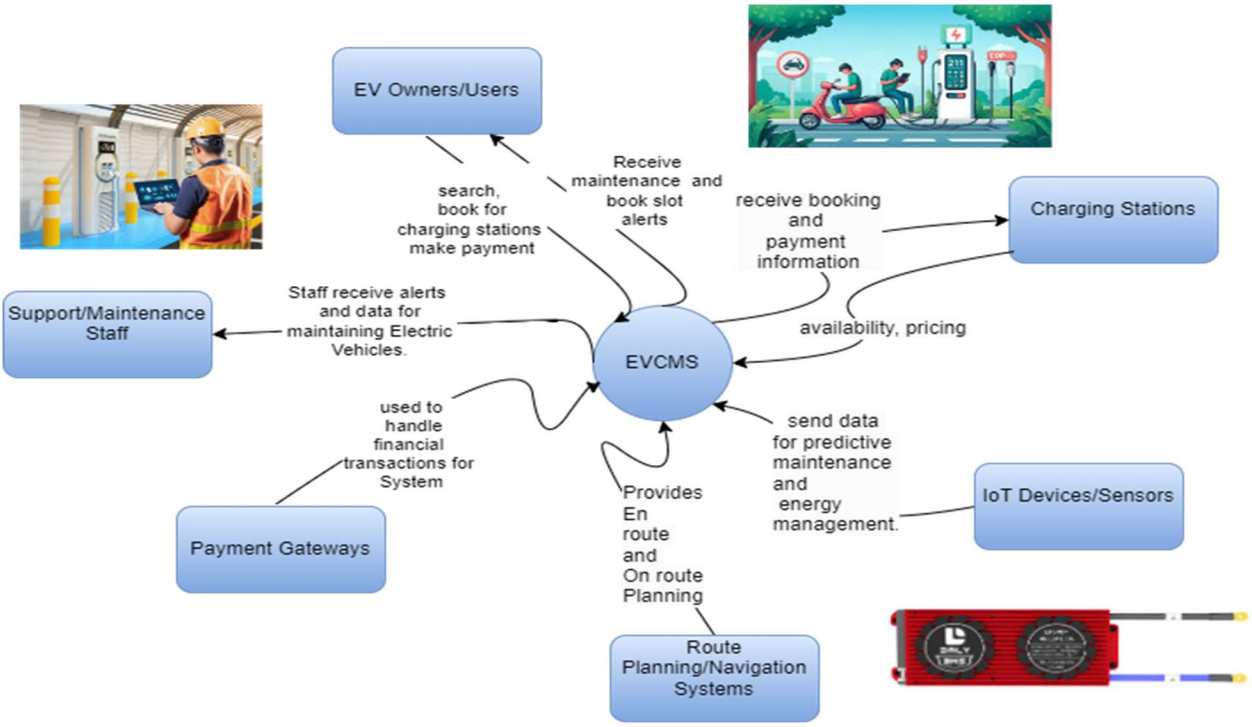


Figure 1 – Context Diagram:

3c Tools and Technologies Used

- Python can be used as the back-end for a mobile app built with Flutter on the front-end. This approach is known as a "cross-platform" or "polyglot" architecture, where the front-end is built with one technology (Flutter) and the back-end is built with another (Python).

  To integrate Python with a Flutter mobile app, we can typically use:

- REST API: We can build a Python-based REST API using a framework like Flask and have our Flutter app communicate with this API over HTTP.

1.Frontend- Flutter:

- Flutter is a popular open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase.
- Integrating Flutter with Python using the Flet library provides developers with a flexible and powerful solution for cross-platform development.
- **Usage in EVCMS**: Flutter is used to create the user interface of the EVCMS mobile application, ensuring a smooth, responsive, and visually appealing user experience across different devices and platforms.

2.Backend-Flask:

- **Role**: Flask is a lightweight and flexible web framework written in Python. It is designed to make it easy to build web applications quickly and with minimal setup.
- **Usage in EVCMS**: Flask is used to handle the backend logic of the EVCMS, including managing API requests, processing data from IoT devices, handling user authentication, and managing communication with the database.

3. Database-SQLite:

- **Role:** SQLite is a self-contained, high-reliability, embedded, relational database management system. It does not require a separate server process and allows for simple and efficient data storage and retrieval.
- **Usage in EVCMS**: Data Storage- Store user data, charging station locations, booking records, and transaction histories schedules, and historical charging patterns. Its simplicity and lightweight nature make it suitable for mobile applications and smaller-scale deployments.
- Data Processing- Analyze data collected from IoT sensors to predict peak usage times and optimize charging schedules.

4.Payment Gateway Integration:

- **Role**: A payment gateway is a service that authorizes and processes online payments, enabling users to securely pay for services via credit cards, debit cards, or other online payment methods.
- **Usage in EVCMS**: Multiple Payment Options- Support various payment methods including credit/debit cards, mobile wallets, and subscription-based payments using Razorpay.

5. GPS and Mapping Services:

- **Role**: GPS provides location data, while mapping services like Google Maps offer detailed maps and route planning capabilities.
- **Usage in EVCMS:** Location Tracking- Provide real-time location tracking to help users find the nearest available charging stations.
- Route Planning- Integrate with mapping services like Google Maps to offer en-route and on-route charging points for long trips.

6.IoT Sensors:[2]

- **BMS Sensors**: The Battery Management System (BMS) monitors and manages the performance and safety of a vehicle's battery.
- **Usage in EVCMS**: BMS sensors are integrated into the EVCMS to monitor the health and performance of EV batteries. Provide data to optimize charging schedules based on battery conditions and data on charging patterns, predict peak usage times, ensuring the longevity and safety of the battery.
    - **Current Sensors and Temperature Sensor**: Monitor the flow of electricity during charging and discharging and measure battery's temperature to prevent overheating
    - **Voltage Sensors:** Track the voltage levels to ensure safe and efficient charging.
    - **Load Sensors:** Help in load balancing by distributing energy evenly across multiple charging stations to prevent overloading and optimize energy use.

3d Work Partitioning

- Business Event List for EVCMS:

Event Name	Input from Adjacent Systems	Output to Adjacent Systems	Summary
1. User Requests Nearby Charging Stations	User Location	List of Charging Stations	The system receives the user's current location and provides a list of nearby charging stations, including availability, connection type, and pricing.
2. User Books a Charging Slot (Registration Process also)	Charging Station Selection	Booking Confirmation	After selecting a charging station, the user requests to book a slot. The system checks availability and confirms the booking, updating the station's schedule.
3. User Starts Charging Session	Charging Slot Booking	Charging Session Status	The user initiates a charging session at a reserved slot. The system monitors the session, providing real-time updates on charging progress and cost.
4. Payment Processing	Charging Session Completion	Payment Receipt	Once charging is complete, the system calculates the total cost and processes the user's payment. A receipt is generated and sent to the user.

5. Predictive Maintenance Alert	Sensor Data	Maintenance Alert	The system analyzes data from IoT sensors in the vehicle and charging stations to predict potential maintenance issues, sending alerts to users or service technicians.
6. Route Planning with Charging Stops	Route Information	Route with Charging Stops	The user inputs a travel route, and the system plans the best route, including necessary charging stops along the way. The optimized route is provided to the user.
7. User Adds Favorite Charging Station	Charging Station Details	Updated Favorites List	The user selects a frequently used charging station to add to their favorites. The system updates the user's profile with the new favorite for quick future access.
8. Real-Time Charging Alerts	Charging Session Status	Charging Alert	During a charging session, the system monitors conditions such as power levels and station load. If any issues arise, the system sends real-time alerts to the user.

9. User Registers for EVCMS	Registration Details	User Account Confirmation	A new user registers on the EVCMS platform by providing personal and vehicle information. The system verifies the details and creates a new user account.
10. Service Technician Schedules Maintenance	Maintenance Request	Maintenance Schedule	A service technician requests a maintenance slot for a charging station or vehicle. The system checks availability and schedules the maintenance task, notifying the technician.

Table 2 – Table of Work Partitioning-Business Event List for EVCMS:

- These business events define the core interactions within the EVCMS, providing a framework for discovering detailed requirements. This partitioning allows for focused development, ensuring that the EVCMS addresses all critical user needs and system responses.

3e Competing Products

EVgo: It operates a large public fast-charging network in another Country, offering charging solutions for a wide range of electric vehicles.

- Features: DC fast charging, real-time station information, mobile app support, and various payment methods.
- Limitations: It may have limited integration with third-party navigation systems and Sensors and with restricted access for other EV brands

Our EVCMS:

- EVCMS searches for nearby charging stations to dynamic scheduling and predictive maintenance.
- The integration of IoT sensors allows for intelligent monitoring and optimization of charging patterns.
- Multiple payment options, on-route charging points, and favorite station features enhance user convenience.
- Not limited to specific EV brands or proprietary networks, EVCMS is designed to be compatible with a wide range of EV models and charging stations.
- The system optimizes energy usage to reduce costs and alleviate grid strain, which is particularly important as EV adoption grows and demands on the electrical grid increase.

3f Actors

1. **Admin**:
   - Responsible for overseeing the entire EVCMS, including managing(Insert, update, delete, notifying) EV owners charging stations, its managers and Service Technicians, handling En route and On-route Functionality, and also managing notifications from sensors, keeps record of maintenance, peak usage, completion time and Next-time Charging Scheduling for EV.
2. **Electric Vehicle Owners (EV Owners)**:
   - Primary users who locate, reserve, and use charging stations, view their charging history, make payments, and receive notifications regarding charging status and updates.

3. **Charging Station Operators**:

    - Manage and maintain charging station Service Staff and Notification for Scheduling Charging slot, updating availability, setting pricing, and handling any technical issues related to the stations.

4. **Service Technicians**:

    - Handle the Predictive maintenance of EV, gives Notifications of completion time and Next when to Schedule Charging.

5. **Payment Gateway Providers**:

    - Facilitate the processing of payments made by EV owners for charging services, ensuring secure and reliable transactions.

6. **IOT Sensors:**

    - IoT sensors the system can monitor and analyze charging patterns, predict peak usage times, and optimize charging schedules to minimize energy costs and reduce grid load.

3g Modules

1. **User Registration & Login**:

    - Users can sign up and log in to the system. The admin reviews and approves new user registrations. Once approved, users receive a confirmation email and can access the system.

2. **Charging Station Locator**:

    - Allows EV owners to search for nearby charging stations, view station details such as availability, pricing, and connector types, and reserve a slot for charging.

3. **Reservation & Booking Management**:

    - Users can book a charging slot in advance. Admins and operators can manage these bookings, view the current schedule, and update availability. Provides adding of favorite charging station like frequently used station for rebooking

4. **Payment Processing**:

    - Facilitates secure payments for charging sessions, generates invoices, and maintains a record of all financial transactions. Integrated with various payment gateways for seamless and multiple payment processing.

5. **Maintenance and Notifications**:

    - Notifying users for Maintenance of EV, completion of Charging Ev and Next Schedule for charging. Sends notifications to users regarding reservation confirmations, payment receipts.

6. **Mapping Services**:

    - Integrates with mapping services to provide EV owners with turn-by-turn navigation to the nearest charging stations

    - Provide Navigation of en route charging points (from source to destination) on maps.
    - Identify on-route charging stations along the travel route to facilitate long-distance travel.

7. **Battery Management System (BMS) Integration and Energy Management**:

    - IoT sensors the system can monitor and analyze charging patterns, predict peak usage times, and optimize charging schedules to minimize energy costs and reduce grid load.
    - Monitors and manages energy consumption at charging stations of EV, optimizes charging schedules to reduce grid load.

4 Product Scenarios

Product scenarios are detailed, narrative descriptions of how end users are expected to interact with a product once it is fully developed. They serve as informal stories that illustrate specific use cases and real-world situations in which the product will be used. Should be understandable by both technical and non-technical stakeholders.

They serve as a tool for discussing and refining product features and functionalities with clients, project managers, and other stakeholders.

4a Product Scenario List

Scenario 1: Finding a Nearby Charging Station

Scenario 2: Booking a Charging Slot

Scenario 3: Making a Payment

Scenario 4: Monitoring Charging Status

Scenario 5: Predictive Maintenance Notification

Scenario 6: Route Planning with Charging Stops

Scenario 7: Adding a Favorite Charging Station

Scenario 8: Receiving Real-Time Charging Alerts

4b Individual Product Scenarios

Scenario 1: Finding a Nearby Charging Station

John is on a road trip and notices that his electric vehicle's battery is running low. He opens the EVCMS app on his phone and selects the option to find a nearby charging station. The app uses GPS to locate his current position and displays a list of the closest charging stations along with details like availability, connection type,

and pricing. John selects the station that best suits his needs and starts navigating to it using the app's built-in mapping service.

Scenario 2: Booking a Charging Slot

Sarah is heading out for a day trip and wants to ensure that she has a charging slot available when she reaches her destination. She opens the EVCMS app, searches for charging stations near her destination, and checks their availability. Sarah selects a station that offers fast charging and reserves a slot for 2 PM. The app confirms her booking and sends her a reminder an hour before the scheduled time.

Scenario 3: Making a Payment

After reaching the charging station, Mike plugs in his vehicle and starts the charging session through the EVCMS app. Once the charging is complete, the app automatically calculates the cost based on the energy used and the time spent charging. Mike selects his preferred payment method, reviews the transaction details, and completes the payment through the app. A receipt is sent to his email, and the transaction history is saved in the app.

Scenario 4: Monitoring Charging Status

Lisa is charging her vehicle at a station while she grabs a coffee nearby. Using the EVCMS app, she monitors the charging status in real-time. The app shows her the current battery level, estimated time to full charge, and the cost so far. When the battery reaches 80%, the app notifies Lisa that the vehicle is almost fully charged, so she can return to the station.

Scenario 5: Predictive Maintenance Notification

Tom has been using his electric vehicle for a year, and the EVCMS app has been tracking his battery performance. One day, Tom receives a notification from the app, informing him that based on the data collected, his battery might need maintenance soon. The app suggests nearby service centers and offers to schedule

an appointment. Tom appreciates the proactive approach and books a service slot for the weekend.

Scenario 6: Route Planning with Charging Stops

Emma is planning a long road trip and wants to ensure she has enough charging stops along the way. She enters her starting point and destination into the EVCMS app and selects the option to include charging stops. The app calculates the best route, highlighting the most convenient charging stations along the way, including those with available slots. Emma reviews the plan and starts her journey with confidence.

Scenario 7: Adding a Favorite Charging Station

Raj frequently uses a particular charging station near his workplace because it's reliable and has fast chargers. To make things easier, he adds this station to his favorites in the EVCMS app. Now, whenever he needs to charge his vehicle, Raj can quickly access this station from his favorites list and check its availability, making the booking process faster.

Scenario 8: Receiving Real-Time Charging Alerts

While his car is charging, Alex receives a real-time alert from the EVCMS app, informing him that the charging speed has slowed down due to an unexpected surge in demand at the station. The app suggests moving to another station nearby that currently has less load and faster charging speeds. Alex appreciates the prompt alert and decides to switch stations to save time.

5 Stakeholders

1. **Electric Vehicle Owners:**

- Individuals who own or are considering purchasing an EV and need a reliable way to manage their charging needs.

2. **Charging Station Operators:**

- **Managers:** Oversee the operations of charging stations and are responsible for ensuring efficient service delivery.
- **Service Providers:** Operate charging stations and are looking to connect with potential customers.

3. **Environmental Advocates:**

- Users who are committed to sustainability and seek solutions that align with their eco-friendly values.

5a The Client

- **Electric Utility Companies -** Companies that generate and distribute electricity may invest in EVCMS to manage their network of EV charging stations and optimize energy usage.
- **Private Charging Station Operators -** Companies that own and operate EV charging stations could be clients, looking to enhance their service offerings and efficiently manage their operations.
- **Automotive Manufacturers –** EV manufacturers, especially those focused on electric vehicles mainly Public Electric Buses, might invest in EVCMS to provide a comprehensive charging solution for their customers.

5b The EV Owners/Users

- Individuals who own electric vehicles or are considering purchasing one. They need a system to manage their charging needs effectively.
- The EV owners are the primary users of the EVCMS. Understanding their needs and preferences is crucial for gathering accurate requirements and ensuring the system meets their expectations.

5c Hands-On Users of the Product

- **User Category:** Electric Vehicle Owners, Charging Station Managers, Service Technicians.
    - EV Owners: Use the system for locating charging stations, booking slots, and managing payments.
    - Charging Station Managers: Manage station operations, monitor usage, and ensure the availability of services.
    - Service Technicians: Perform maintenance tasks based on data from sensors and system alerts.
- **Subject Matter Experience:**
    - EV Owners: Journeyman level in vehicle usage and basic technology.
    - Charging Station Managers: Master level in station operations.
    - Service Technicians: Master level in maintenance and repair.
- **Technological Experience:**
    - EV Owners: Novice to journeyman, depending on user familiarity with technology.
    - Charging Station Managers: Journeyman to master.
    - Service Technicians: Master.
- Defining the characteristics of these users is vital for designing a user-friendly and effective product. Their feedback and interactions will guide the usability and functionality of the EVCMS.

5d Priorities Assigned to Users

- **Key Users:**
    - **Electric Vehicle Owners:** They are the primary users, and their satisfaction is crucial to the system's success.
    - **Charging Station Managers:** Their feedback will directly impact the operations and effectiveness of the EVCMS.
- **Secondary Users:**
    - **Service Technicians:** While important, their requirements will be secondary to those of EV owners and managers.
- **Unimportant Users:**
    - **Casual users** who may only occasionally interact with the system.
- Prioritizing users ensures that the design focuses on the needs of those who will most impact the system's success.

5e User Participation

- **Electric Vehicle Owners:** Provide input on usability, interface design, and feature preferences. They should participate in usability testing and feedback sessions.
- **Charging Station Managers:** Contribute insights into operational requirements and system functionality. Participation in design workshops and main testing is essential.
- **Service Technicians:** Involved in defining maintenance requirements and testing the system's diagnostic tools.
- Ensuring adequate user participation is crucial for gathering accurate requirements and achieving a successful implementation.

5f Maintenance Users and Service Technicians

- Maintenance Users- EV Owners: The app provides predictive maintenance features based on usage patterns and registration details. EV owners would use this information to schedule maintenance, ensuring their vehicles remain in optimal condition.
- Service Technicians- Charging Station Technicians: These professionals are responsible for the upkeep and repair of the EV's. The IoT sensors and data collected by the EVCMS Application can provide these technicians with valuable insights into EV's performance and potential issues.

5g Other Stakeholders

- **Manufacturers of Charging Stations:** Companies that produce and supply the physical charging equipment used in the network of charging stations.
- **Software and Technology Vendors:** Vendors providing software solutions, including those responsible for the development of the EVCMS application itself or integrating technologies like IoT sensors and data analytics tools.
- **Payment Processing Providers:** Companies that offer payment gateway services to facilitate transactions for charging sessions within the app.
- **Electricity Providers:** Vendors that supply the electricity used at charging stations. They might collaborate with EVCMS to manage energy distribution, pricing, and load balancing.
- **Sponsor:** The organization or individual funding the EVCMS project.
- **Legal Experts:** Ensure the system complies with regulations and standards.

6 Mandated Constraints

6a Solution Constraints

The EVCMS use the following technologies as stated in 3c:

- **Frontend:** Flutter (latest stable version at the time of implementation) for developing cross-platform mobile applications.
- **Backend:** Python (Flask framework, version 2.3.2 or latest) for building the server-side logic.
- **Database:** SQLite (version 3.39.3 or latest) for data storage and management.
- **Payment Gateway Integration:** The system must integrate with Razorpay (version 2.9.4) to process transactions securely.
- **GPS and Mapping Services:** Integration with Google Maps API (version 3.x) for real-time location tracking and mapping.
- **Battery Management System (BMS):** Must utilize sensors for current, temperature, and voltage, with compatibility to the chosen microcontroller (MPC5775B and MPC5775E). Also MOSFETs manage the connection between battery cells and the load sensor or Charger.[2]

- **Rationale:**
  - **Flutter** is chosen for its ability to develop cross-platform mobile applications with a single codebase, which reduces development time and maintenance costs.
  - **Python (Flask)** is selected for its lightweight nature, ease of integration with various technologies, and the team's familiarity with the language and framework.
  - **SQLite** is chosen for its simplicity, ease of use, and ability to operate without a separate server process.

- **The specific payment gateway** is mandated due to the client's existing contractual agreements with the service provider.
- **Google Maps API** is mandated because it provides robust, reliable, and well-documented mapping services.
- **BMS Sensors** are essential for accurate monitoring and management of the battery's health, which is crucial for the performance of electric vehicles.

6b Implementation Environment of the Current System

The Electric Vehicle Charging Management System (EVCMS) will be deployed within an environment that integrates various automated, mechanical, organizational, and nonhuman systems. These systems work together to facilitate seamless EV charging operations, from locating charging stations to processing payments and managing energy use.

- **Technological Environment:**
  - **Frontend:**
    - **Technology:** Flutter (for mobile applications).
    - **Devices:** Android smartphones and tablets, which users will use to access the EVCMS app.
  - **Backend:**
    - **Technology:** Python (Flask framework).
    - **Database:** SQLite, chosen for its simplicity and ease of use in managing data locally on devices or small-scale server environments.
    - **Server:** As of now no server required. for larger deployments or those requiring high concurrency, we might consider transitioning

to a more scalable database solution like PostgreSQL as the system grows.

- **Communication**: The system shall use RESTful APIs for communication between the mobile app and the backend.

- **IoT Integration:**
  - **Devices:** IoT sensors (current, temperature, voltage, and load sensors) installed at charging stations.
  - **Communication Protocol:** Data from sensors will be transmitted to the backend server for optimization of charging sessions.

- **Physical Environment:**
  - **Charging Stations:** Equipped with IoT sensors and payment terminals.
  - **User Devices:** Users will interact with the EVCMS primarily through mobile devices, which must have Mapping Service and internet capabilities for locating charging stations and making reservations.
- **Nonhuman Adjacent Systems:**
  - **Payment Gateways:** Integrated with multiple online payment systems to process transactions securely.
  - **Mapping Services:** Utilized for real-time location tracking and providing 6c. Partner or Collaborative Applications

6c Partner or Collaborative Applications

- The EVCMS will collaborate with the following external applications:
  - **Google Maps API:** For providing location-based services and navigation.
  - **Payment Gateways:** Integration with Stripe or PayPal for processing payments.
- These applications provide essential services that are critical to the functionality of the EVCMS.

- The system shall successfully integrate with these applications and demonstrate smooth operation during testing and in production.

6d Off-the-Shelf Software

The Electric Vehicle Charging Management System (EVCMS) utilizes various off-the-shelf (OTS) software tools to enhance its functionality and streamline implementation. These tools include:

- Payment Gateway Integration(Razorpay): To handle transactions and manage multiple payment options, including credit/debit cards and digital wallets. These gateways ensure secure and reliable payment processing for charging services.
- Security Solutions-SSL/TLS Certificates: Purpose: To secure data transmission between the EVCMS application and its users, ensuring that sensitive information, such as payment details and personal data, is encrypted and protected against interception.
- Google Maps API: To provide users with accurate location services, route planning, and real-time navigation to nearby charging stations. These APIs integrate mapping features and enhance the user experience for locating and navigating to charging points.
- IOT Integration: Purpose: To connect and manage IoT sensors that monitor charging patterns, predict peak usage times, and optimize charging schedules.

References to Supplier's Specifications: Include links or references to the official documentation or technical specifications provided by the suppliers of the OTS components. For example:

- **SQLite Documentation**: https://www.sqlite.org/docs.html
- **Flutter SDK Specifications**: https://docs.flutter.dev/
- **IoT Sensor Specifications**: https://www.synopsys.com/glossary/what-is-a-battery-management-system.html

These elements ensure that all necessary details are included to guide the integration and use of off-the-shelf components in the EVCMS project.

6e Anticipated Workplace Environment

- Electric Vehicle Owners: At home, at charging stations, and on the road.
- Charging Station Managers: In office environments, monitoring multiple stations.
- Service Technicians: In outdoor and potentially harsh environments where the charging stations are located.
- Example: **Noisy Urban Environments**:
    - Charging stations in busy urban areas can be surrounded by traffic noise and other disturbances, making audible signals ineffective. Instead, the EVCMS should rely on visual notifications, such as LED indicators or smartphone alerts, to inform users of charging status or any issues.
    - The system should avoid relying on sound-based alerts and instead use vibrations for notifications.

6f Schedule Constraints

- Development Phase: The system must be fully developed within 12 months from the project start date.
- Testing Phase: The system shall undergo 3 months of testing and debugging.
- Deployment Deadline: The EVCMS must be deployed by the end of the calendar year to align with market demands.

What happens if we don't build the EVCMS by the end of the calendar year?

- **Operational Impact**: Without the EVCMS, electric vehicle (EV) owners may face difficulties in finding and reserving charging stations, leading to frustration and a potential decline in EV adoption in the target area.
- **Reputation Impact**: Delays could harm the reputation of the company developing the EVCMS, particularly if competitors release similar products on time.

What is the financial impact of not having the EVCMS by the beginning of the Christmas buying season?

- **Revenue Loss**: The Christmas buying season is a peak period for purchases, including new electric vehicles and related technologies.
- **Missed Market Opportunities**: The company may miss out on partnerships with EV manufacturers or retailers who are planning promotions or product launches tied to the holiday season.

6g Budget Constraints

- The budget for the Electric Vehicle Charging Management System (EVCMS) project is expressed in terms of financial resources, available technology, and human resources. The total budget allocation includes costs for software development, hardware procurement, integration of IoT sensors, testing, and deployment.
- The goal is to ensure that the EVCMS is both financially viable and meets the essential needs of users.

Note: Cost-Benefit Analysis: A thorough cost-benefit analysis should be conducted to determine which features provide the most value relative to their cost. Just mentioned about which things budget needs to be noted, not stated the approx. budget.

6h Feasibility Study

The feasibility study for the Electric Vehicle Charging Management System (EVCMS) confirms its viability from technical, operational, economic perspectives.

- Technical Feasibility: The system employs modern technologies such as Flutter for the frontend, Python with Flask for the backend, and IoT integration for real-time monitoring. It is designed to be scalable, secure, and compatible with various devices, ensuring a smooth user experience across platforms.

- Operational Feasibility: EVCMS streamlines critical operations such as dynamic scheduling, payment processing, and real-time alerts. It improves efficiency by automating tasks like reservation management, charging session monitoring, and predictive maintenance alerts, thus providing a seamless experience for users and service providers.

Costs:

- **Initial Development Costs**: Includes expenses for hiring developers, integrating payment gateways, and IoT sensor integration. This also covers system design, coding, testing, and deployment.
- **Infrastructure and Hosting**: Ongoing costs for server hosting as currently using SQLite, data storage, and system maintenance to ensure continuous operation.
- **Operational Costs**: Includes routine updates, bug fixes, and customer support services to address user issues and maintain system integrity.
- **Security Measures**: Costs associated with securing the system, such as purchasing SSL certificates, implementing encryption, and conducting regular security audits to protect user data.

Benefits:

- **Efficiency Gains**: Automates tasks like dynamic scheduling and charging session monitoring, significantly reducing manual intervention and improving overall operational efficiency.
- **Scalability**: Designed to accommodate the growing demand for electric vehicle charging, the system allows for easy expansion without requiring major reinvestments.
- **Personalization:** Features like favorite stations and predictive maintenance enhance user convenience.
- **Optimized Charging:** Predictive analytics and dynamic scheduling can reduce energy costs and manage grid load more effectively.

- **Reduced Downtime:** Predictive maintenance helps in minimizing unexpected issues and downtime.
- **Sustainability:** By promoting the use of electric vehicles and optimizing charging, the application supports a reduction in carbon emissions.
- **Energy Efficiency:** Optimized charging schedules can lower energy costs and potentially reduce operational costs for charging stations.
- **Flexibility:** Various payment methods cater to different user preferences and increase transaction convenience.

The EVCMS is a robust solution that not only meets current needs but also adapts to future requirements, making it a valuable investment for stakeholders in the electric vehicle ecosystem.

7 Naming Conventions and Definitions

7a Definitions of Key Terms

- **EVCMS (Electric Vehicle Charging Management System):** A software platform designed to manage and optimize the operation of electric vehicle (EV) charging stations, including user interactions, charging session management, and payment processing.
- **Charging Station:** A physical location equipped with one or more chargers for electric vehicles to recharge their batteries.
- **User Profile:** A collection of personal and vehicle information associated with a registered user of the EVCMS.
- **Reservation:** A confirmed booking of a specific time slot at a charging station made by a user through the EVCMS.
- **IoT Sensors (Internet of Things Sensors):** Devices used to collect data on various aspects of the charging process, such as power levels and station usage, which are integrated into the EVCMS for real-time monitoring and analysis.

- **Real-Time Data:** Information that is continuously updated and reflects the current status of charging stations and sessions.
- **Payment Gateway:** An online service that processes transactions and payments made by users for charging services.
- **Predictive Maintenance:** Techniques used to anticipate and address potential issues with charging stations or infrastructure before they cause failures.
- **Route Planning(EN route and ON route):** The process of determining the optimal travel route for users, including necessary charging stops, based on their input and available charging infrastructure.
- **Charging Session:** The period during which an electric vehicle is connected to a charging station and receives power.

7b UML and Other Notation Used in This Document

UML Notations:

- **Class Diagram:** Represents the static structure of the system including classes, attributes, and relationships.
- **Use Case Diagram:** Illustrates the interactions between users (actors) and the system, showing different functionalities or use cases.
- **Sequence Diagram:** Shows the sequence of interactions between objects in the system for a particular use case.
- **Activity Diagram:** Depicts the workflow of a process or system, including actions and decisions.

Symbols and Notations:

- **Solid Arrow:** Represents a direct association or relationship between classes or components.

- **Hollow Arrow:** Indicates an inheritance or generalization relationship in class diagrams.
- **Diamond Shape:** Denotes aggregation or composition relationships.

7c Data Dictionary for Any Included Models

Data Flows and Stores:

- **User Registration Data:** Personal information collected during user sign-up, including name, email, and vehicle details.
- **Charging Station Availability:** Real-time data indicating the status and availability of each charging station.
- **Payment Transaction Data:** Details of payments processed, including amount, transaction ID, and payment method.
- **Session Monitoring Data:** Information from IoT sensors about the completion and next-charging Schedule of a charging session, including energy management to reduce grid load.

Technical Specifications:

- **API Interfaces:** Specifications for integration with external systems, such as payment gateways and third-party apps.
- **Database Schemas:** Definitions for storing user profiles, reservation details, and charging station data.

8 Relevant Facts and Assumptions

8a Facts:

- **Market Penetration:** The number of electric vehicles is increasing, leading to higher demand for charging infrastructure and increase in growth of EVCMS.

- **Existing Infrastructure:** Many urban areas have existing charging stations that need to be integrated into the EVCMS.

8b Assumptions:

- **Technology Compatibility:** It is assumed that the charging stations will support standard communication protocols for integration with the EVCMS.
- **Data Security:** The system will have access to secure and reliable data storage and transmission technologies to protect user and transaction data.
- **User Adoption:** Users are expected to have basic familiarity with smartphone apps or web interfaces for interacting with the EVCMS.

What software tools are you expecting to be available?

- **Development and Integration Tools:**
    - **Integrated Development Environments (IDEs):** Tools like **Visual Studio**, **Eclipse**, or **IntelliJ IDEA** for coding and debugging.
    - **Version Control Systems: Git** (with platforms such as **GitHub** or **GitLab**) for source code management and collaboration.
    - **Database Management Systems (DBMS): MySQL**, **PostgreSQL**, or **MongoDB** for managing user data, charging station details, and reservations.
    - **API Development Tools: Postman** for developing and testing APIs.
    - **Project Management Tools: Jira** for tracking development progress and managing tasks.

Will there be any new software products?

- **Custom EVCMS Modules:** Specific modules or features, such as custom reservation systems, real-time charging status updates, or advanced analytics, might be developed.
- **Integration with Emerging Technologies:** Potential integration with new technologies like blockchain for secure transactions or advanced AI for predictive maintenance and energy management.
- **User Experience Enhancements:** New software products or third-party services to improve user interfaces, such as augmented reality for locating charging stations.

Are there any business changes you are assuming we will be able to deal with?

- **Increased EV Adoption:** The assumption that the number of electric vehicles will continue to grow, leading to a higher demand for charging infrastructure and services.
- **Partnerships and Collaborations:** Potential changes in business partnerships with charging station providers, vehicle manufacturers, or energy companies.

II Requirements

9 SRS (Software Requirements Specifications)

Software Requirement Specification (SRS) Format as the name suggests, is a complete specification and description of requirements of the software that need to be fulfilled for the successful development of the software system. These requirements can be functional as well as non-functional depending upon the type of requirement.[1]

9a Functional Requirements

Functional requirements define the specific behaviors and functions of the EVCMS. They describe what the system should do.

1. **User Management:**
    - **User Registration:** The system allow users to register by providing personal and vehicle information.
    - **User Login:** The system authenticate users via username and password.
    - **User Profile Management:** Users should be able to view and update their personal and vehicle details.
2. **Charging Station Management:**
    - **Search for Charging Stations:** The system allow users to search for nearby charging stations based on their current location.
    - **Charging Station Details:** The system provides detailed information about each charging station, including availability, types of chargers, and location.

3. **Reservation System:**

    - **Schedule Charging Sessions:** Users must be able to reserve time slots at charging stations.
    - **Real-Time Updates:** The system provide real-time updates on charging station availability and reservation status.

4. **Charging Session Management:**

    - **Initiate Charging:** The system start a charging session based on user authentication and reservation.
    - **Monitor Charging:** The system monitor the status of the charging session, including power levels to reduce grid load and session duration.
    - **End Charging Session:** The system stop the charging session once completed and process the payment and notify the EV owner.

5. **Payment Processing:**

    - **Payment Collection:** The system handle payments for charging sessions through integrated payment gateways.
    - **Generate Receipts:** The system provide users with payment receipts via email.

6. **Notifications and Alerts:**

    - **Session Notifications:** The system send notifications to users regarding the start and end of charging sessions, including alerts for any issues.
    - **Maintenance Alerts:** The system notify users of scheduled maintenance or next-charge Schedule with maintenance.

7. **Feedback Collection:**

    - **User Feedback:** After a charging session, the system prompt users to provide feedback and collect this data for analysis.

8. **Energy Management:**

    - **Energy Usage Analysis:** The system analyzes energy consumption data to reduce grid load and recommendations for optimizing charging schedules.

9. **Route Planning(EN Route and ON route):**

    - **Plan Routes with Charging Stops:** The system allow users to input a travel route and suggest optimal charging stops along the way.

9b Non-Functional Requirements

Non-functional requirements specify how the system performs its functions and includes quality attributes such as performance, reliability, and usability.

1. **Payment gateways for payments(Razorpay).**

2. **Data Storage:**

    - **Database Design:** The system should use a relational or NoSQL database to store user profiles, charging station data, reservations, and transaction details.

    - **Data Backup:** Regular backups must be implemented to ensure data integrity and recovery in case of failures.

3. **Algorithm:**
    - Searching and Shortest Path Algorithm to search nearest charging station.

4. **Notifications:**

    - Notifying users for Maintenance of EV, completion of Charging Ev and Next Schedule for charging. Sends notifications to users regarding reservation confirmations, payment receipts.

5. **Performance:**

    - **Response Time:** The system should provide search results and update availability in real-time, with a response time of less than 2 seconds.

6. **Error Handling:** The system should gracefully handle errors and provide meaningful error messages to users.

7. **User Interface:** The system must have an intuitive and user-friendly interface for both mobile and web applications.

8. **Security:**

    - **Data Protection:** The system must encrypt user data and payment information to protect against unauthorized access and breaches.
    - **Authentication:** The system must implement secure authentication mechanisms, including support for multi-factor authentication.

9c Design Requirements

Design requirements outline the architectural and technical aspects that must be considered in the system's design.

1. **Architecture:**

    - **Service-Oriented Architecture (SOA):** Consider using SOA principles for integration with external systems such as payment gateways and third-party APIs.

2. **Integration:**

   - **API Integration:** The system should integrate with external services (e.g., payment gateways, map services) through well-defined APIs.
   - **IoT Integration:** The system must interface with IoT sensors at charging stations to collect real-time data on charging sessions.

3. **Secure Communication:**
   - Use HTTPS for secure communication between clients and servers.

4. **User-Friendly Interface:**

   - **Fast and Responsive:** Users expect the system to be fast and responsive, providing quick search results for charging stations, prompt reservation confirmations, and real-time updates without delays.
   - **Responsive Design:** The user interface should be responsive and optimized for different devices, including smartphones, tablets, and desktops.
   - **Intuitive Navigation:** Users should find the interface easy to navigate, with clear options for managing their profiles, making reservations, and checking charging statuses.
   - **Responsive Design:** The system should work effectively across various devices, including smartphones, tablets, and desktops, adapting its layout for optimal usability on each device.

9d Recommended Hardware and Software requirement analysis

1. **Hardware Requirements:**

   - Processor minimum 1.3GHz.
   - 4GB of the system Memory.

- Keyboard
- Mouse
- Colour Monitor
- Charging Stations should have BMS(Battery Management Sensor having temperature, voltage, current ad load sensor)

2. Software Requirements:

- Front End : Flutter
- Back End : Python(Flask)
- Database: SQLite
- Web Browser : Any
- Operating System : Recommended(Windows 10 or 11)

10 SDLC Model

The goal of the SDLC life cycle model is to deliver high-quality, maintainable software that meets the user's requirements. SDLC in software engineering models outlines the plan for each stage so that each stage of the software development model can perform its task efficiently to deliver the software at a low cost within a given time frame that meets users' requirements.

- **The 7 Phases Of SDLC (Software Development Life Cycle)[3]**

Stage 1: Project Planning

The first stage of SDLC is all about "What do we want?" Project planning is a vital role in the software delivery lifecycle since this is the part where the team estimates the cost and defines the requirements of the new software.

Stage 2: Gathering Requirements & Analysis

The second step of SDLC is gathering maximum information from the client requirements for the product. Discuss each detail and specification of the product with the customer. The development team will then analyze the requirements keeping the design and code of the software in mind. Further, investigating the validity and possibility of incorporating these requirements into the software system. The main goal of this stage is that everyone understands even the minute detail of the requirement. Hardware, operating systems, programming, and security are to name the few requirements.

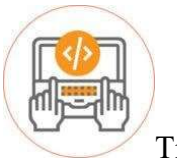
Stage 3: Design

In the design phase (3rd step of SDLC), the program developer scrutinizes whether the prepared software suffices all the requirements of the end-user. Additionally, if the project is feasible for the customer technologically, practically, and financially. Once the developer decides on the best design approach, he then selects the program languages like Oracle, Java, etc., that will suit the software.

Once the design specification is prepared, all the stakeholders will review this plan and provide their feedback and suggestions. It is absolutely mandatory to collect and

incorporate stakeholder's input in the document, as a small mistake can lead to cost overrun.

Stage 4: Coding or Implementation



Time to code! It means translating the design to a computer-legible language. In this fourth stage of SDLC, the tasks are divided into modules or units and assigned to various developers. The developers will then start building the entire system by writing code using the programming languages they chose. This stage is considered to be one of the longest in SDLC. The developers need certain predefined coding guidelines, and programming tools like interpreters, compilers, debugger to implement the code.

The developers can show the work done to the business analysts in case if any modifications or enhancements required.

Stage 5: Testing

Once the developers build the software, then it is deployed in the testing environment. Then the testing team tests the functionality of the entire system. In this fifth phase of SDLC, the testing is done to ensure that the entire application works according to the customer requirements.

After testing, the QA and testing team might find some bugs or defects and communicate the same with the developers. The development team then fixes the bugs and send it to QA for a re-test. This process goes on until the software is stable, bug-free and working according to the business requirements of that system.

Stage 6: Deployment

The sixth phase of SDLC: Once the testing is done, and the product is ready for deployment, it is released for customers to use. The size of the project determines the complexity of the deployment. The users are then provided with the training or documentation that will help them to operate the software. Again, a small round of testing is performed on production to ensure environmental issues or any impact of the new release.

Stage 7: Maintenance

The actual problem starts when the customer actually starts using the developed system and those needs to be solved from time to time. Maintenance is the seventh phase of SDLC where the developed product is taken care of. According to the changing user end environment or technology, the software is updated timely.

10a SDLC Model Used-Agile Model

- SDLC (Software Development Life Cycle) Model used for EVCMS(Electric Vehicle Charging Management System)- **Agile Model**
- The **Agile model** is a flexible and iterative approach to software development that emphasizes continuous delivery, collaboration, and adaptability. It breaks the project into small, manageable increments called sprints, typically lasting two to four weeks. Each sprint involves planning, development, testing, and review, allowing teams to regularly assess and adjust the project based on feedback and changing requirements. Agile focuses on delivering functional software quickly and iteratively, engaging stakeholders throughout the process to ensure that the final product meets their needs and expectations.

Advantages for EVCMS:

- **Flexibility:** Allows for frequent changes and adaptations based on evolving requirements and user feedback.
- **User Involvement:** Encourages continuous feedback from users, ensuring the system meets their needs and expectations.
- **Rapid Delivery:** Delivers functional increments quickly, providing early value and enabling the system to adapt to emerging needs or changes in technology.

How to Implement Agile model?

- **Sprint Planning**: Plan development in sprints with defined goals for each iteration, allowing regular adjustments based on feedback.
- **User Stories**: Develop features based on user stories to ensure that the application meets user needs effectively.
- **Continuous Integration**: Regularly integrate and test new features to maintain high quality and address issues promptly.
- **Stakeholder Engagement**: Involve stakeholders in review meetings to align the development with business goals and user expectations.

10b Why Agile is Best Suitable for EVCMS?

1. Dynamic Requirements:

- The EVCMS Application deals with rapidly evolving technology and user needs. The Agile model allows for flexibility and iterative development, accommodating changes in requirements as the project progresses.

2. Frequent Updates:

- Agile emphasizes continuous improvement and frequent releases. This aligns with the need to incorporate feedback on user experience, update features, and adapt to new technologies in the EV sector.

3. User-Centric Approach:

- Agile involves regular stakeholder feedback and user testing. This is crucial for a system that aims to improve user experience through features like real-time reservations, dynamic scheduling, and route planning.

4. Integration with IoT:

- The Agile model supports iterative development and integration. As the application integrates IoT sensors for monitoring and analysis, Agile allows for incremental integration and testing of these technologies.

5. Complex Features:

- With features like real-time notifications, dynamic scheduling, and support for multiple payment options, Agile's iterative approach helps in managing complex functionalities by breaking them into smaller, manageable components.

6. Adaptation to Feedback:

- As new trends in EV technology and user preferences emerge, Agile's iterative cycles enable the development team to adapt and refine the application based on user feedback and market trends.

7. Risk Management:

- Agile promotes early and frequent testing, which helps in identifying and addressing potential issues early in the development process, reducing overall risk.

By choosing the Agile model, you can ensure that the EVCMS Application evolves effectively, remains adaptable to changing needs, and consistently meets user expectations.

10c Why Other Models weren't used for EVCMS?

1. Waterfall Model: The Waterfall model is a linear, sequential approach where each phase (requirements, design, implementation, testing, deployment) must be completed before moving to the next.

Challenges for EVCMS:

- **Inflexibility:** Difficult to accommodate changes once a phase is completed, which can be problematic for projects with evolving requirements.
- **Late Testing:** Testing occurs only after development is complete, which may lead to late discovery of issues.

2. Spiral Model: The Spiral model combines iterative development with a focus on risk assessment and management, involving cycles of planning, risk analysis, engineering, and evaluation.

Challenges for EVCMS:

- **Complexity:** Can be complex to manage due to its multiple phases and detailed risk management requirements.
- **Cost:** Potentially higher costs due to extensive planning and risk assessment activities.

3. Iterative and Incremental Model: This model involves developing the system in increments or iterations, with each iteration adding functionality and refining the system based on feedback.

Challenges for EVCMS:

- **Scope Creep:** Frequent iterations can lead to scope creep if not managed properly.
- **Integration:** Each increment needs thorough testing and integration, which can be challenging.

4. Prototyping Model: The Prototyping model involves creating prototypes or preliminary versions of the system to gather user feedback and refine requirements.

Challenges for EVCMS:

- **Scope Creep:** Continuous changes based on feedback can lead to scope creep.
- **Representativeness:** Prototypes may not always accurately represent the final system's performance.

5. RAD (Rapid Application Development) Model: RAD focuses on rapid prototyping and iterative development to quickly deliver functional software with frequent user feedback.

Challenges for EVCMS:

- **Speed vs. Thoroughness:** Rapid development can sometimes compromise thoroughness in testing and documentation.
- **Scalability:** May face challenges in scaling and maintaining the system if not managed carefully.

11 Product Use Cases

A use case diagram is a representation of a user's interaction with the system. [4]

- **System Boundary**: Defines the limits of the system.
- **Actors**: Who interacts with the system.
- **Use Cases**: Scenarios of interaction to achieve goals.
- **Association**: Links actors to use cases.
- **Generalization**: Hierarchical relationships to reuse definitions.
- **Extend**: Optional behaviors added to use cases.
- **Include**: Required behaviors shared across use cases.

11a Use Case Diagrams

A **Use Case Diagram** visualizes the interaction between users (actors) and the system, breaking down the main functionalities into use cases. For the **Electric Vehicle Charging Management System (EVCMS)**, the key components include:

1. Actors:

- **EV Owner**: Primary user who interacts with the system to find charging stations, book slots, and make payments.
- **EV Charging Station Manager**: Responsible for managing the operations of charging stations, including monitoring availability and scheduling maintenance.
- **Service or Maintenance Staff**: Handles station maintenance tasks and receives notifications for predictive maintenance.
- **Sensors**: IoT sensors installed at charging stations and EVs to monitor energy usage, charging status, and detect anomalies.
- **Admin**: Oversees the overall system, managing users, payment records, and system configurations.

2. Use Cases:

- **User Registration & Login**: Allows EV owners to register, log in, and access the system.
- **Find Nearest Charging Station and Reserve a Charging Slot**: Uses GPS to locate nearby charging stations, displaying relevant details such as availability and pricing. Allows users to book a charging station in advance. Facilitates secure payment for charging services and generates invoices.
- **Favorite Charging Stations**: Enables users to save frequently visited stations for quick access and rebooking.
- **Predictive Maintenance**: IoT sensors trigger alerts when maintenance is required, scheduling it based on sensor data.

- **Manage Charging Stations**: Charging station managers can monitor and manage station status, availability, and maintenance schedules.
- **Generate Reports**: Admins generate usage, payment, and maintenance reports.

3. System Boundary:

- The system boundary defines the scope of the EVCMS and separates the internal functionalities from the external actors (users, staff, sensors).

4. Relationships:

- **Association** (solid line): EV Owner interacts with the "Find Nearest Charging Station" use case.
- **Include** (dashed arrow with <<include>>): "Payment Processing" might be included in "Reserve a Charging Slot".
- **Extend** (dashed arrow with <<extend>>): "Favorite Charging Stations" extends "Reserve a Charging Slot" for frequently used stations.
- **Generalization** (solid arrow with an empty arrowhead): Admin is a specialized version of EV Charging Station Manager, with additional permissions.

USE CASE DIAGRAM:

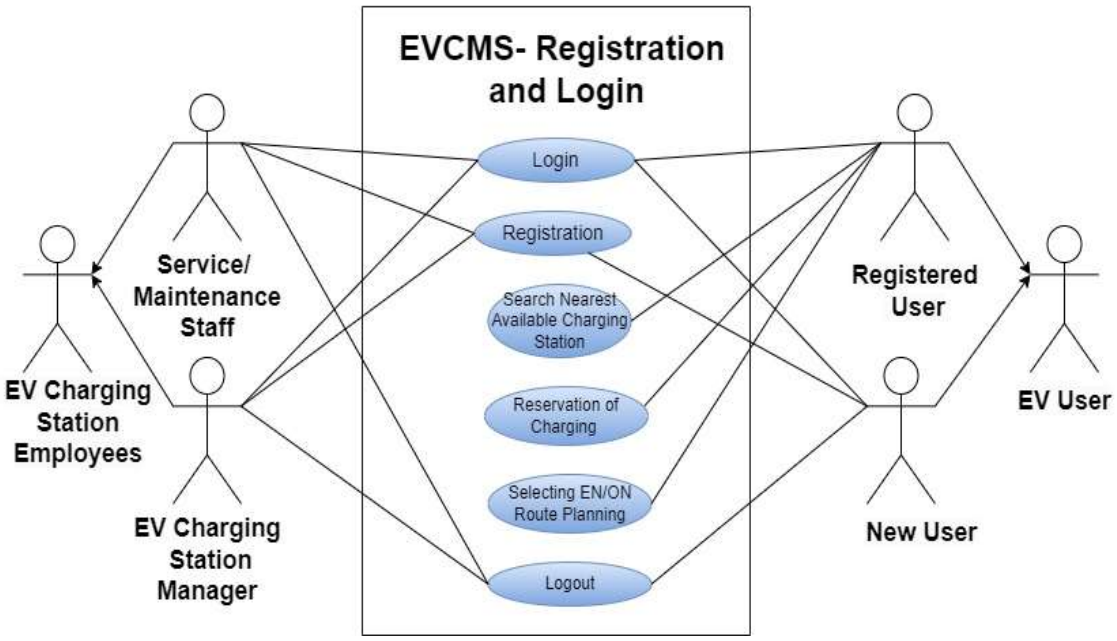


Figure 2 – UseCase Diagram for Registration and Login:

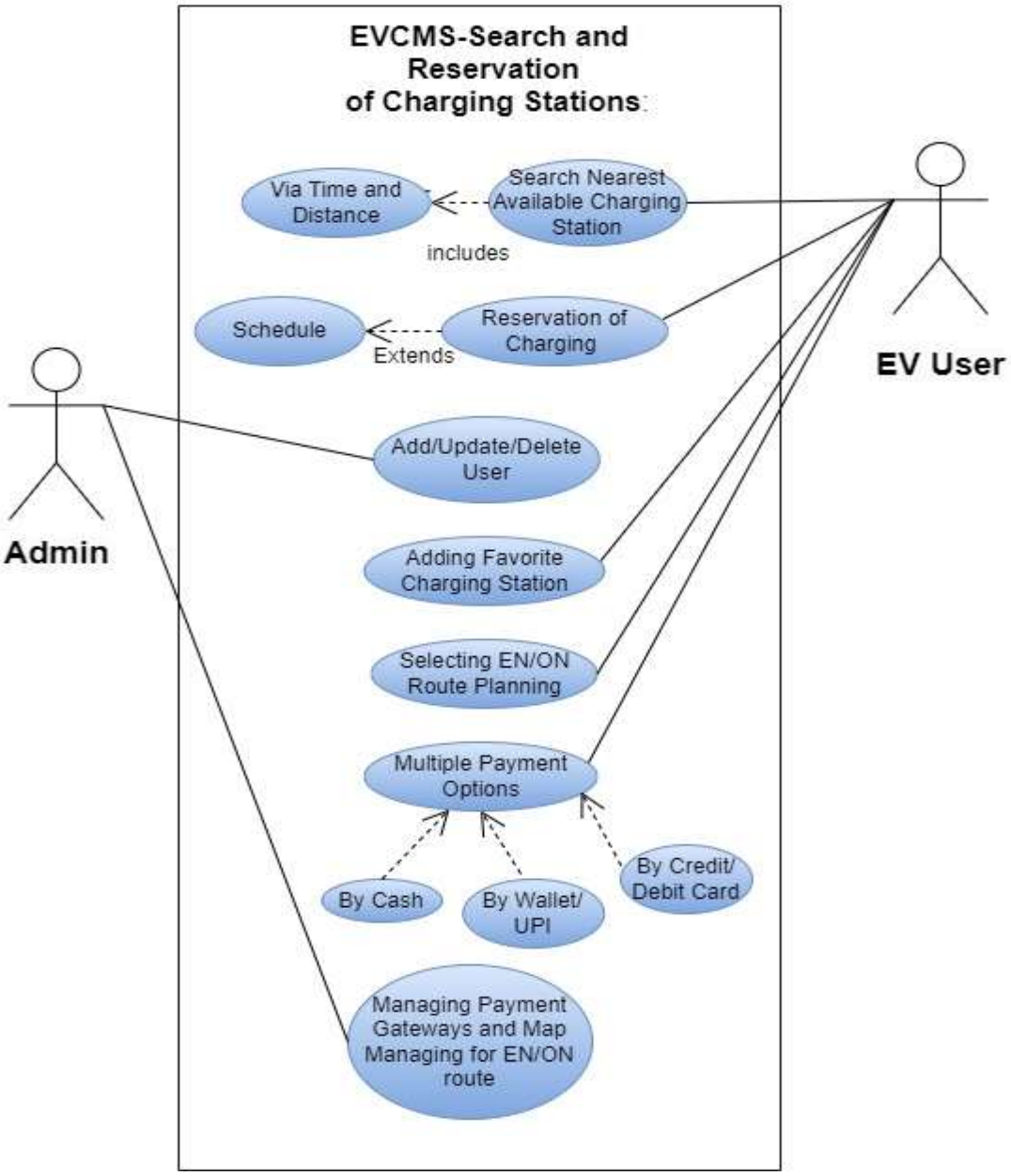


Figure 3 – UseCase Diagram for Search and reserve Charging Stations:

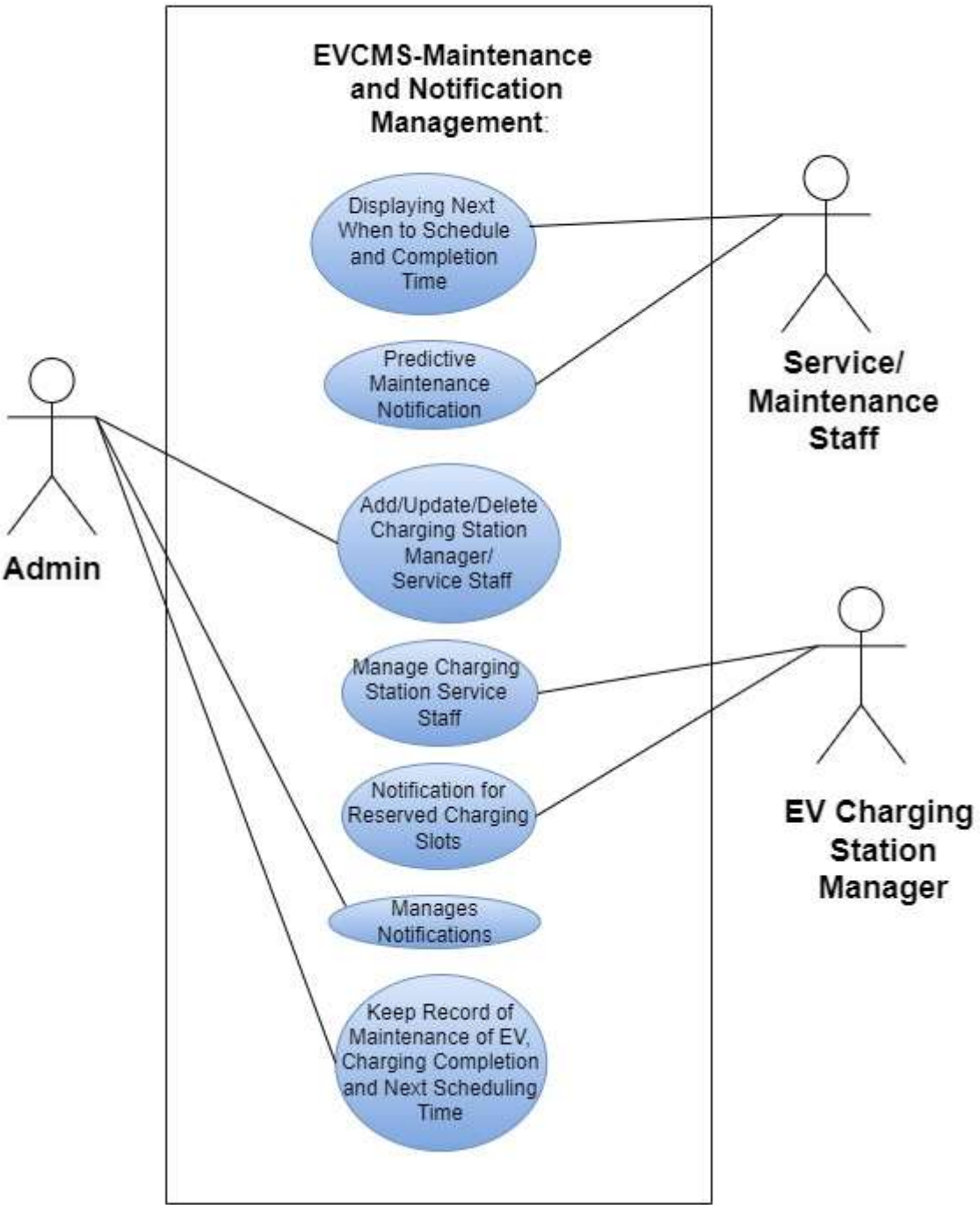


Figure 4 – UseCase Diagram for Maintenance and Notification Management:

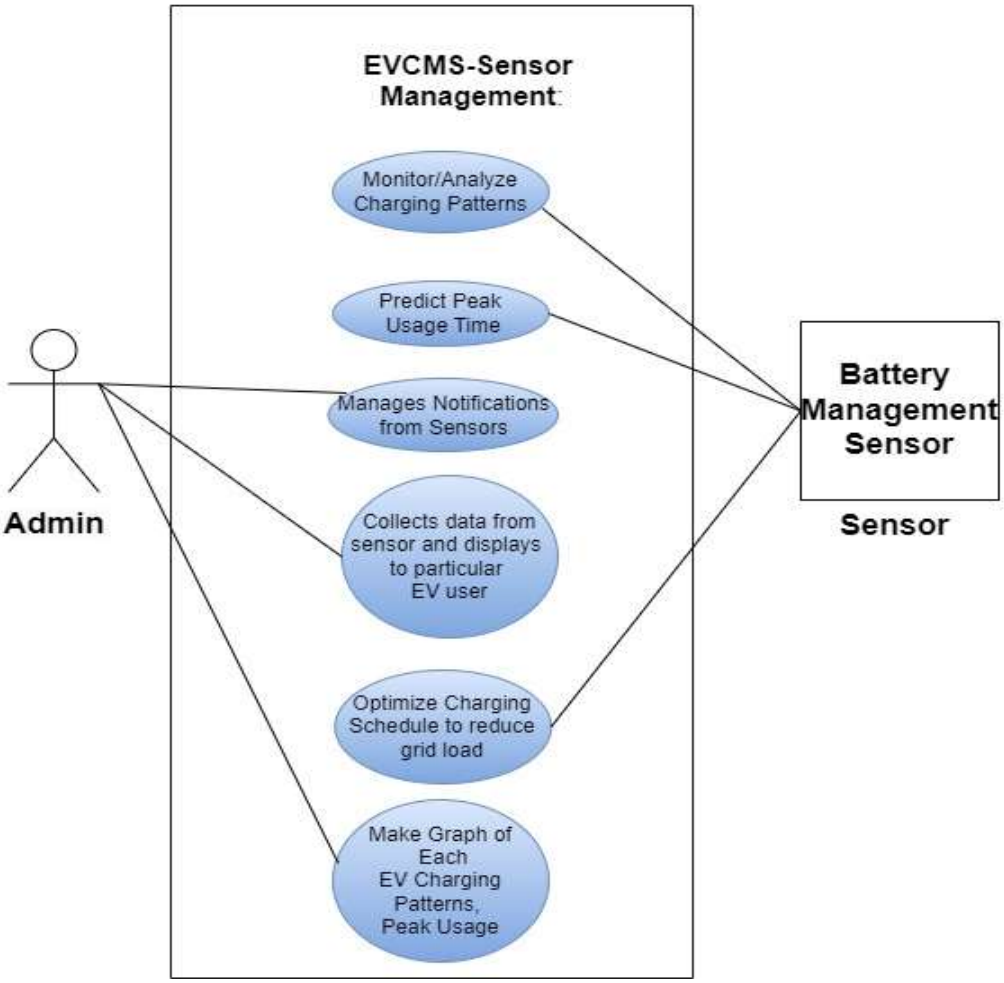


Figure 5 – UseCase Diagram for Sensor Management:

USE CASE DIAGRAM(Generalized):

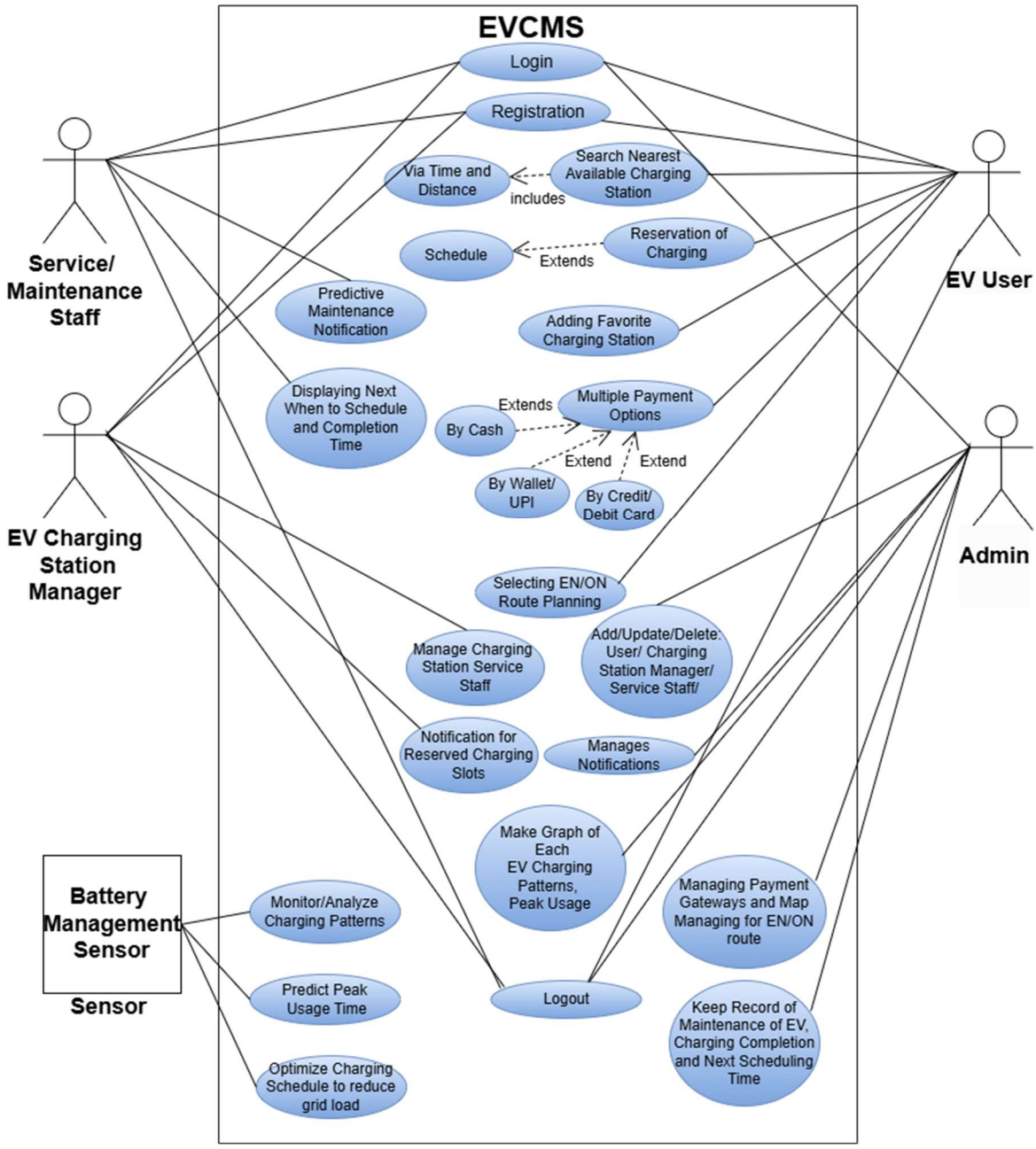


Figure 6 – Generalized UseCase Diagram:

11b Product Use Case List(For Generalized UseCase Diagram)

- **User Registration:** EV User, Admin, EV Charging Station Manager, and Service Staff can register into the system.
- **Login:** All actors can log in to the system.
- **Search Nearest Available Charging Station:** EV Users can search for the nearest charging stations based on their location and time/distance.
- **Reservation of Charging Station:** EV Users can reserve a charging slot at a selected charging station.
- **Adding Favorite Charging Stations:** EV Users can add charging stations to their favorites for quick access.
- **Payment Options:** EV Users can make payments using various methods:
  - By Cash
  - By Wallet/UPI
  - By Credit/Debit Card
- **Predictive Maintenance Notification:** Service/Maintenance Staff can receive notifications about predictive maintenance for the charging stations.
- **Displaying Next Charging Schedule and Completion Time:** The system displays the next available charging schedule and completion time for EV Users.
- **Manage Charging Station Service Staff:** Charging Station Managers can manage service staff associated with their stations.
- **Notification for Reserved Charging Slots:** Charging Station Managers can manage notifications for reserved slots to ensure efficient service.
- **Monitor and Analyze Charging Patterns:** Battery Management Sensors collect data on charging patterns to be monitored and analyzed.
- **Predict Peak Usage Time:** The system predicts peak usage times based on historical data from Battery Management Sensors.
- **Optimize Charging Schedule:** Based on peak usage analysis, the system optimizes the charging schedule to reduce grid load.

- **Make Graph of Each EV Charging Patterns and Peak Usage:** Charging Station Managers or Admins can generate graphs showing EV charging patterns and peak usage.
- **EN/ON Route Planning:** EV Users can plan their routes based on charging station availability along the way.
- **Managing Charging Stations (Add/Update/Delete):** Admins can manage charging stations by adding, updating, or deleting them from the system.
- **Manage Users and Service Staff:** Admins can add, update, or delete EV Charging Station Managers and Service/Maintenance Staff.
- **Manage Payment Gateways:** Admins can manage the integration of payment gateways for smooth transactions.
- **Keep Record of Maintenance:** Admins keep records of vehicle maintenance schedules and completion times for future planning.
- **Notification Management:** Charging Station Managers can manage notifications for service, maintenance, and reserved slots.
- **Monitor Charging Sessions:** IoT sensors continuously monitor charging sessions and relay the data to the system.
- **Logout:** All actors can log out of the system after completing their tasks.

12 Data Requirements

Data Models:

1. **EV Owners Attributes:**

    - User ID: Unique identifier for each EV owner.
    - Name: Full name of the user.
    - Email: Contact email of the user.
    - Vehicle Details: Information such as make, model, and battery capacity.
    - Charging History: Logs of previous charging sessions (time, date, station used).

2. **Charging Stations Attributes:**

    - Station ID: Unique identifier for each charging station.
    - Location: GPS coordinates or address of the charging station.
    - Availability: Status of the charging station (e.g., available, occupied, out of service).
    - Price: Charging cost per kWh or session.
    - Connector Types: Information about supported connector types (e.g., CCS).

3. **Reservations Attributes:**

    - Reservation ID: Unique identifier for the reservation.
    - Station ID: Reference to the station where the slot is reserved.
    - User ID: Reference to the user who made the reservation.
    - Time Slot: Start and end times for the reservation.
    - Status: Status of the reservation (e.g., pending, confirmed, canceled).

4. **Payments Attributes:**

    - Payment ID: Unique identifier for the payment transaction.
    - Amount: Total amount paid.
    - Method: Payment method used (cash, credit card, UPI).
    - User ID: Reference to the user who made the payment.
    - Reservation ID: Reference to the reservation for which the payment was made.

5. **IoT Sensors Attributes:**

    - Sensor ID: Unique identifier for the IoT sensor.
    - Station ID: Reference to the charging station to which the sensor is attached.
    - Sensor Type: Type of sensor (e.g., temperature, voltage).
    - Data: Recorded data from the sensor, such as real-time voltage or temperature readings.

6. **Battery Management System (BMS) Attributes:**

    - System ID: Unique identifier for the BMS system.
    - Battery Health: Health status of the battery.
    - Energy Usage: Detailed log of energy consumption over time, usage patterns.

13 Performance Requirements

13a Speed and Latency Requirements

- The EVCMS shall return a list of available charging stations within a maximum response time of 2 seconds when a user searches for nearby stations.

- Upon submitting a reservation request, the system shall confirm the reservation within 3 seconds(assumption).
- The EVCMS shall check and update charging slot availability every 10 seconds during peak hours.
- The payment system shall process transactions within a maximum of 5 seconds to ensure a smooth user experience at the charging station.

- Searching for and reserving charging stations during travel requires immediate responsiveness to avoid user frustration and potential downtime.
- Updating charging preferences or accessing infrequent reports may allow for slightly longer response times, but should still aim to maintain a responsive user experience.

13b Precision or Accuracy Requirements

Precision Requirements

- The GPS coordinates of charging stations shall be accurate to within ±5 meters to ensure users can easily locate them.

- The EVCMS shall report charging station availability with a precision of 95%, meaning that the reported status matches the actual status at least 95% of the time.
- The EVCMS shall provide reservation details (e.g., connection type, model compatibility) with precision to ensure users have exactly what they need for their charging sessions, with 100% adherence to the specified parameters.

Accuracy Requirements

- The details (e.g., charging speed, pricing) provided for each charging station shall be accurate to within ±10% of the actual values to avoid misleading users.
- Estimated completion times for charging sessions shall be calculated with an accuracy of ±5 minutes to ensure users can effectively plan their schedules.
- High precision and accuracy are vital for building user trust in the EVCMS Application, particularly for real-time functionalities.

13c Capacity Requirements

- The EVCMS shall support a minimum of 10,000 concurrent users during peak hours without degradation in performance.
- The EVCMS shall support a database of at least 20,000 charging stations, including detailed information such as location, availability, connection types, and pricing.
- The EVCMS must handle updates to charging station status (e.g., availability changes) in real-time, accommodating at least 1,000 updates per minute during peak usage.
- IoT sensor readings and charging history must be capable of holding 10 years of data without performance degradation.

14 Dependability Requirements

14a Reliability Requirements

- **Availability**: The EVCMS shall maintain an uptime of 99.9% over any given month, ensuring that users can access the service consistently.

- **Downtime**: Any planned maintenance should be communicated in advance and should not exceed 2 hours per month.
- **Data Consistency**: The EVCMS shall ensure that charging station information and user data remain consistent across all platforms (mobile, web) with a consistency rate of 99.5%.
- **Backup Frequency**: The EVCMS must perform data backups at least once every 24 hours to prevent data loss.
- **Graceful Degradation**: In the event of a failure, the EVCMS should provide a fallback mode, allowing users to access critical functionalities (e.g., viewing nearby stations) even when some services are unavailable.

14b Availability Requirements

- The EVCMS shall be available **24/7**, 365 days a year, to accommodate charging needs at any time.
- The EVCMS shall maintain a **99.5% uptime** to ensure availability during critical times.
- High availability is crucial for users who rely on the system for timely access to charging stations, especially in emergencies or during long trips.
- Planned maintenance should be scheduled during off-peak hours and communicated to users at least 48 hours in advance.

14c Robustness or Fault-Tolerance Requirements

- In the event of a **power outage**, the EVCMS should provide **10 minutes of emergency operation** to allow ongoing charging sessions to be safely terminated.
- The EVCMS shall degrade gracefully in the event of a failure, maintaining core functionalities (e.g., searching for charging stations) even if some features are temporarily unavailable.
- Users should receive clear notifications regarding any issues or degraded performance, including alternative options available during outages.

- The EVCMS shall perform automated backups of all critical data +every 24 hours, with the capability to restore to the last backup point within 1 hour of any data loss incident.
- Database connections should include failover mechanisms that automatically switch to a backup database in case of primary database failure.

14d Safety-Critical Requirements

- Charging stations must be designed to **automatically disconnect** in case of overload or electrical faults to avoid harm to vehicles and users.
- **Temperature sensors** should trigger an alert if the temperature rises above safe operating limits, such as **+45°C**, and automatically cut off charging to prevent overheating.

- Safety is critical in avoiding harm to users, their vehicles, and the environment, especially considering the risks associated with electricity and high voltages used in EV charging.

- The EVCMS must comply with international **electrical safety standards** for EV charging.[5]

15 Maintainability and Supportability Requirements

15a Maintenance Requirements

- The EVCMS must allow new charging stations to be added with minimal disruption, ideally within one working day of receiving the configuration information.
- Regular updates and additions are expected as new stations are installed and as the system grows. The ability to quickly integrate new components or fix issues is crucial to ensuring the system remains up-to-date and functional.
- All updates should be documented with version control to track changes, allowing for rollback if critical issues arise.

- Maintain logs of user activity for at least 6 months to analyze usage patterns, detect anomalies, and support troubleshooting.

15b Supportability Requirements

- The EVCMS shall have integrated error-reporting tools that automatically notify administrators of any technical problems, allowing for proactive maintenance.
- Establish a dedicated helpdesk available around the clock 24/7 to assist users with technical issues and inquiries.
- Response Time Standards:
  - Non-critical issues: Response within 24 hours.
  - Critical issues (e.g., payment failures): Response within 1 hour.
- Provide accessible online user manuals covering functionalities, troubleshooting steps, and FAQs.
- Maintain up-to-date technical documentation for support and development teams, including system architecture, APIs, and known issues.
- Incorporate feedback tools within the application to capture user experiences, suggestions, and issues.

15c Adaptability Requirements

- The EVCMS must be adaptable to different environments and platforms as the system expands geographically or technologically.
- be adaptable to run on multiple platforms, including mobile, web-based applications, and embedded systems within the charging stations.
- The EVCMS must be designed for multi-language support, enabling use in different countries as the service expands globally.
- The EVCMS shall allow users to customize their interface preferences (e.g., themes, layouts) to enhance their experience.
- Users should be able to configure dynamic dashboards to display relevant information, such as favorite charging stations or usage statistics.

15d Scalability or Extensibility Requirements

- The EVCMS must be capable of handling **500,000 active users** within 3-4 years of operation.
- It should scale to support **1,000 charging stations** and process up to **20,000 transactions per hour** during peak periods.
- As the demand for electric vehicles increases, the system must be able to handle larger user loads and more data without degradation in performance.

15e Longevity Requirements

- The system is expected to operate with routine maintenance for **a minimum of 10 years**.
- It should be able to incorporate new technologies, such as **wireless charging** during its lifetime.
- Planning for hardware and software updates during the system's lifetime ensures that it stays relevant and can adapt to future demands.

16 Security Requirements

16a Access Requirements

- Users must create an account with secure login (email/password).
- Admin should have elevated permissions to manage charging station listings, user accounts, and application settings.
- Charging Station Managers and Maintenance staff: Ability to list new charging stations and manage existing listings.
- Access to GPS/location data to provide real-time information about nearby charging stations.

- Access to data from IoT sensors at charging stations for real-time monitoring of availability and usage patterns.
- Permissions for data analytics to predict peak usage times and optimize scheduling to reduce grid load.

16b Integrity Requirements

- The EVCMS (Electric Vehicle Charging Management System) will implement comprehensive integrity measures to ensure the accuracy, reliability, and protection of its databases, files, and the overall system. The following specifications outline the required integrity for data handling and system processes.
- The EVCMS shall prevent incorrect or invalid data from being entered into the system through rigorous validation checks during user input.
- The EVCMS shall implement regular automated backups of all critical data to prevent data loss due to corruption, accidental deletion, or system failures.
- Personal user data and payment information will be encrypted using industry-standard encryption algorithms.

16c Privacy Requirements

- Encryption for data transmitted from IoT sensors to prevent interception.
- Secure APIs for accessing on-route data to prevent unauthorized data access.
- Ensure that the list of favorite stations is securely stored and not modifiable by unauthorized users.
- Encrypt sensitive user data stored in the database and ensure secure connections (e.g., TLS/SSL) for data transmission to protect against unauthorized access.

- Implement strict access controls to ensure that only authorized personnel can access sensitive user data.

- Only share user data with third parties if necessary and with user consent.

16d Audit Requirements

- User Activity Logs: The system must retain records of all user activities, including:
  - Logins and logouts
  - Search queries for charging stations
  - Reservation requests and cancellations
  - Payment transactions and methods used
- Maintain logs of all reservation requests, modifications, and cancellations with timestamps and user IDs.
- Maintain logs of requests for roadmap charging points, ensuring accountability for the features used.

16e Immunity Requirements

- Validate user inputs during searching for nearest available charging station (e.g., location queries) to prevent injection attacks.
- To Implement digital signatures or unique transaction identifiers to ensure users cannot deny making a reservation.
- Use encryption for data transmitted from IoT sensors to prevent interception.
- Ensure that all payment methods comply with security standards, using tokenization where possible to protect card details.

17 Usability and Humanity Requirements for EVCMS

17a Ease of Use Requirements

- **Filters and Sorting**: Include filters for charging station types, availability, and distance, allowing users to customize their search results.
- **Simple Reservation Process**: Offer a straightforward, step-by-step reservation process with clear instructions at each stage.
- **Calendar Integration**: Allow users to view available time slots in a calendar format, making it easier to choose preferred dates and times.

- **Multiple Payment Options**: Support a variety of payment methods (credit/debit cards, digital wallets) for user convenience.
- **Notifications and Reminders**: Send proactive notifications to users about upcoming maintenance schedules, including easy links to schedule services.
- **Real-Time Data Display**: Present real-time data from IoT sensors in an easy-to-understand format, using graphs or visual indicators.
- **Recommendations Based on Preferences**: Provide recommendations for charging stations based on user preferences (e.g., favorite stations).
- **One-Click Favorite Feature**: Implement a simple one-click option to add charging stations to the favorites list.
- **User-Friendly Route Input**: Simplify the process for users to input their starting point and destination to find charging points.
- **Visual Route Overview**: Offer a visual overview of the planned route, highlighting charging points along the way.
- By incorporating these ease of use requirements, the EVCMS can enhance user experience, making it more intuitive and accessible for users of all technical backgrounds.

17b Personalization and Internationalization Requirements

Personalization Requirements

- Provide tailored recommendations for charging stations based on user behavior, favorite locations, and previous reservations.
- Automatically sort favorite stations based on distance, availability, or user ratings.
- Allow users to set preferences for notifications related to charging station availability, maintenance reminders, and promotional offers.
- Offer suggestions for optimizing charging based on historical usage patterns.

Internationalization Requirements

- Allow users to choose their preferred language from a comprehensive list during onboarding and in settings.

- Provide localized content, such as charging station information, pricing, and maintenance tips, relevant to the user's region.
- Automatically adjust times for reservations and notifications to reflect the user's local time zone.
- Provide access to support resources in multiple languages, including FAQs and customer service.
- Implement feedback mechanisms to gather user input on localization and personalization features to continually improve them.

17c Learning Requirements

- Understanding different types of electric vehicles and their charging requirements to provide connector types and other informations during registration.
- Knowledge of the various types of charging stations (Level 1, Level 2, DC fast charging).
- Basics of IoT architecture and how BMS work in monitoring and collecting data to see whether peak usage time is correct or not .
- Knowledge of various payment gateways and transaction processes.
- The EVCMS is easy for users with no technical background to use after watching a 5-minute tutorial.
- 90% of new users should be able to perform a charging session after a short tutorial video.

17d Understandability and Politeness Requirements

Understandability Requirements:

- Simple explanations of how to book a slot, including step-by-step guidance.
- Visual indicators (like calendars) for available time slots.
- Clear icons or labels for connection types and models.
- Visual representation of charging points along the route.

- Avoided using technical jargon in the user interface to make it more understandable for non-experts.

Politeness Requirements:

- Friendly prompts like "Looking for a nearby charging station? We can help!"
- Gentle reminders if no stations are found, e.g., "Sorry, no stations available nearby. Please try a different location."
- Courteous messages during the reservation process, e.g., "Thank you for reserving your slot!"
- Notifications about upcoming reservations, such as "Reminder: Your charging slot starts in 30 minutes."
- Use reassuring language, such as "Choose the connection type that best fits your vehicle's needs."
- Friendly prompts, such as "Choose your preferred payment method, we're here to make it easy!"
- Supportive language, e.g., "Let's find convenient charging stops along your journey!"
- Encouragement to save favorites: "Save your go-to stations for easy access next time!"
- Notifications when a favorite station has availability: "Your favorite station has an opening—check it out!"

17e Accessibility Requirements

- The interface must support screen readers to announce available charging stations clearly.
- Voice command functionality to allow hands-free searching.
- Provide a text-to-speech option to read out station details.
- Options for audio reminders or notifications about maintenance checks.
- Clear, audible confirmation messages should be provided for successful transactions.

- Route input should allow for voice commands to assist users with limited mobility.
- Users should be able to save and retrieve favorite stations easily through voice commands or keyboard shortcuts.

17f User Documentation Requirements

1. User Manual

- To provide a comprehensive guide on how to navigate and utilize all features of the EVCMS Application.
- Provided to End-users (EV owners).
- Updated regularly with each app version release; user feedback can be incorporated to improve clarity.

2. Installation Manual

- To guide users on how to download, install, and set up the application on various devices.
- Updated for compatibility with new operating systems and devices.

3. Technical Specifications Document

- To detail the technical requirements, system architecture, and integration points for the application.
- Provided to Developers, system integrators, and IT teams.

4. Service Manual

- To provide troubleshooting steps and guidelines for common issues that users may encounter.
- Regular updates based on user feedback and common issues reported.

5. FAQs and Troubleshooting Guide

- To answer common questions and provide quick solutions to typical problems.
- Continuously updated based on user inquiries and issues reported.

6. User Feedback and Suggestions Document

- To collect and track user suggestions for future improvements and features.

17g Training Requirements

- **User Training for EV Owners:**
  - How to navigate the app (searching for stations, making reservations)
  - Understanding different charging station types and their specifications
  - Payment options and processes
  - Adding and managing favorite charging stations
- **Technical Training for Charging Station Operators:**
  - Managing and updating station information (availability, pricing, etc.)
  - Understanding the integration of IoT sensors and data analysis
  - Monitoring charging patterns and adjusting schedules
- **Who Will Provide the Training:**
  - Team members with expertise in the application and electric vehicle technologies
  - Experienced staff who understand user needs and can address common challenges
- Comprehensive guides and FAQs to assist users post-training.

18 Operational and Environmental Requirements

18a Expected Physical Environment

- The app may require offline functionality for areas with weak signals.
- Sensors capable of operating in various weather conditions (temperature range -40°C to +85°C).
- Large, readable fonts and clear icons for use in diverse lighting conditions (daylight, low light).
- To accommodate high demand and optimize user experience during peak hours.
- Charging stations equipped with IoT technology, which may be exposed to various weather conditions.
- Users should receive route recommendations even when connectivity drops.

18b Requirements for Interfacing with Adjacent Systems

- **Charging Station Data Interface**
    - Charging station location, availability status, connection types (e.g., CCS), pricing.
    - **Physical Material Content**: JSON or XML data formats.
    - **Medium**: RESTful APIs over HTTPS.
    - **Frequency**: Real-time updates with a polling interval of 30 seconds for station availability; daily updates for pricing information.
    - **Volume**: Up to 1,000 requests per minute during peak usage periods.

- **Payment Gateway Integration**
    - User payment information, transaction history, charging session costs, and receipts.
    - **Medium**: HTTPS-based API calls.
    - **Frequency**: Transaction requests processed in real-time; summary data synced every 24 hours.

- **Volume**: Expected 500 transactions per minute during peak hours.

- **IoT Sensor Data Interface**
  - **Data Content**: Charging patterns, energy consumption metrics, and predictive maintenance alerts.
  - **Medium**: MQTT protocol for lightweight messaging.
  - **Frequency**: Real-time data transmission every 5 seconds during charging sessions.
  - **Volume**: Estimated 10,000 data points per session.

- **Route Planning and Mapping Services**
  - **Data Content**: Route data, on-route charging station locations, user preferences, and traffic conditions.
  - **Medium**: Integration with mapping APIs (e.g., Google Maps API).
  - **Frequency**: Route updates triggered by user interaction or when charging stations are added/removed.
  - **Volume**: Up to 1,000 route requests per minute.

18c Productization Requirements

- Users should expect to install the application in less than 10 minutes, with a straightforward installation process requires minimal technical knowledge.
- The reservation process shall be completed in no more than three steps: select station, choose time, confirm booking.
- Documentation shall be provided on setting up and managing IoT devices.
- Users should expect to set up IoT devices in less than 15 minutes, requiring no advanced technical skills.
- Allow users to download maps for offline use during long trips.
- The favorites feature shall be easily accessible from the main menu and allow users to manage their favorites with minimal clicks.

- Allow users to label or categorize their favorites for easier management (e.g., "Home," "Work," "Road Trip").

18d Release Requirements

- Updates for the charging station locator will be released quarterly, including data updates and performance improvements.
- Major feature updates will occur biannually, with minor updates and bug fixes rolled out monthly.
- All updates will maintain seamless payment functionality.
- New algorithms shall improve prediction accuracy without disrupting existing maintenance notifications.
- Keeping IoT integration up-to-date is essential for accurate monitoring.
- Enhancements to routing features will occur quarterly, with data updates for charging points provided monthly.

19 Cultural and Political Requirements

19a Cultural Requirements

- Adapted the locator feature to accommodate local preferences for charging station types (e.g., fast chargers vs. standard chargers).
- Provides multilingual support to cater to diverse user demographics.
- Incorporated cultural considerations regarding scheduling behaviors (e.g., peak usage times during festivals or holidays).
- Integrated popular local payment methods, considering cultural preferences (e.g., mobile payments, cash options).
- Provides educational resources on the payment process, especially in regions where digital payments are less common.
- Tailored routing algorithms to account for varying infrastructure quality in different regions.

- The importance of road trip culture varies; in some cultures, long drives are a common family activity, while in others, public transport is preferred. so, Incorporate features that suggest family-friendly charging stops
- Allow users to customize their favorites based on cultural considerations, such as preferred charging station brands or locations.

19b Political Requirements

- Collaborated with local governments to access public charging infrastructure data and promote the use of public chargers of Charging Stations.
- Support for government initiatives aimed at promoting electric vehicle (EV) use, such as incentives for using charging stations during off-peak hours.
- Compliance with local financial regulations and support for national payment systems.
- Address any local tax implications related to charging fees and ensure transparent pricing to comply with regulations.
- Adheres to industry standards and government regulations related to IoT device integration and security.
- Ensure that the app features charging points that are consistent with government-approved routes and plans for EV infrastructure.
- Coordination with governmental travel and transportation policies to ensure adequate infrastructure support for long-distance EV travel.

20 Legal Requirements

20a Compliance Requirements

- Ensure that all listed charging stations comply with municipal regulations, including permits and safety standards.
- Clearly communicate reservation policies, cancellation terms, and potential fees to users, following local consumer protection regulations.

- Clearly display all relevant information, ensuring prices are transparent and comply with local pricing laws and guidelines.
- Ensure that all IoT devices used in the application meet industry standards for security and data handling, following relevant guidelines.
- Implement secure payment processing systems and ensure that user payment data is handled in accordance with applicable financial regulations.
- Ensure that all routing and charging point information aligns with officially approved transportation plans and EV infrastructure initiatives.
- Provide accurate and timely information about on-route charging stations, ensuring that it adheres to local regulations regarding navigation and road use.
- Ensure that users can easily manage their favorite stations and that any associated data handling complies with relevant privacy laws.

20b Standards Requirements

- The product shall comply with IEEE 802.11 standards for wireless communication to ensure reliable geolocation services.
- Compliance with these standards will enhance the accuracy and reliability of location-based services, avoiding delays in search functionality.
- The product shall be developed according to Agile development standards to facilitate iterative enhancements.
- The product shall comply with the International Electrotechnical Commission (IEC) standards for electric vehicle charging systems (IEC 61851) that the charging station feature aligns with the required standards.
- The product shall adhere to the Internet of Things (IoT) standards set forth by the IEEE 802.15.4 confirming that the IoT integrations meet the LR-WPAN standards.
- The product shall comply with PCI DSS (Payment Card Industry Data Security Standards) for secure payment processing.[6]
- The product shall comply with the National Institute of Standards and Technology (NIST) standards for location-based services.

III Diagrams and Design

21 UML Diagrams

21a Class Diagram

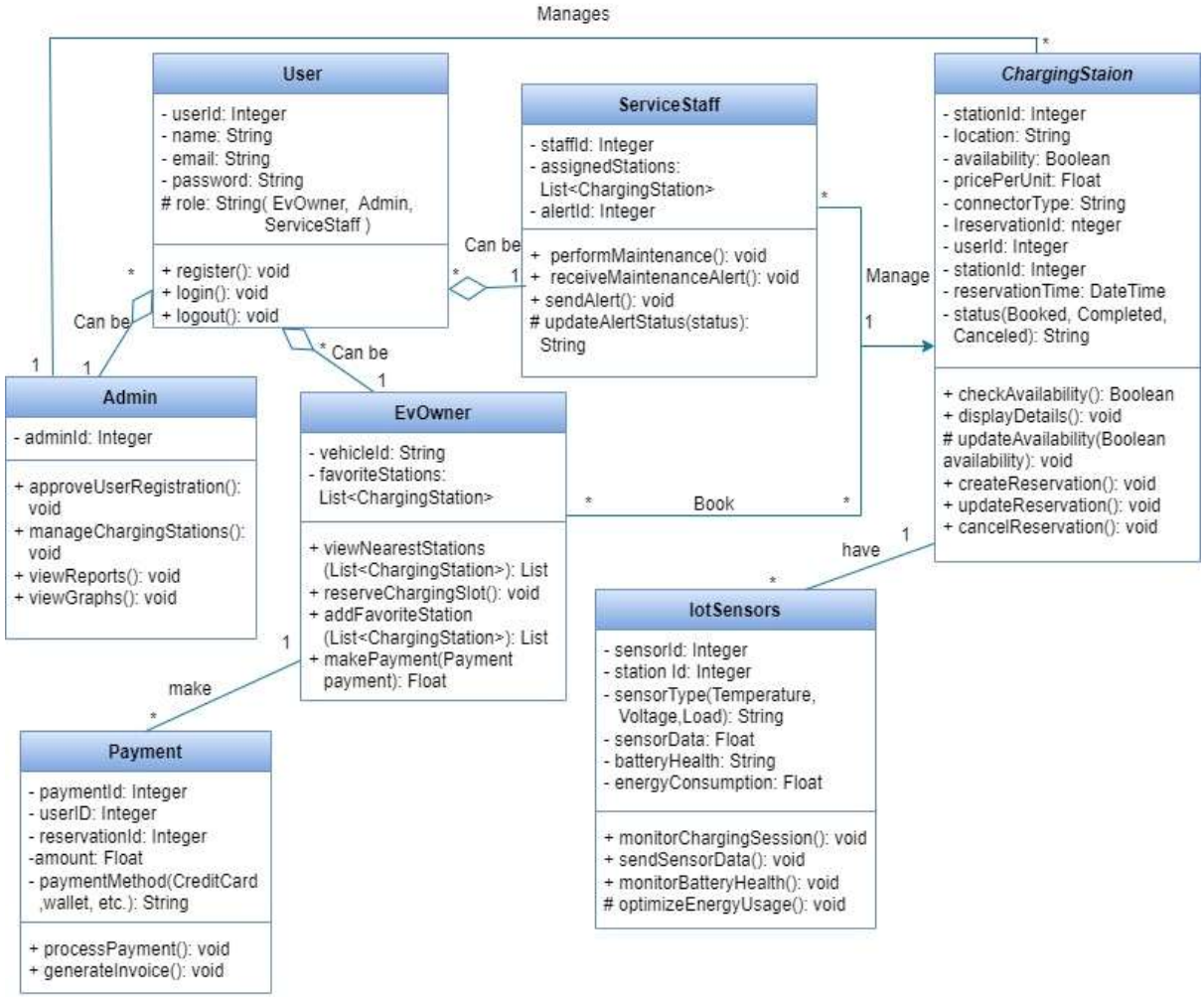


Figure 7 – Class Diagram:

21b Activity Diagram

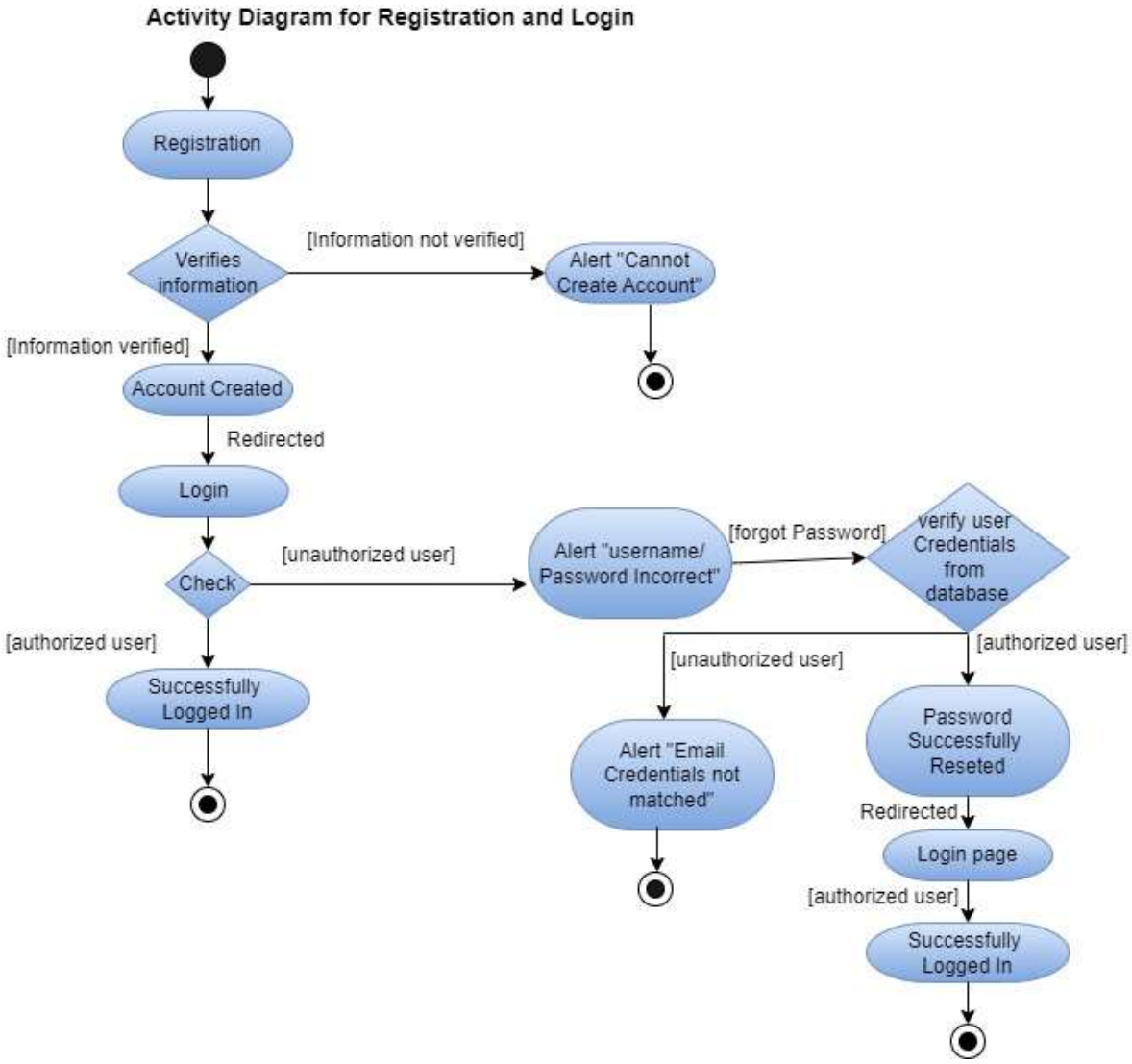


Figure 8 – Activity Diagram for Registration and Login:

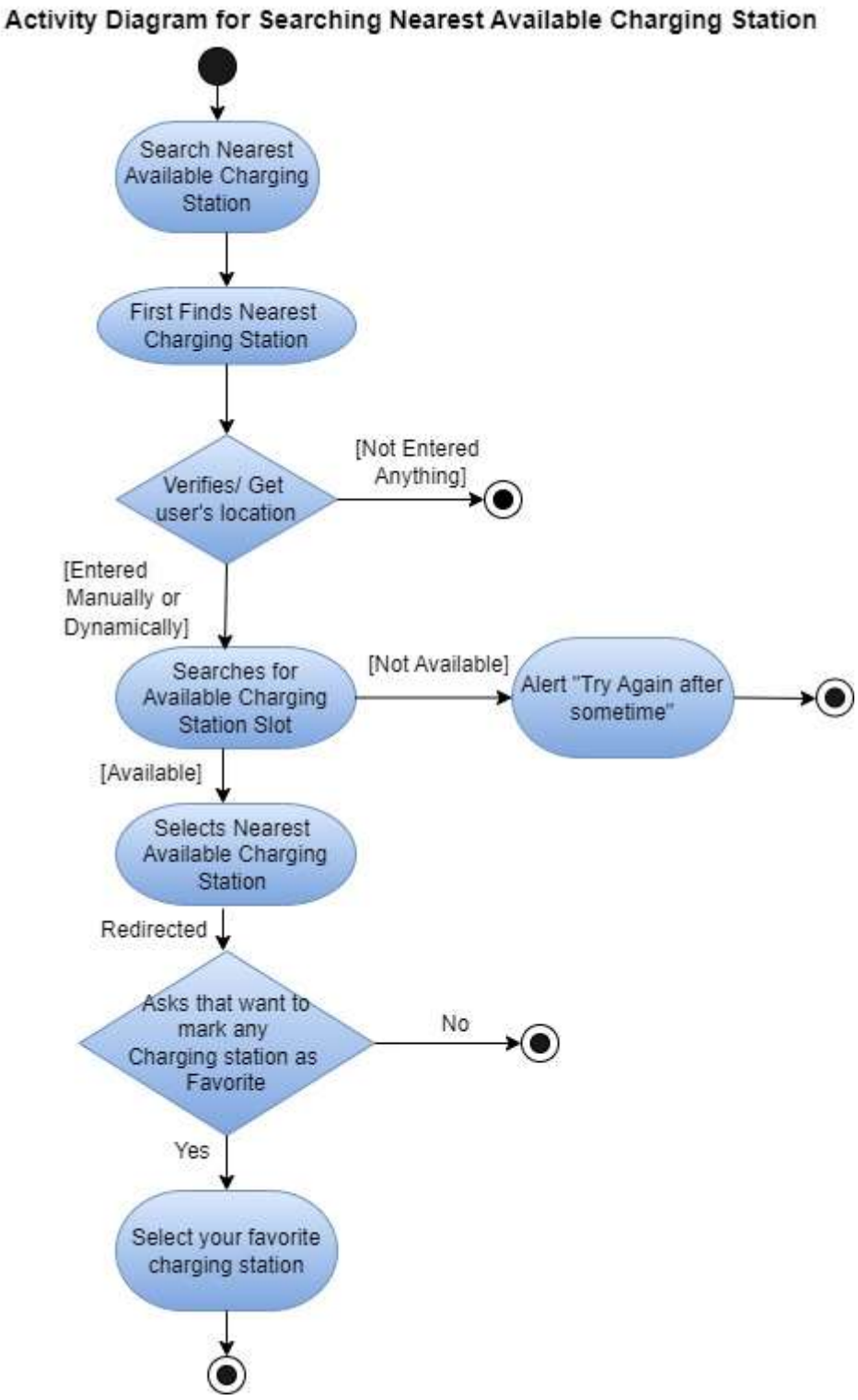


Figure 9 – Activity Diagram for Searching Nearest Available Charging Station:

Activity Diagram for Dynamic Scheduling

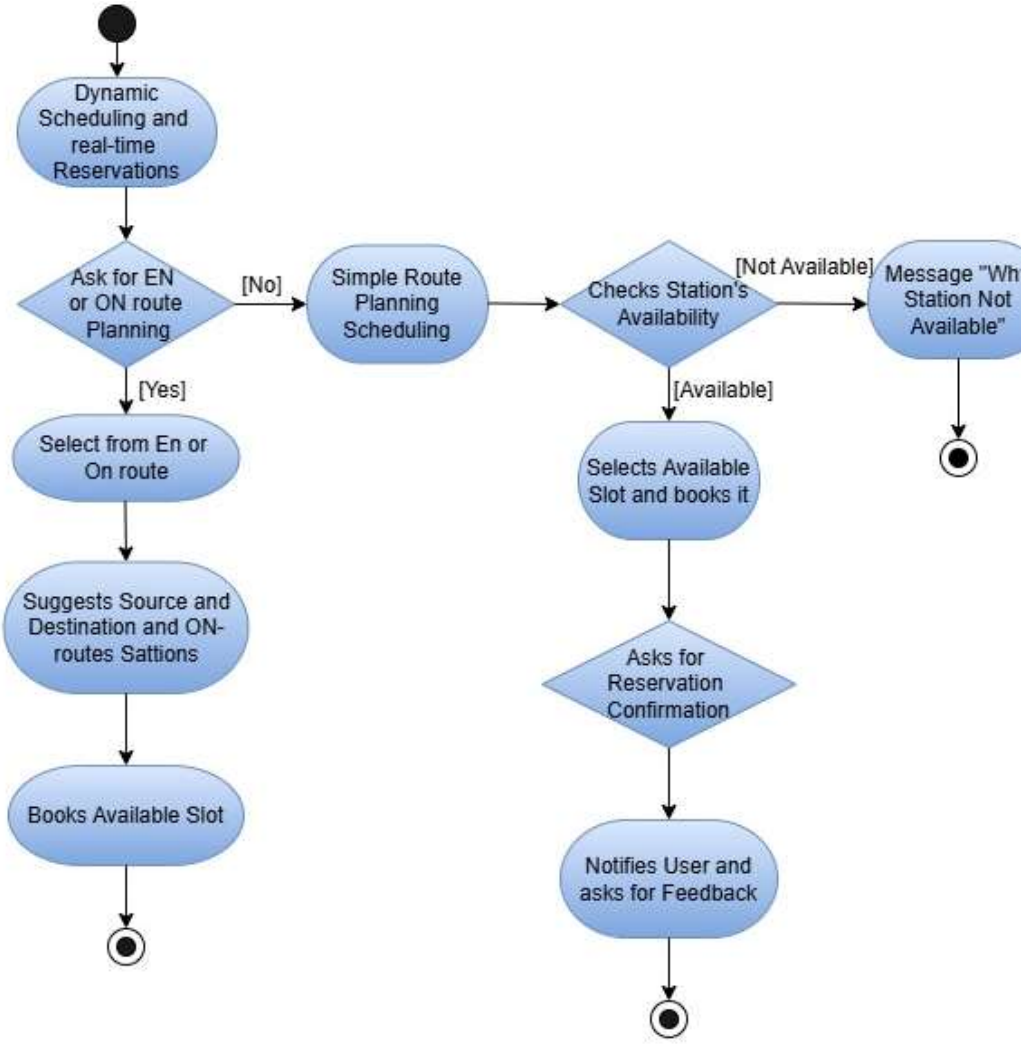


Figure 10 – Activity Diagram for Dynamic Scheduling:

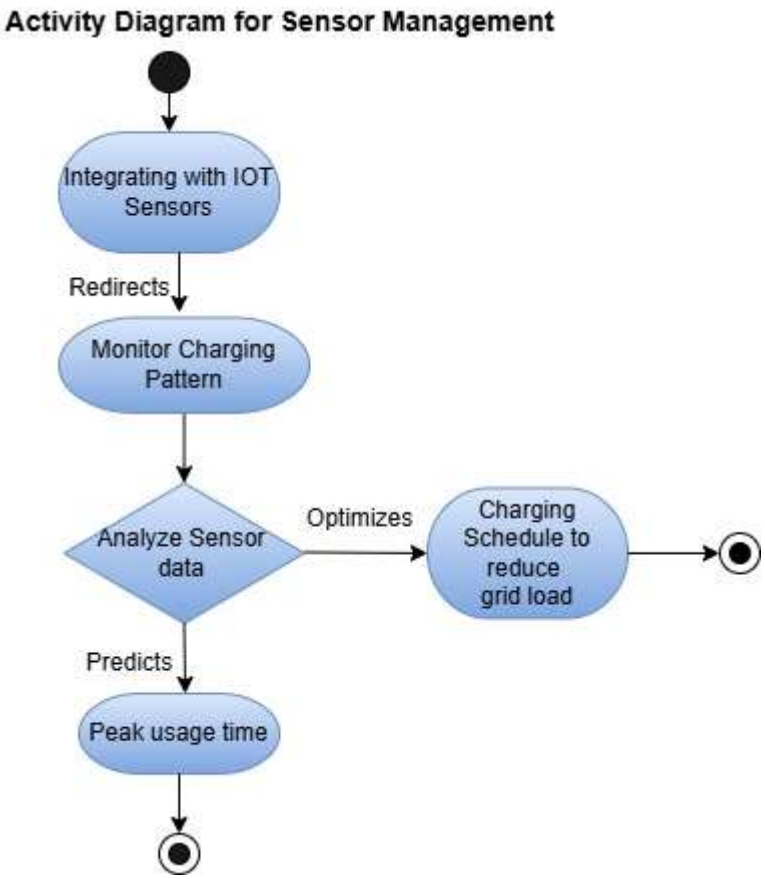


Figure 11 – Activity Diagram for Sensor Management:

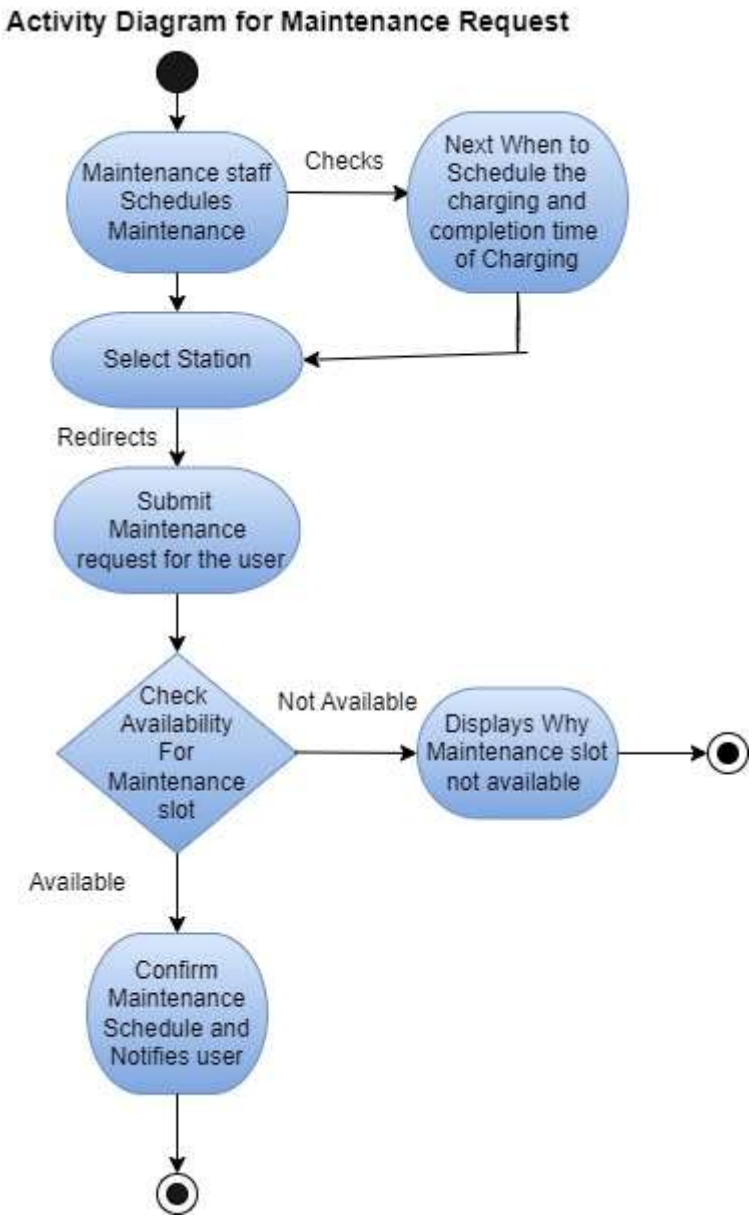


Figure 12 – Activity Diagram for Maintenance Request:

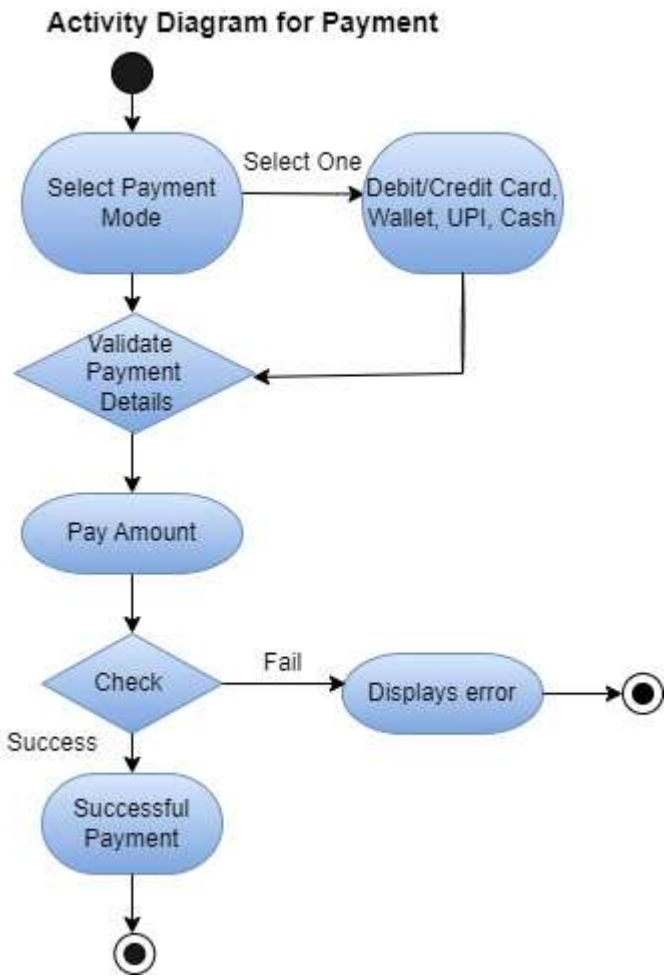


Figure 13 – Activity Diagram for Payment:

21c Swimlane Diagram

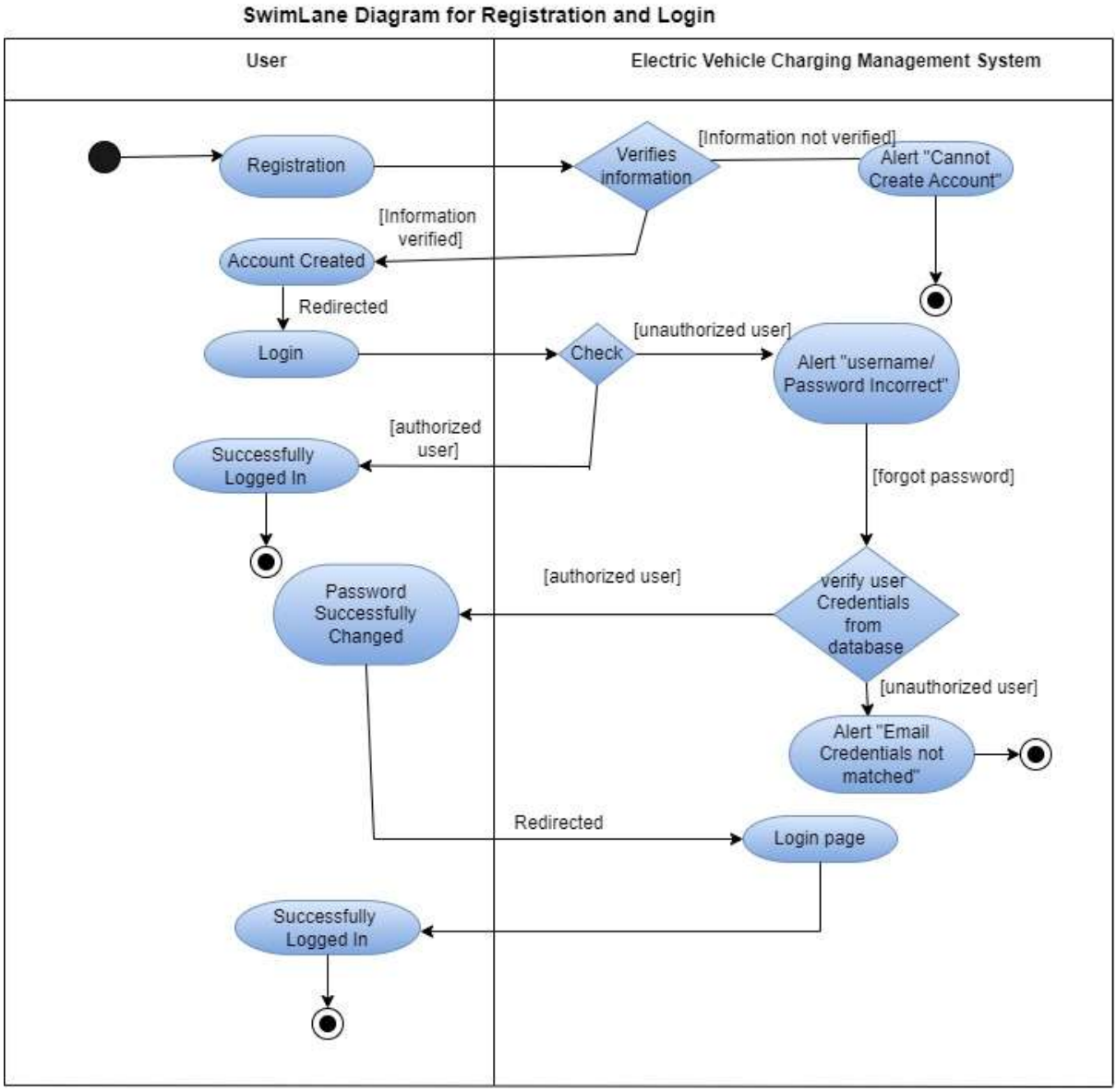


Figure 14 – Swimlane Diagram for Registration and Login:

Swimlane Diagram for Searching Nearest Available Charging Station

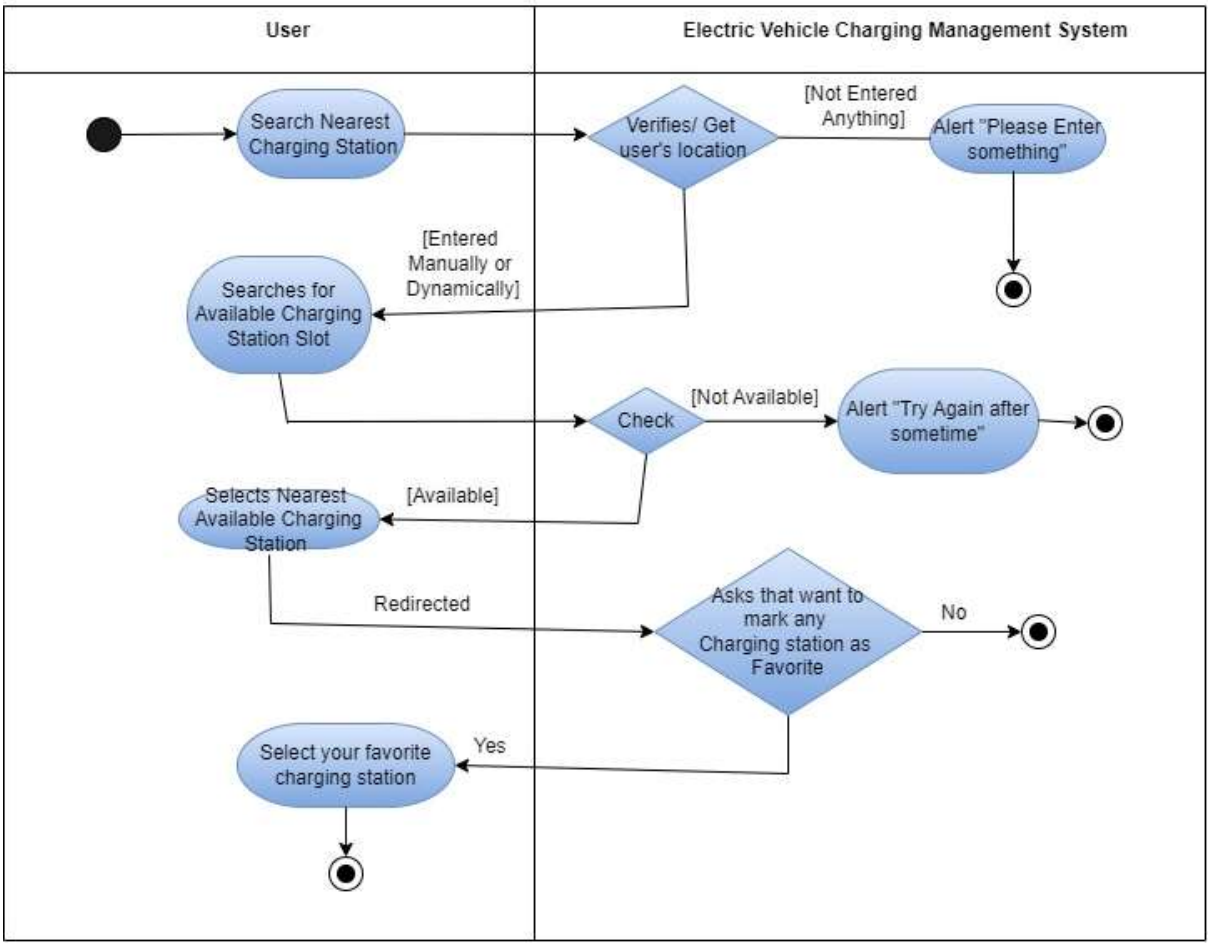


Figure 15 – Swimlane Diagram for Searching Nearest Available Charging Station:

Swimlane Diagram for Dynamic Scheduling

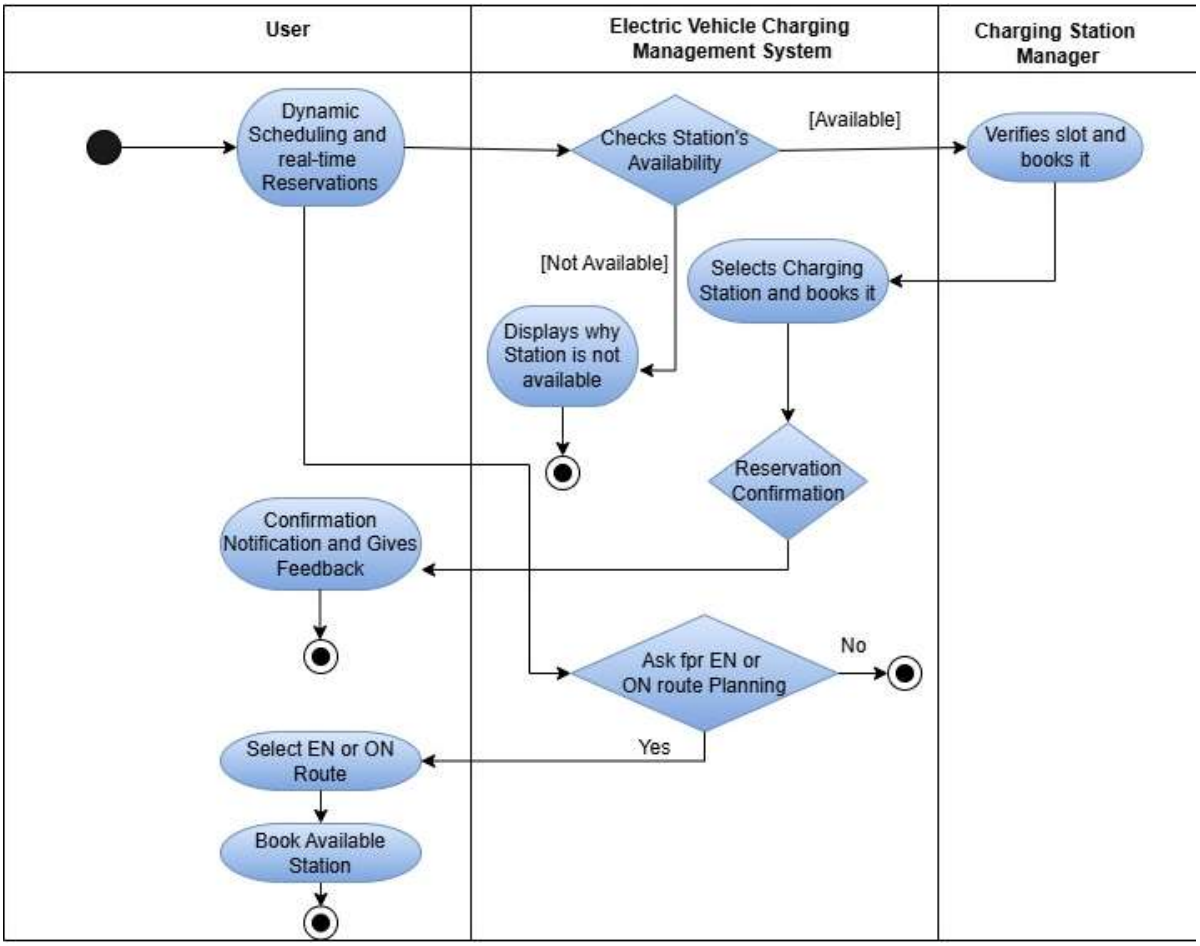


Figure 16 – Swimlane Diagram for Dynamic Scheduling:

Swimlane Diagram for Sensor Management

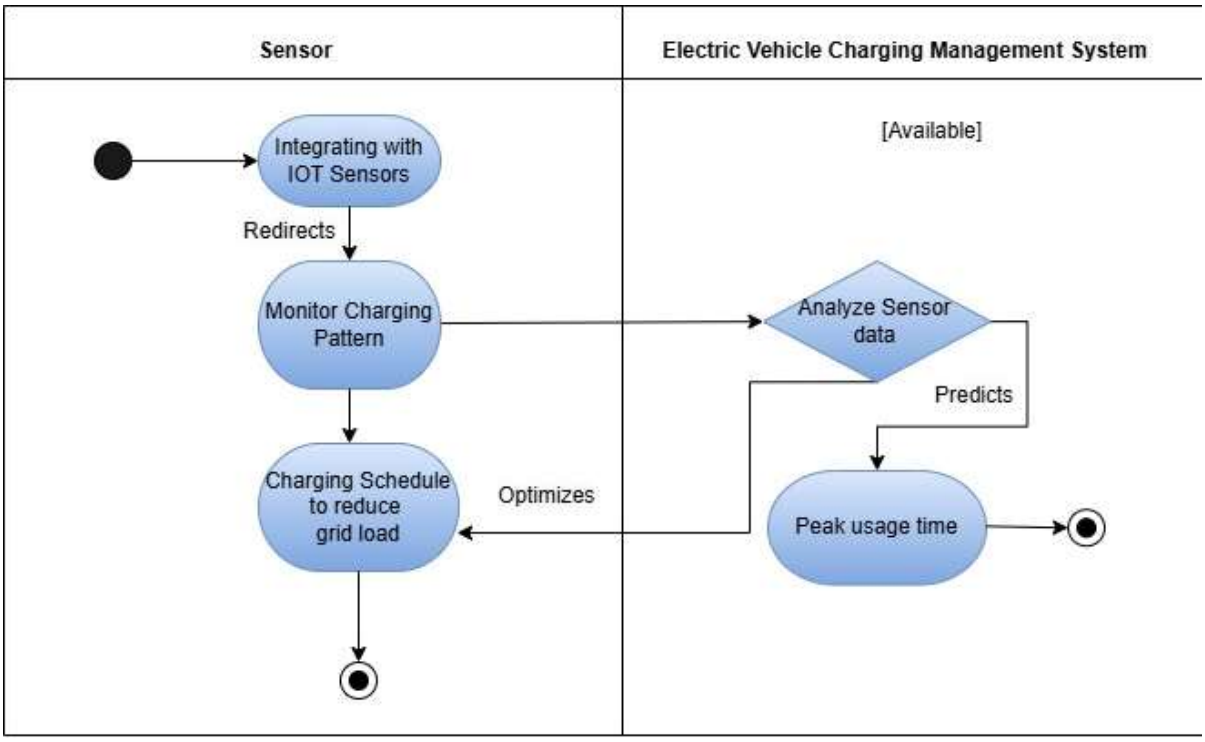


Figure 17 – Swimlane Diagram for Sensor Management:

Swimlane Diagram for Maintenance Request

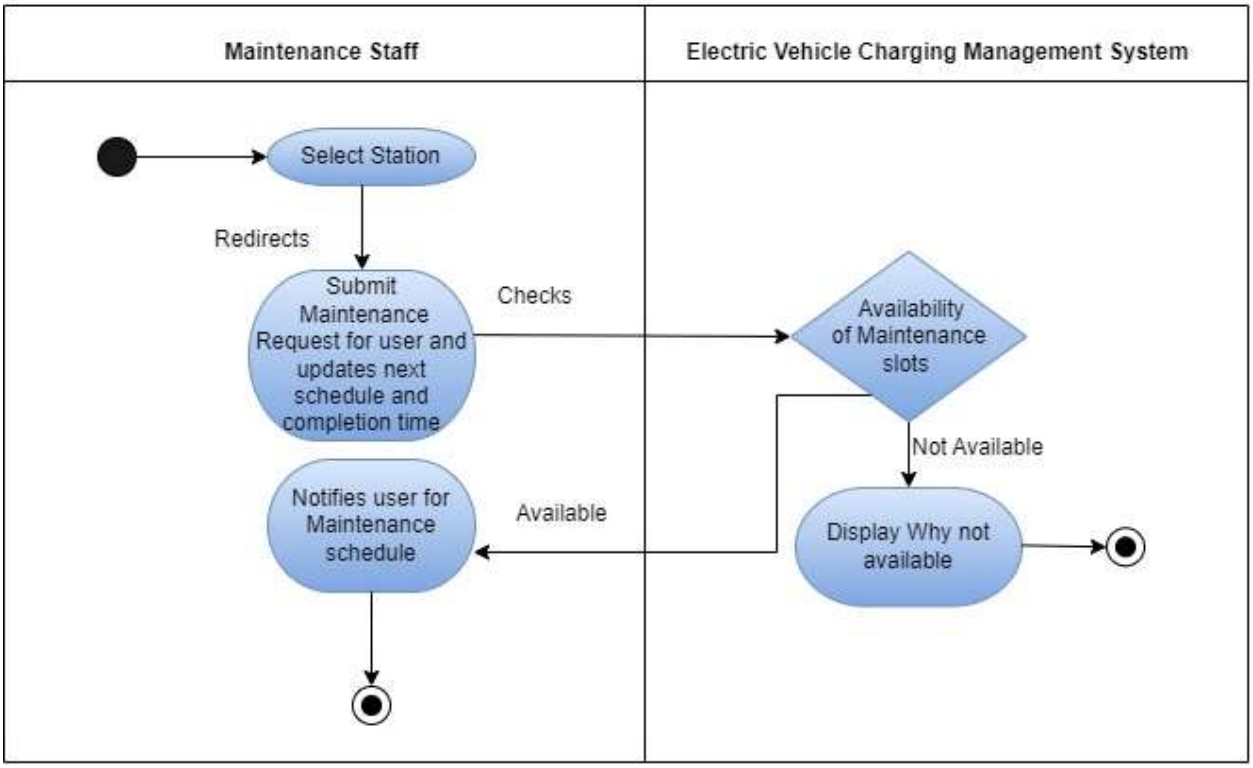


Figure 18 – Swimlane Diagram for Maintenance Request:

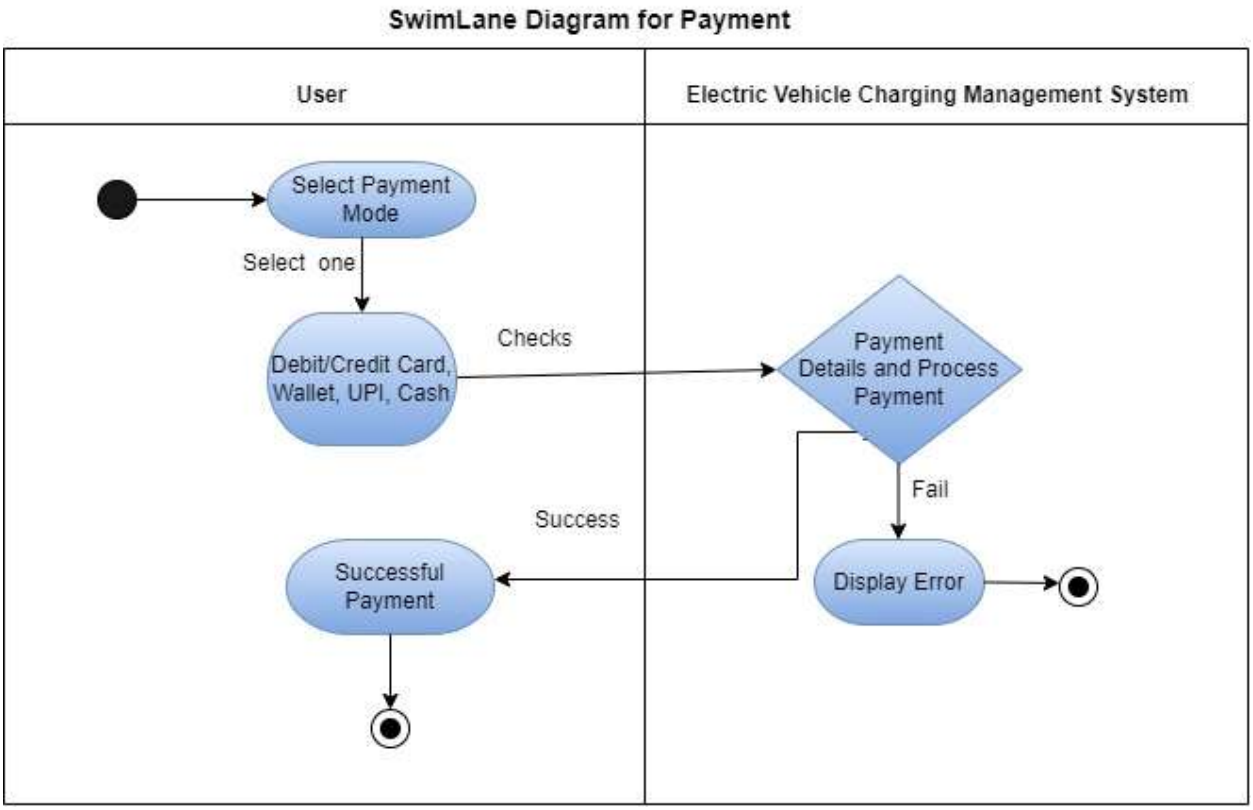


Figure 19 – Swimlane Diagram for Payment:

21d ER Diagram

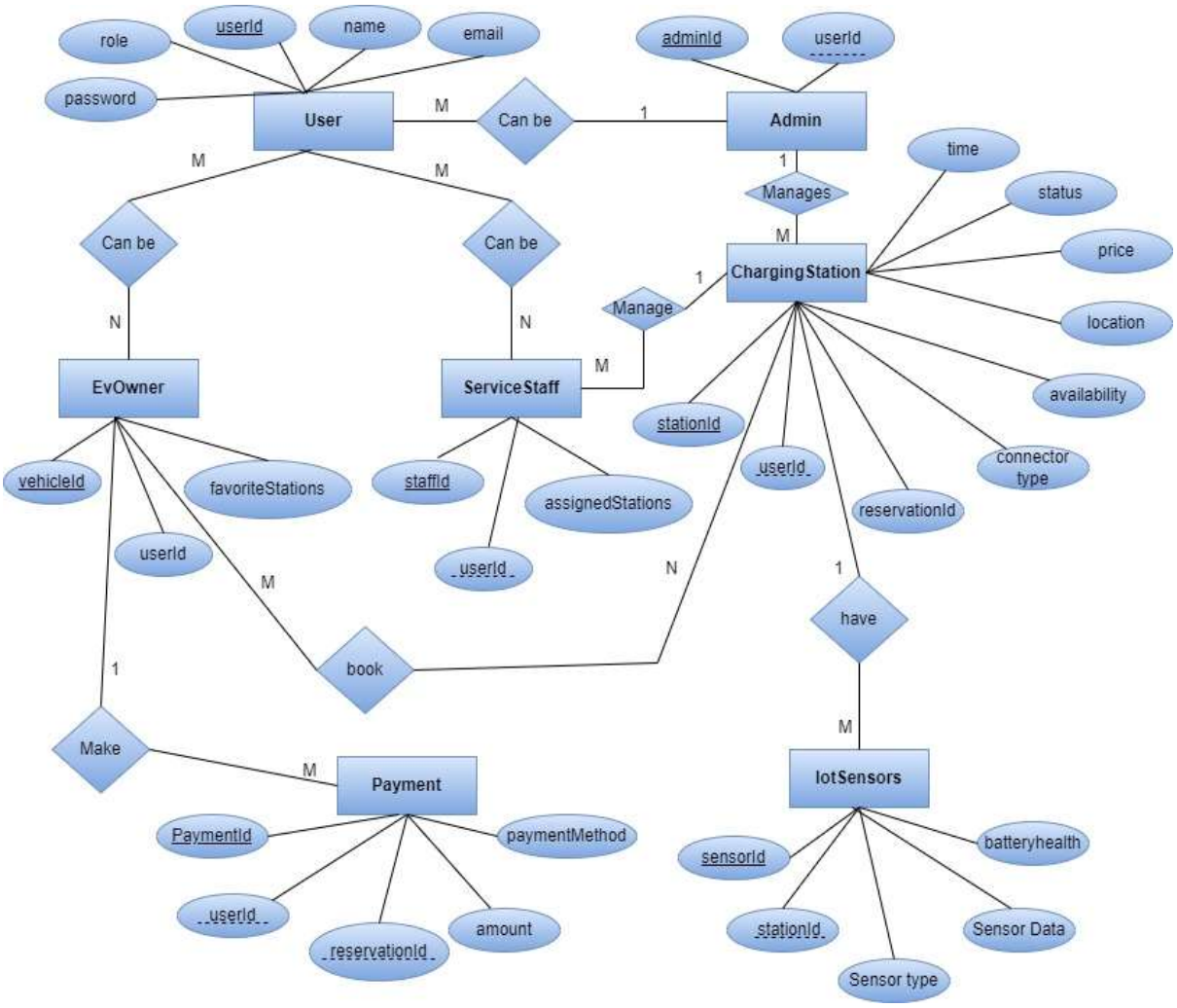


Figure 20 – ER Diagram:

21e DFD Diagram

EVCMS (Electric Vehicle Charging Management System) - Level 0 DFD:

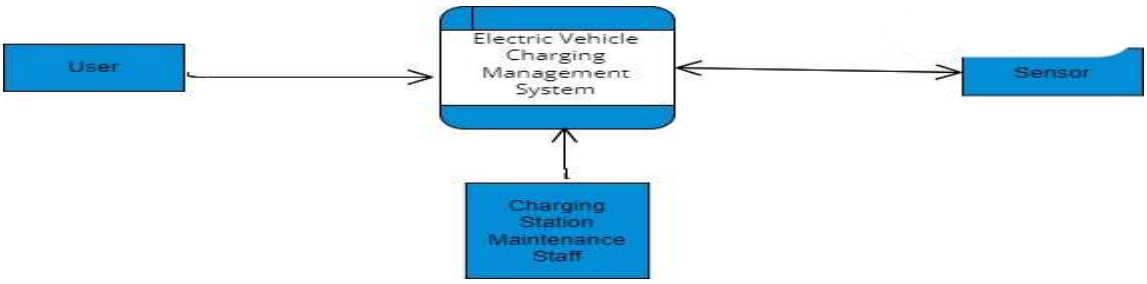


Figure 21 – DFD Diagram- Level 0:

EVCMS - Level 1 DFD:

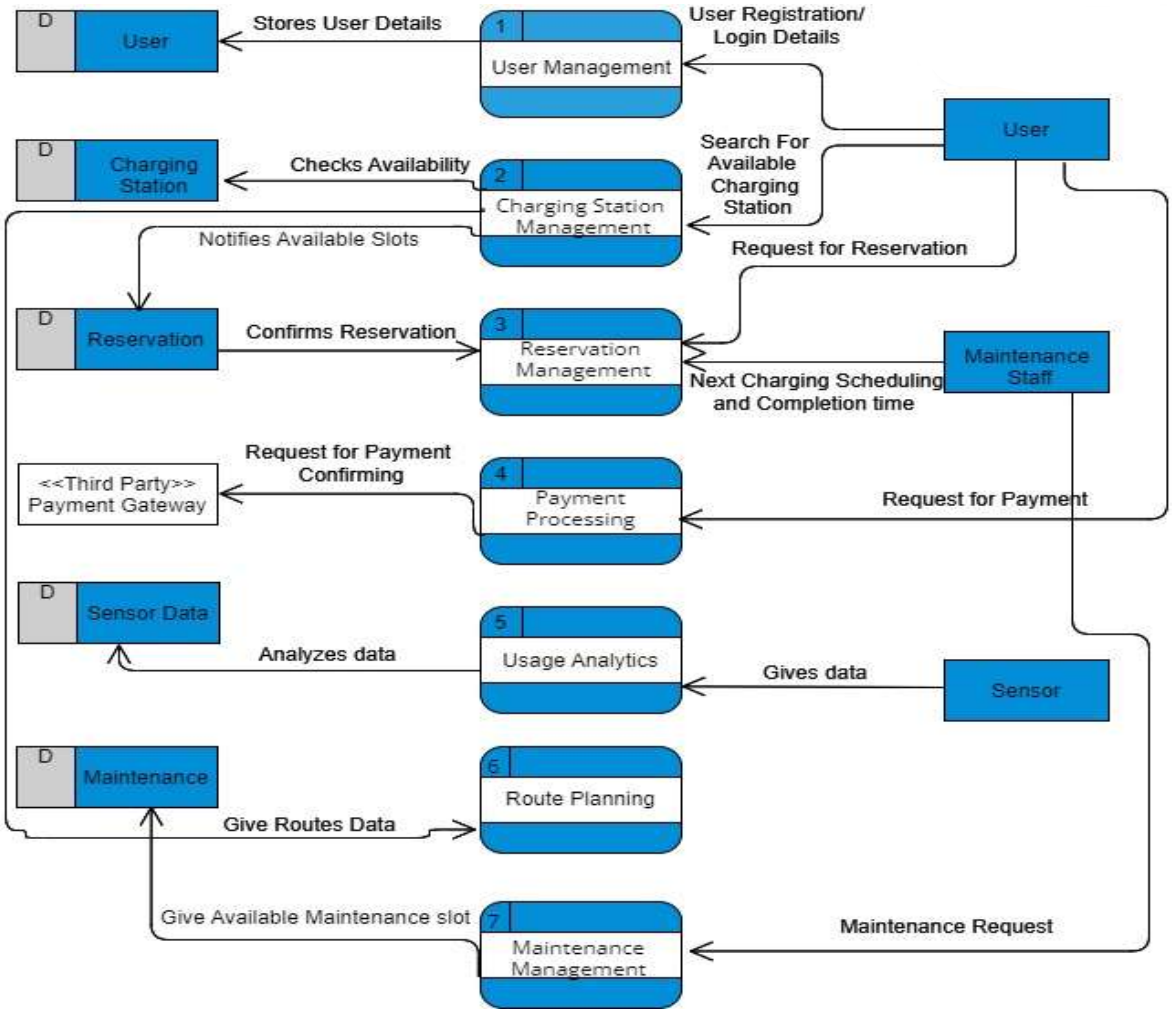


Figure 22 – DFD Diagram- Level 1:

EVCMS - Level 2 DFD:

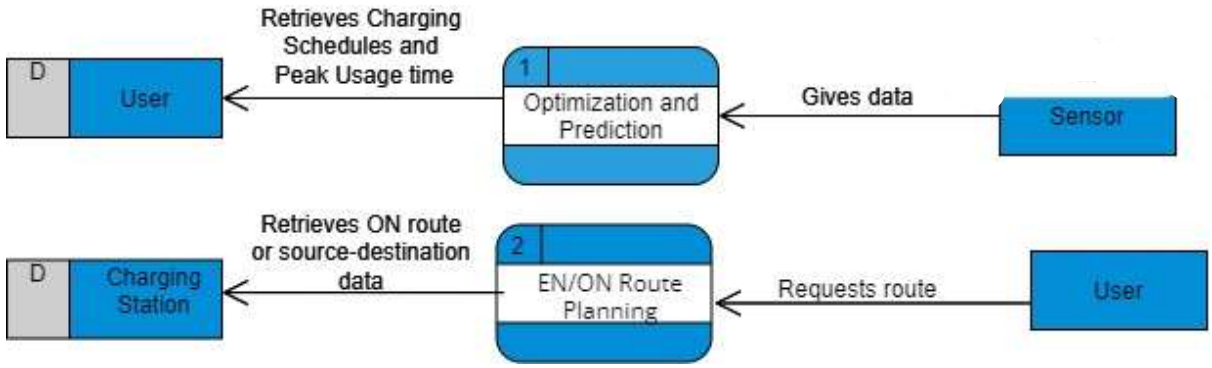


Figure 23 – DFD Diagram- Level 2:

22 System Design

The system design for the "Electric Vehicle Charging Management System" application is centered around creating a user-friendly, engaging, and efficient platform that allows users to play the game seamlessly.

The design aims to balance simplicity and functionality while ensuring a smooth user experience across various devices.

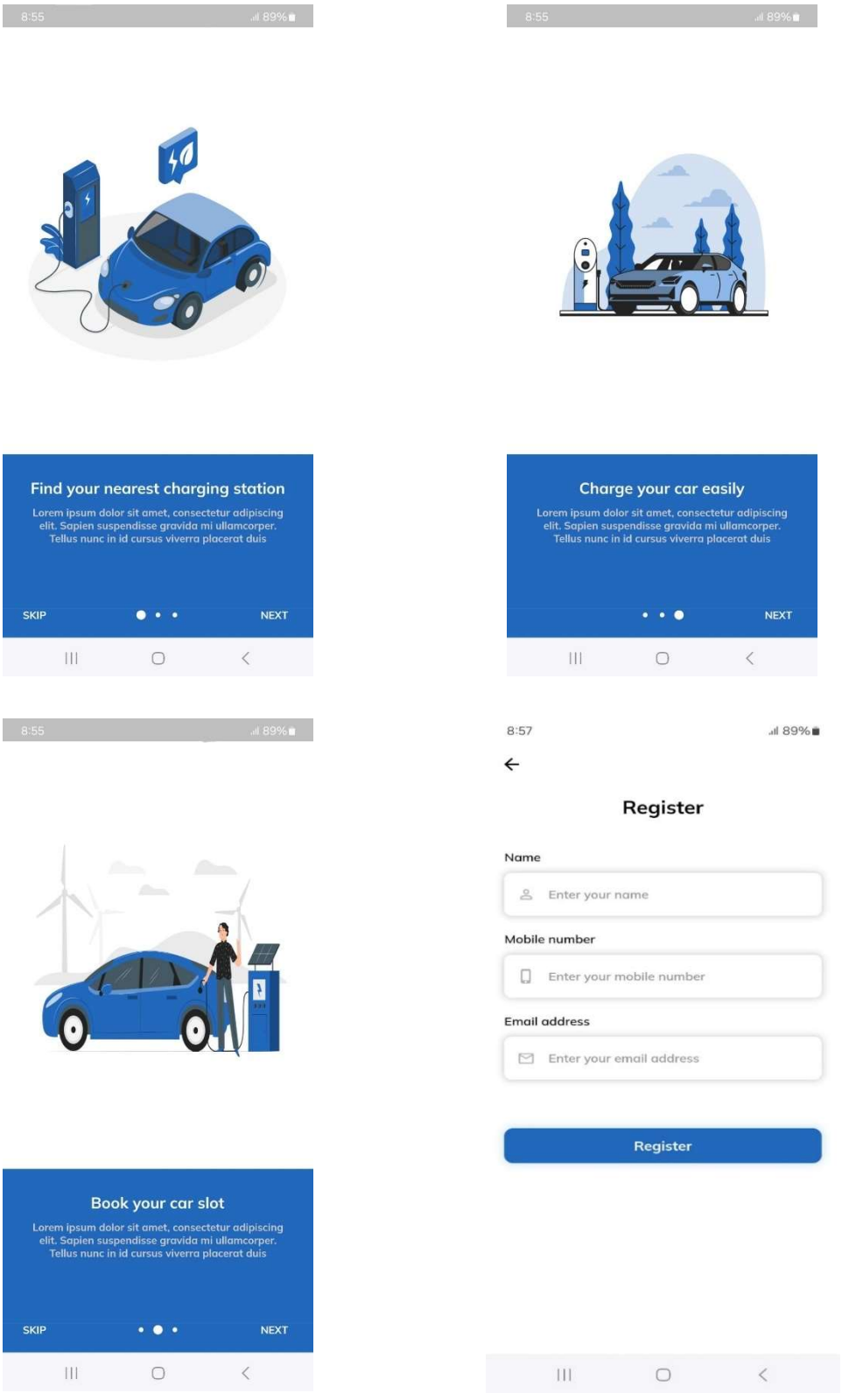


Figure 24 – Home Screen and Registration Screen:

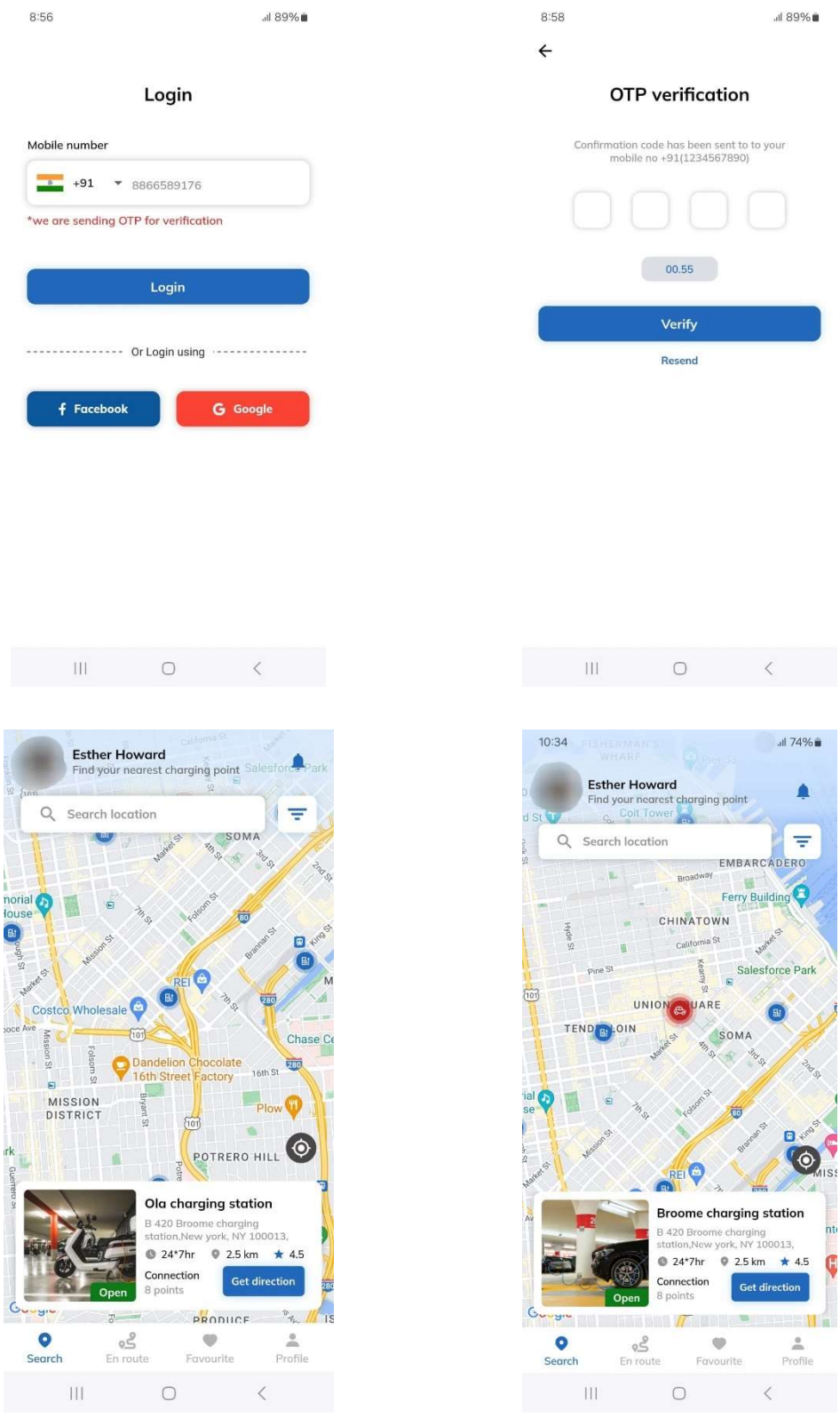


Figure 25 – Login, OTP Verification, Search Charging Station Screen:

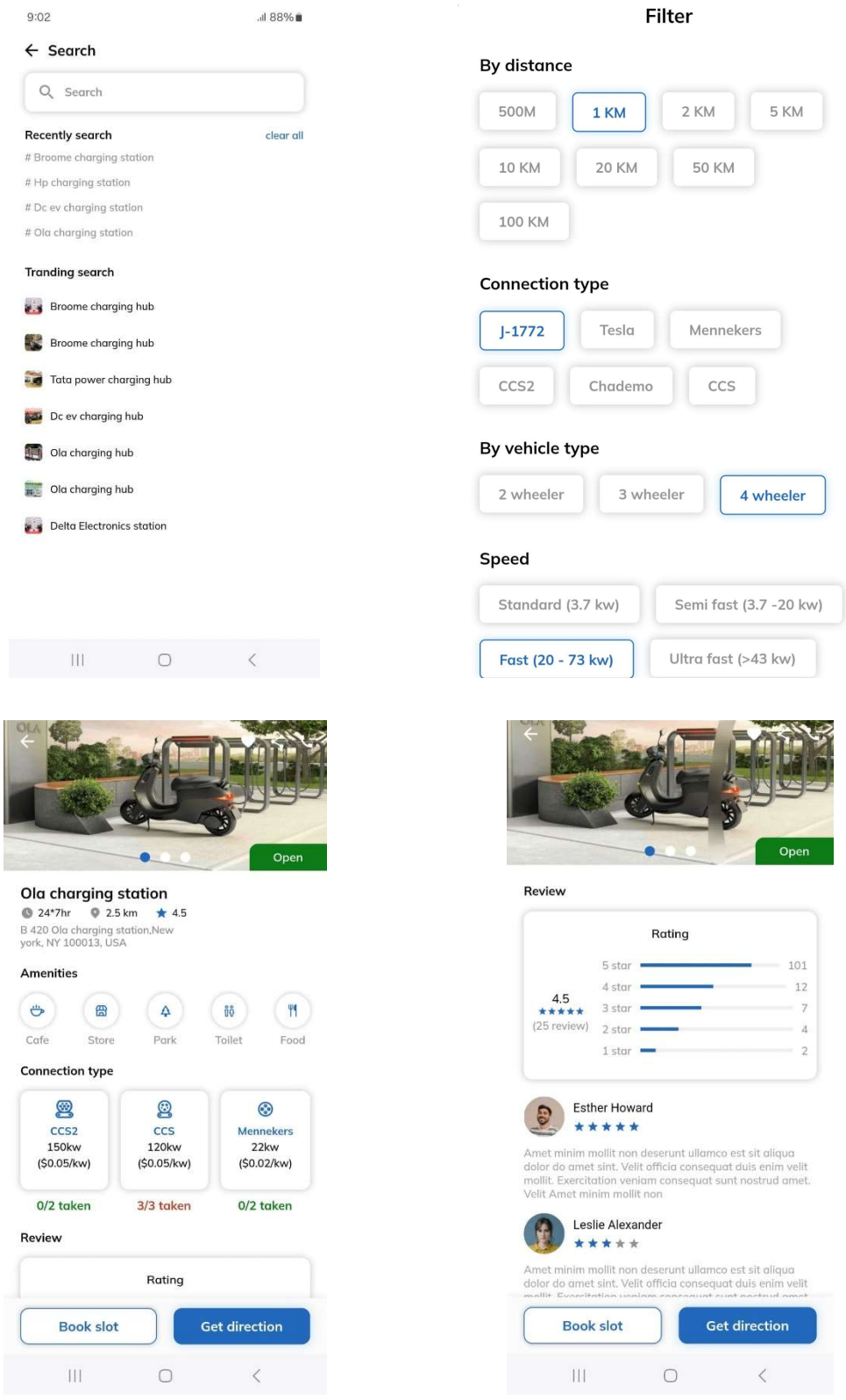


Figure 26 – Recently Search, Filter and All info about Searched Charging Station Screen:

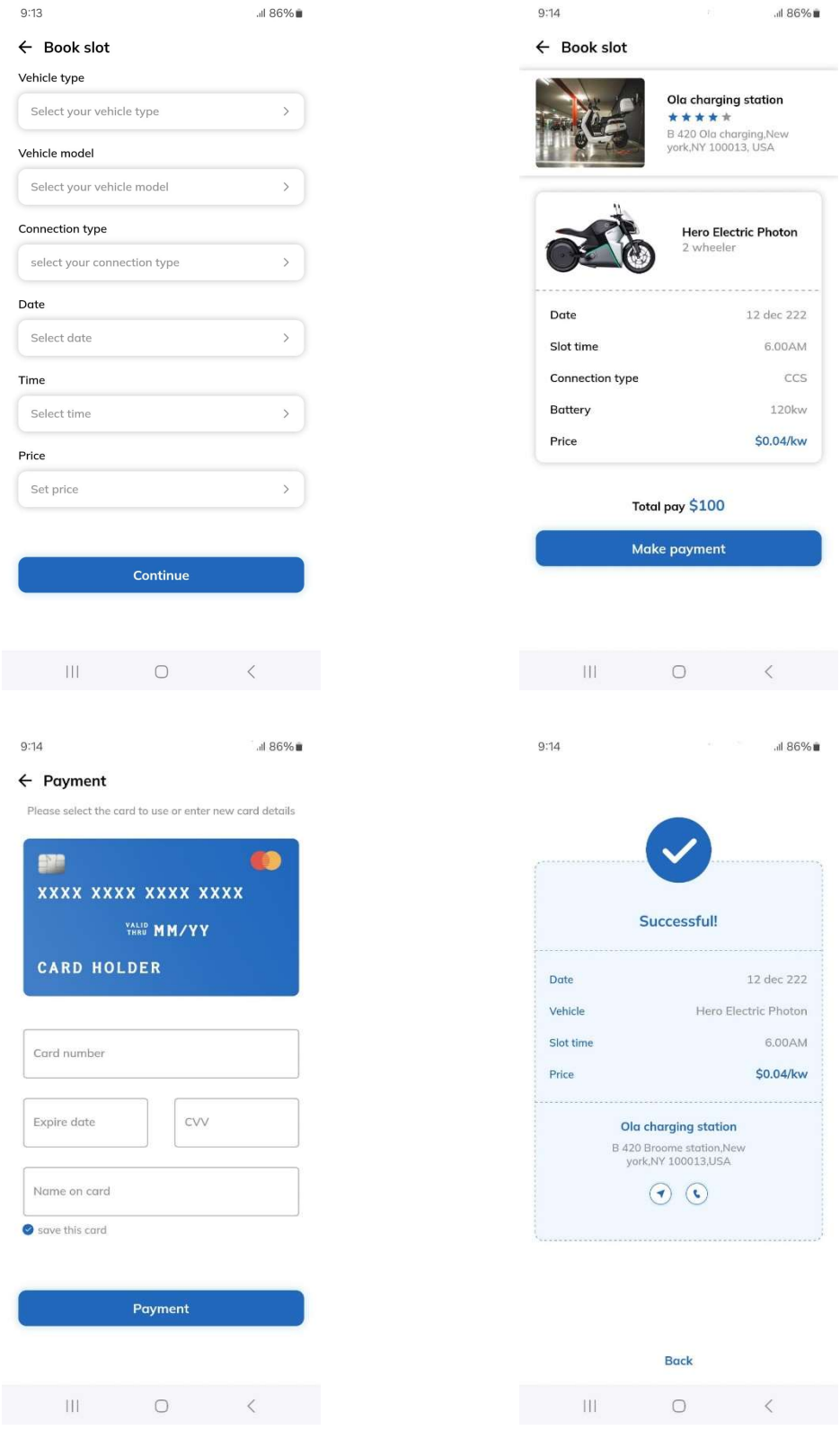


Figure 27 – Book Slot Form, Make Payment , details of and Successful Payment Screen:

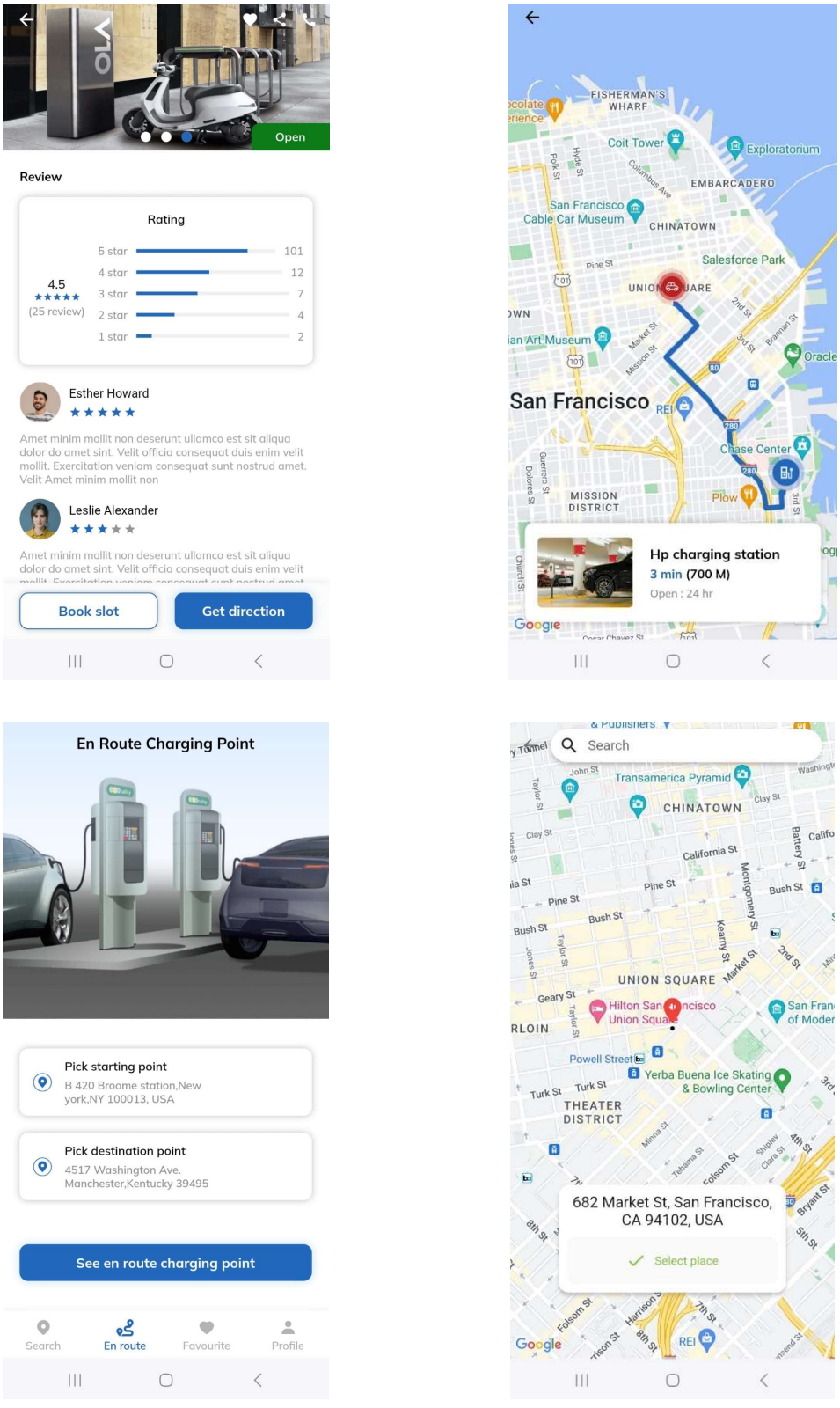


Figure 28 – Ratings, Get Direction, EN-Route, Change Start or Destination Point Screen:

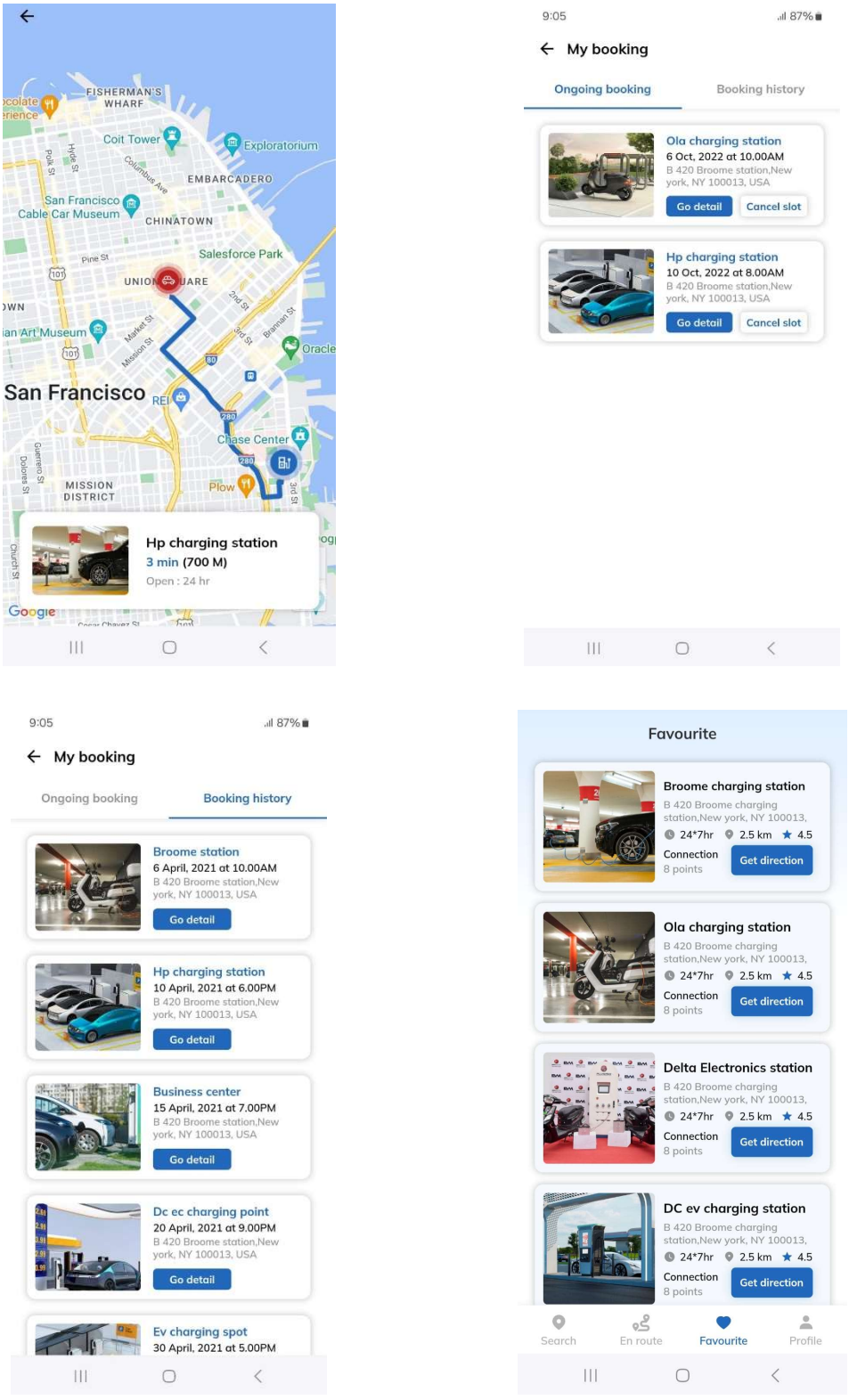


Figure 29 – Charging Station Status, Ongoing and History of Boking , Favorite Charging Station Screen:

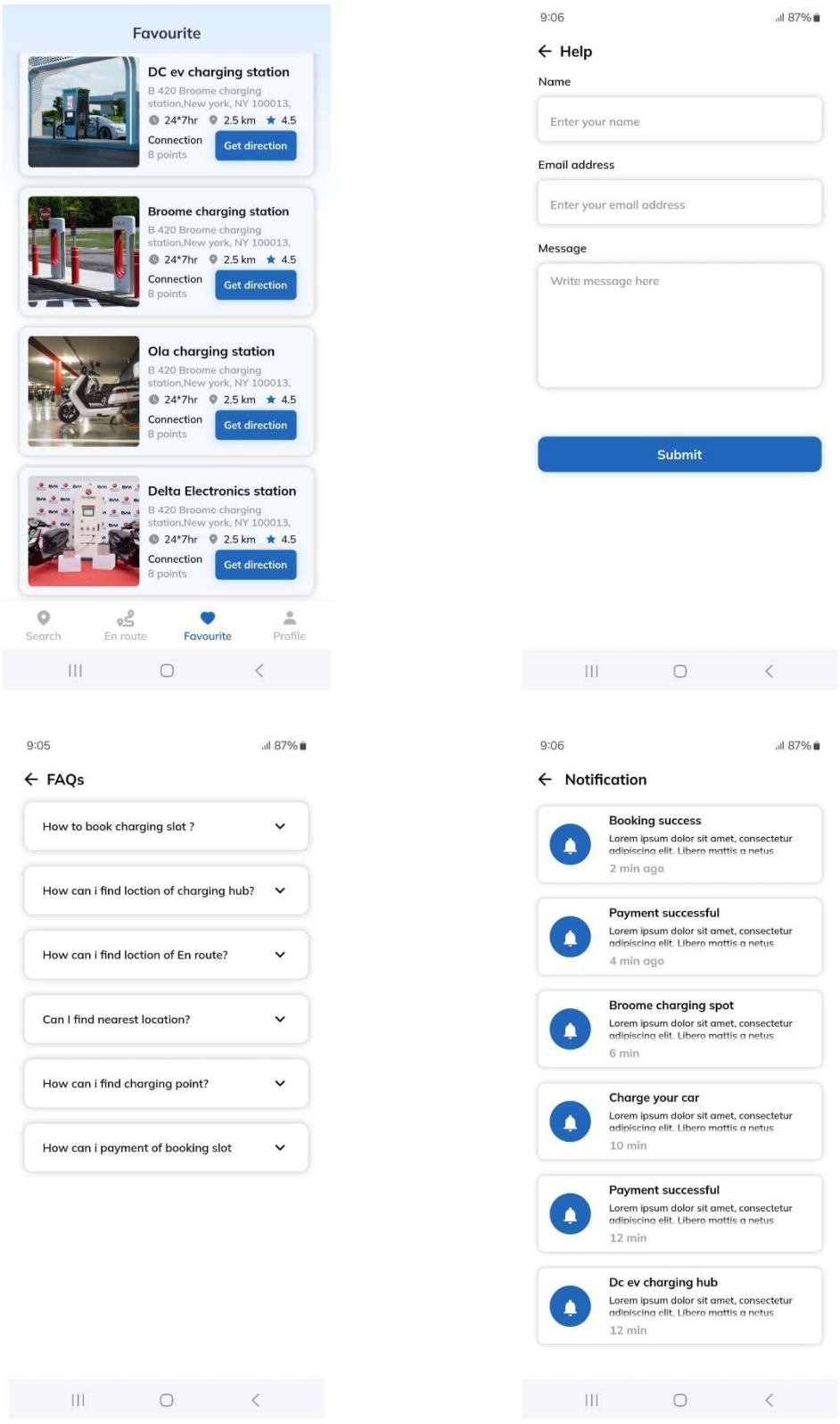


Figure 30 – Favorite Charging Station, Help, FAQS and Notification Screen:

22a Design goals [7]

A clear overview helps stakeholders, including developers, testers, and project managers, to understand the overall architecture and how components function together. It establishes a common understanding, ensuring that all team members align their work with the system's objectives.

- **Ease of Use**: The application should offer an intuitive interface that makes it easy for users to search for, reserve, and navigate to charging stations. This includes providing clear and simple navigation for EV owners, including the ability to find charging stations by location, availability, and type of charger.
- **Real-Time Updates**: Ensure that users receive timely notifications for station availability, charging slot reservations, maintenance schedules, and estimated completion times.
- **Seamless Booking and Reservation**: Allow users to book and schedule charging slots dynamically and in real-time, optimizing their travel and charging plans.
- **Multi-Language and Region Support**: The application should support multiple languages and currencies to serve a global audience and offer region-specific features (e.g., localization of charging stations, payment methods).
- **Multiple Payment Options**: Provide a variety of payment options, including credit/debit cards, e-wallets, and other region-specific payment systems, to ensure convenience and flexibility for users.
- **Energy Efficiency**: The system should prioritize energy-saving features such as scheduling charging during off-peak hours, optimizing energy usage based on grid load, and utilizing renewable energy sources wherever possible.
- **Grid Load Optimization**: Use IoT sensors and data analytics to optimize the distribution of charging demands and predict peak usage times, thus reducing grid stress and minimizing carbon footprints.
- **IoT Integration for Charging Station Monitoring**: Implement IoT sensors in charging stations to monitor usage patterns, charging rates, and maintenance status. This would allow the application to provide users with real-time information on station availability and performance.

- **En Route Charging Support**: The application should facilitate long-distance EV travel by providing users with charging points along their route (based on the source and destination) and real-time information about the availability of these stations.
- **Favorites and Quick Rebooking**: Allow users to save their frequently used charging stations as favorites for quicker future bookings.
- **Integration with Emerging Technologies**: Design the system to easily integrate with new technologies such as V2G (Vehicle-to-Grid) systems, renewable energy sources, and advanced charging technologies to remain relevant in a rapidly evolving market.
- **High Availability**: The system must have high uptime and be reliable, particularly for critical functions such as real-time charging station availability and reservation features.
- **Fast Response Times**: Minimize delays in processing user requests, such as charging slot bookings, real-time station availability queries, and payment processing.

Example:

- **Charging Station A**: 2 miles away, fast charging, available.
- **Charging Station B**: 5 miles away, standard charging, available.
- **Charging Station C**: 3 miles away, fast charging, currently occupied.
- **Notification**: "Charging Station B is undergoing maintenance. The station will be unavailable for the next 2 hours. Find alternative stations nearby."
- **Status**: "Your reserved spot is now available, charging slot ready."
- **Estimated Charging Time**: 45 minutes.
- A user frequently charges their EV at Charging Station A located near their home. After a few visits, the user saves Charging Station A as a favorite in the app.
- Next time they need to reserve a charging slot, the user opens the app and selects Favorite Charging Station A from the list. The app automatically displays available time slots for that station, and the user can quickly reserve their usual spot without having to search for nearby stations again.

- **Recommendation**: “Consider scheduling your charging session during off-peak hours to save on energy costs.”

23 Current Software Architecture

The software architecture of the Electric Vehicle Charging Management System (EVCMS) is designed to be scalable**,** secure, and user-friendly to handle a wide range of tasks, including locating charging stations, scheduling charging sessions, handling real-time updates, and managing payments.

The architecture follows a client-server model, where the frontend is responsible for user interaction, and the backend handles the business logic, user data, charging station information, and communication between users and charging stations.

By separating the frontend and backend, the system can efficiently manage real-time interactions between users and charging stations, ensuring that users can find available stations, reserve charging slots, and monitor their sessions without lag or downtime.

- **Seamless User Experience**: Quick, real-time updates and easy-to-navigate UI.
- **Scalability**: Ability to handle increasing numbers of users and charging stations as the EV market grows.
- **Security**: Protecting sensitive data, including payment information and user profiles.
- **Efficiency**: Optimizing charging schedules and minimizing energy costs for users.
- **Performance**: The system must handle multiple users searching for stations, reserving slots, and tracking charging sessions simultaneously without any noticeable delay.
- **Scalability**: As the number of electric vehicles and charging stations increases, the system must be able to scale horizontally. This may involve adding more servers or utilizing cloud-based solutions that dynamically allocate resources based on user demand.
- **Real-Time Data Handling**: Since the system deals with real-time charging availability, slot reservations, and dynamic pricing, it must handle real-time updates with minimal delay to ensure a smooth user experience.

- **Maintenance and Extensibility**: The system must be easy to maintain and extend. As new features (such as integration with electric grid systems or new payment methods) are added, they should be able to integrate seamlessly without disrupting existing services.

Example Architecture:

The EVCMS application could follow a **three-tier architecture** to separate concerns and optimize performance and scalability:

1. **Frontend (User Interface Layer)**:
   The frontend of the EVCMS application is responsible for user interaction. It displays the map of charging stations, allows users to reserve charging slots, and provides real-time updates on the charging status. The frontend is built using modern web technologies such as **Flutter** for dynamic, responsive interfaces.

   Key features on the frontend include:

   - Real-time updates for station availability.
   - Interactive map to locate nearby charging stations.
   - Options for scheduling, reserving, and paying for charging sessions.

2. **Backend (API Layer)**:
   The backend of the system is responsible for processing business logic, managing user authentication, handling real-time interactions (e.g., booking, status updates), and ensuring efficient scheduling and optimization of charging slots. The backend can be built using **Python** (using frameworks like **Flask**).

   The backend is responsible for:

   - Handling user requests and data (user profiles, payment methods, etc.).
   - Communicating with charging stations to check availability and monitor usage in real-time.

- Sending real-time notifications for maintenance, availability, or charging status updates.

Backend components include:

- **Authentication & Authorization**: Secure user login and access control (using OAuth2).
- **Charging Management**: Logic for station availability, reservations, and scheduling.
- **Payment Integration**: Processing payments using multiple methods (credit cards, e-wallets).
- **Real-Time Data Processing**: Using WebSockets to update users on station availability.

3. **Database Layer (Data Storage)**:
   The database stores all necessary information related to user profiles, charging stations, reservation history, payment records. **SQLITE** can be used based on the nature of the data.

   Key database components include:

   - **User Data**: Stores personal details, payment methods, and reservation history.
   - **Charging Station Information**: Details of stations, including location, charging capacity, status (available, under maintenance), and real-time usage data.
   - **Reservation Data**: Tracks scheduled charging slots, user reservations, and completion statuses.
   - **Payment Data**: Stores transaction history and payment methods used by the users.

Example:

1. **User Search**: A user opens the app and enters their location to search for nearby charging stations. The frontend sends a request to the backend API, which queries the database for stations in the vicinity and returns the results with availability and charging type.
2. **Reservation**: The user selects a station, chooses a time slot, and reserves the charging slot. The frontend sends this reservation request to the backend, which updates the database with the reservation details and confirms it to the user.
3. **Payment**: When the charging session is complete, the user proceeds to pay. The frontend sends the payment details to the backend, which processes the payment using integrated APIs (e.g., Razorpay), and updates the database with the transaction details.
4. **Real-Time Updates**: The backend continuously monitors charging station availability and sends real-time updates to users via WebSockets , ensuring that users are notified of changes, such as maintenance or the availability of previously booked slots.

24 Proposed Software Architecture

24a Overview

The proposed software architecture for the Electric Vehicle Charging Management System (EVCMS) is designed to improve performance**,** scalability, and user experience by leveraging modern technologies and best practices in software engineering. The architecture follows a microservices-based model, where the application is divided into smaller, independent services that can be developed, deployed, and scaled independently. This approach allows for greater flexibility, faster development cycles, and efficient resource management.

1. Frontend (Client-Side):

- The frontend will be built using modern Flutter**.** The UI will include features such as an interactive map to display charging stations, reservation options, user profiles, and payment interfaces.
- The frontend communicates with the backend via RESTful APIs and WebSocket connections for real-time updates.

2. Backend (Server-Side):

- The backend of the EVCMS will be designed using a microservices architecture, allowing for modular services that handle different aspects of the application., such as user management**,** charging station management**,** real-time scheduling, and payment processing. Each microservice will be independently deployable and scalable.
- Each microservice communicates with others via RESTful APIs for non-real-time operations, and WebSockets for real-time communication (e.g., live updates on station availability and charging status).

3. Database:

- The database layer will use a distributed, scalable database system like SQLITE, chosen based on the specific needs of the application. The database will store essential data, such as User Profiles, Charging Stations, Reservation Data, Transaction Data**.**

4. APIs:[8]

- **RESTful APIs**: The backend will expose RESTful APIs for communication between the frontend and the backend. These APIs will handle various operations such as User registration, login, and authentication, Querying charging station availability, locations, and

reservations, Payment processing and transaction management, Accessing and managing user preferences and favorite charging stations.

- **WebSocket APIs**: In addition to RESTful APIs, WebSocket APIs will be used for real-time communication between the frontend and the backend, enabling instantaneous updates on charging station availability, booking status, charging progress, and other time-sensitive information.

6. Charging Session Server:

- The EVCMS will include a dedicated server responsible for managing charging session states, coordinating real-time reservation updates, and synchronizing data between users and charging stations.

24b Dynamic Model [9]

For the EVCMS (Electric Vehicle Charging Management System), we can create dynamic models for each feature. The dynamic model for each feature will outline the sequence of interactions between the system and the users (EV owners, station managers, etc.), as well as how the system responds to events, ensuring smooth and efficient operation. Below is the dynamic model representation for each key feature of the EVCMS.

1. Search Nearest Available Charging Stations

- **Actors**: EV Owner (User), EVCMS System, Charging Station Database
- **User Action**: User initiates a search for nearest available charging stations.
- **System Response**: EVCMS retrieves the current location of the user and queries the charging station database.
- **System Action**: System fetches available stations based on proximity and real-time availability.

- **System Response**: The system displays a list of nearby charging stations, sorted by distance and availability.
- **User Action**: User selects a station from the list.
- **System Response**: System updates the status of the selected station (e.g., available, in-use, under maintenance).

2. Dynamic Scheduling and Real-Time Reservation of Charging Slots

- **Actors**: EV Owner (User), EVCMS System, Charging Station Database, IoT Sensors
- **User Action**: User selects a charging station and requests to schedule a charging session (selecting time or reserving a slot).
- **System Response**: The system checks available time slots for the selected station, considering peak hours, real-time station availability, and user preferences.
- **System Action**: System dynamically schedules the charging slot, confirms the reservation, and updates the station status.
- **User Action**: User receives reservation confirmation with time details and charging duration.
- **System Action**: The system updates charging station status to "Reserved" for that time slot.

3. Notification for Maintenance and Estimated Completion Times

- **Actors**: EVCMS System, Maintenance System, EV Owner (User), Charging Station Database
- **System Action**: IoT sensors detect a maintenance requirement.

- **System Response**: System schedules maintenance or sends an alert to Service technicians.
- **System Action**: The system sends notifications to users about the charging station's unavailability and expected maintenance duration.
- **User Action**: User receives the maintenance notification and adjusts plans if necessary.
- **System Action**: Once maintenance is completed, the system updates the station status and sends a notification about availability.

4. Monitor and Analyze Charging Patterns (IoT Sensors and Optimization)

- **Actors**: EVCMS System, IoT Sensors, EV Owner (User), Charging Station Database
- **System Action**: IoT sensors continuously collect data on charging station usage (e.g., demand, energy consumption, battery levels).
- **System Action**: The system processes the collected data in real-time and analyzes charging patterns.
- **System Response**: The system predicts peak demand periods and adjusts the charging schedule or pricing accordingly.
- **System Action**: Optimized charging schedules are sent to users to minimize energy costs and avoid grid overload.
- **User Action**: User receives an updated schedule or pricing, if applicable.

5. Multiple Payment Options

- **Actors**: EV Owner (User), EVCMS System, Payment Gateway, Charging Station Database

- **User Action**: User initiates payment for a charging session (either pre-scheduled or in-progress).
- **System Response**: The system presents available payment options (credit/debit card, mobile wallet, EV tokens).
- **User Action**: User selects a payment method and enters payment details.
- **System Action**: The system processes the payment using a secure payment gateway and checks for payment confirmation.
- **System Response**: Payment is confirmed, and the system updates the user's transaction history and charging status.
- **System Action**: The system sends the user a confirmation message.

6. Improved Travel Experience (Route-Based Charging Points)

- **Actors**: EV Owner (User), EVCMS System, GPS System, Charging Station Database
- **User Action**: User inputs a travel route (starting point and destination) for long-distance travel.
- **System Action**: The system calculates the best route and identifies charging stations along the way (en route charging points).
- **System Response**: The system displays available charging points at specific intervals, considering battery levels and distance.
- **User Action**: User selects a charging point along the route and reserves a slot if needed.
- **System Action**: The system updates the user's route and charging station reservation.

- **System Response**: The system sends navigation updates and real-time availability data during travel.

7. Favorite Charging Stations (Rebooking Option)

- **Actors**: EV Owner (User), EVCMS System, Charging Station Database
- **User Action**: User adds a charging station to their list of favorite stations.
- **System Action**: The system saves the station for future use.
- **User Action**: In future, the user selects a favorite station from the list for rebooking.
- **System Response**: The system fetches availability for the selected station and presents available slots.
- **User Action**: User selects a time slot and confirms reservation.
- **System Action**: The system reserves the slot and updates the station status.

24c Subsystem Decomposition

For the EVCMS (Electric Vehicle Charging Management System), subsystem decomposition breaks down the system into smaller, manageable components that handle specific responsibilities, ensuring a modular and scalable architecture. Each subsystem focuses on a key part of the system's functionality and interacts with other subsystems to deliver the complete feature set. Below is the decomposition of the subsystems for each feature of EVCMS:

1. Search Nearest Available Charging Stations

- **User Interface (UI) Subsystem**: Manages the user’s input (location search), displays results, and finding nearby charging stations.

- **Location Service Subsystem**: Handles location tracking and geospatial calculations, such as determining the user's current location and finding nearby stations based on geospatial data.
- **Charging Station Database Subsystem**: Stores and retrieves data about charging stations, including availability, station details, and operational status.
- **Data Analytics Subsystem**: Provides real-time insights into station availability, usage patterns, and predictive analytics for managing peak times and grid load.

2. Dynamic Scheduling and Real-Time Reservation of Charging Slots

- **User Interface (UI) Subsystem**: Allows users to schedule or reserve charging slots through an easy-to-navigate interface, displaying available slots and booking options.
- **Charging Slot Management Subsystem**: Manages the availability and allocation of charging slots at stations, ensuring no overbooking occurs and dynamically updating availability.
- **Real-Time Availability Subsystem**: Uses data feeds to check station availability and update the reservation system in real time.
- **Reservation Confirmation Subsystem**: Processes reservation requests, confirms them, and sends confirmations to users via notifications (email, SMS, or app notifications).
- **Payment and Billing Subsystem**: Handles payment processing for reserved slots, including managing multiple payment methods (credit/debit, wallets, etc.).

3. Notification for Maintenance and Estimated Completion Times

- **Maintenance Monitoring Subsystem**: Monitors the status of charging stations through IoT sensors to detect faults, wear and tear of EV's, or scheduled maintenance needs.

- **User Notification Subsystem**: Sends real-time notifications to users regarding maintenance issues or station uptimes, downtimes, as well as updates on the estimated completion time.
- **Station Status Subsystem**: Updates the status of charging stations (e.g., "Under Maintenance", "Available", "In Use") and ensures that this information is reflected across the system for users to view.

4. Monitor and Analyze Charging Patterns (IoT Sensors and Optimization)

- **IoT Sensor Subsystem**: Collects real-time data from IoT sensors deployed at charging stations, such as energy consumption, battery charging rates, and usage patterns.
- **Data Analytics Subsystem**: Analyzes charging data to predict peak usage times, identify patterns in charging behavior, and recommend optimizations for energy distribution.
- **Load Balancing Subsystem**: Optimizes energy distribution to avoid overloading the grid by adjusting charging schedules.

5. Multiple Payment Options

- **User Interface (UI) Subsystem**: Manages payment interfaces, allowing users to select from multiple payment methods (credit card, mobile wallet, etc.) during checkout.
- **Payment Gateway Subsystem**: Integrates with third-party payment processors to securely handle transactions, validate payment details, and process charges for charging sessions or reservations.

6. Improved Travel Experience (Route-Based Charging Points)

- **Route Calculation Subsystem**: Uses GPS and mapping services to calculate the best route for users, factoring in battery consumption, charging needs, and available charging stations along the way.

- **Charging Point Discovery Subsystem**: Identifies charging stations along the user's route, displaying available stations with estimated charging times.
- **User Interface (UI) Subsystem**: Displays the travel route, nearby charging stations, and allows users to book charging slots at stations along the route.
- **Navigation Subsystem**: Integrates with navigation systems to provide turn-by-turn directions and updates as the user progresses along their route, including real-time updates on station availability.
- **Real-Time Availability and Update Subsystem**: Continuously monitors the availability of charging stations along the user's route and sends updates if there are changes in station status (e.g., station is full, out of service).

7. Favorite Charging Stations (Rebooking Option)

- **User Profile Management Subsystem**: Stores user preferences, including favorite stations, booking history, and previous interactions with charging stations.
- **Charging Station Database Subsystem**: Maintains and provides access to a comprehensive list of charging stations, allowing users to mark favorites.
- **Favorite Station Management Subsystem**: Manages the list of favorite charging stations, providing users with easy access to frequently used stations for quick rebooking.
- **Notification Subsystem**: Sends users updates on their favorite stations, including availability, maintenance updates, or promotional offers.

24d Hardware / software mapping

Component	Software	Hardware	Component	Software
Frontend Devices	Mobile/Web Application, User Interface	Smartphones, Tablets, Computers (iOS, Android, Web	Frontend Devices	Mobile/Web Application, User Interface

		browsers)		
Backend Servers	Application Logic, API Services, Data Processing, Real-time Scheduling	Cloud Servers or On-premise Servers	Backend Servers	Application Logic, API Services, Data Processing, Real-time Scheduling
Database Servers	DBMS (Relational/NoSQL), Data Analytics	Database Servers (Cloud or On-premise)	Database Servers	DBMS (Relational/NoSQL), Data Analytics
IoT Sensors & Charging Stations	IoT Data Processing, Maintenance Alerts	IoT Sensors, Charging Stations (physical hardware)	IoT Sensors & Charging Stations	IoT Data Processing, Maintenance Alerts
Real-Time Communication	WebSocket, MQTT, Firebase, Real-time Messaging	Networking Infrastructure (Routers, Switches, Communication Hardware)	Real-Time Communication	WebSocket, MQTT, Firebase, Real-time Messaging
Payment Gateway	Payment Processing Software	Secure Payment Terminals or Cloud Payment Infrastructure	Payment Gateway	Payment Processing Software

Table 3 – Table of Hardware/ Software Mapping:

24e Data Dictionary

Below is the data dictionary for each of the modules in the EVCMS Application:

1. User Registration & Login

SR No.	Attribute	Data Type	Description	Key Constraint
1	user_id	Integer	Unique identifier for each user.	Primary Key, Auto-increment
2	first_name	String (255)	First name of the user.	Not Null
3	last_name	String (255)	Last name of the user.	Not Null
4	email	String (255)	Email address for user authentication and notifications.	Unique, Not Null
5	password	String (255)	Encrypted password for user authentication.	Not Null
6	phone_number	String (15)	User's phone number for contact.	Unique, Optional
7	role	Enum ('user', 'admin', 'operator')	Role of the user within the system (admin, user, operator).	Not Null
8	status	Enum ('pending', 'approved', 'rejected')	Status of user registration. Approved or pending approval.	Default: 'pending', Not Null
9	registration_date	DateTime	Date and time when the user registered.	Not Null
10	last_login	DateTime	Date and time when the user last logged in.	Null

Table 4 – Login and Registration Module Data Dictionary:

2. Charging Station Locator

SR No.	Attribute	Data Type	Description	Key Constraint
1	station_id	Integer	Unique identifier for each charging station.	Primary Key, Auto-increment
2	station_name	String (255)	Name of the charging station.	Not Null
3	location	String (255)	Address or coordinates of the charging station.	Not Null
4	availability_status	Enum ('available', 'busy', 'under maintenance')	Current status of the station.	Not Null
5	connector_type	Enum ('Type 1', 'Type 2', 'CCS', 'CHAdeMO')	Types of connectors supported by the station.	Not Null
6	pricing	Decimal (10, 2)	Price per unit of energy (per kWh).	Not Null
7	operating_hours	String (255)	Operating hours of the charging station.	Null
8	last_updated	DateTime	Timestamp for the last update of station details.	Not Null

Table 5 – Charging Station Locator Module Data Dictionary:

3. Reservation & Booking Management

SR No.	Attribute	Data Type	Description	Key Constraint
1	reservation_id	Integer	Unique identifier for each reservation.	Primary Key, Auto-increment
2	user_id	Integer	Foreign key referencing the	Foreign Key

			user who made the reservation.	(users.user_id)
3	station_id	Integer	Foreign key referencing the charging station reserved.	Foreign Key (stations.station_id)
4	start_time	DateTime	The start time of the reserved charging session.	Not Null
5	end_time	DateTime	The end time of the reserved charging session.	Not Null
6	status	Enum ('reserved', 'completed', 'cancelled')	Status of the reservation.	Not Null
7	favorite_station	Boolean	Flag indicating if the station is marked as a favorite.	Default: False, Not Null

Table 6 – Reservation & Booking Management Module Data Dictionary:

4. Payment Processing

SR No.	Attribute	Data Type	Description	Key Constraint
1	payment_id	Integer	Unique identifier for each payment transaction.	Primary Key, Auto-increment
2	reservation_id	Integer	Foreign key referencing the associated reservation.	Foreign Key (reservations.reservation_id)
3	amount	Decimal (10, 2)	Total amount paid for the charging session.	Not Null

4	payment_date	DateTime	Date and time the payment was made.	Not Null
5	payment_method	Enum ('Credit Card', 'PayPal', 'Digital Wallet')	Method of payment used for the transaction.	Not Null
6	payment_status	Enum ('pending', 'completed', 'failed')	Status of the payment.	Not Null

Table 7 – Payment Processing Module Data Dictionary:

5. Maintenance & Notifications Management

SR No.	Attribute	Data Type	Description	Key Constraint
1	notification_id	Integer	Unique identifier for each notification.	Primary Key, Auto-increment
2	user_id	Integer	Foreign key referencing the user who receives the notification.	Foreign Key (users.user_id)
3	message	String (500)	Content of the notification message.	Not Null
4	notification_type	Enum ('maintenance', 'charging_complete', 'payment_received', 'reservation_confirmed')	Type of notification being sent.	Not Null
5	timestamp	DateTime	Date and time when	Not Null

			the notification was sent.	
6	read_status	Boolean	Flag indicating whether the user has read the notification.	Default: False

Table 8 – Maintenance & Notifications Management Module Data Dictionary:

6. Mapping Services

SR No.	Attribute	Data Type	Description	Key Constraint
1	route_id	Integer	Unique identifier for each route with charging stations.	Primary Key, Auto-increment
2	source_location	String (255)	Starting point of the travel route.	Not Null
3	destination_location	String (255)	Destination point of the travel route.	Not Null
4	charging_station_id	Integer	Foreign key referencing a charging station on the route.	Foreign Key (stations.station_id)
5	station_location	String (255)	Location of the charging station along the route.	Not Null
6	distance	Decimal (10, 2)	Distance to the charging station from the current position.	Not Null
7	estimated_arrival_time	DateTime	Estimated time of arrival at the charging station.	Not Null

Table 9 – Mapping Services Module Data Dictionary:

7. Battery Management System (BMS) Sensor Integration.

SR No.	Attribute	Data Type	Description	Key Constraint
1	bms_id	Integer	Unique identifier for each BMS record.	Primary Key, Auto-increment
2	station_id	Integer	Foreign key referencing the charging station's BMS system.	Foreign Key (stations.station_id)
3	energy_consumed	Decimal (10, 2)	Total energy consumed during a charging session (in kWh).	Not Null
4	peak_usage_time	DateTime	Predicted or actual peak usage time for the charging station.	Not Null
5	load_reduction	Decimal (10, 2)	Energy load reduction achieved by optimized charging schedule.	Not Null
6	timestamp	DateTime	Timestamp for the BMS data entry.	Not Null
SR No.	Attribute	Data Type	Description	Key Constraint

Table 10 – Battery Management System (BMS) Sensor Integration Module Data Dictionary:

8. Favorite Charging Station & Rebooking Management

SR No.	Attribute	Data Type	Description	Key Constraint
1	favorite_id	Integer	Unique identifier for each favorite charging station.	Primary Key, Auto-increment
2	user_id	Integer	Foreign key	Foreign Key (users.user_id)

			referencing the user who marked the station as a favorite.	
3	station_id	Integer	Foreign key referencing the charging station marked as a favorite.	Foreign Key (stations.station_id)
4	is_favorite	Boolean	Flag indicating whether the station is a favorite.	Default: False
5	added_date	DateTime	Date and time when the station was added to the favorites list.	Not Null
6	rebooking_status	Enum ('pending', 'booked', 'cancelled')	Status of the rebooking request for the favorite station.	Default: 'pending', Not Null
7	rebooking_date	DateTime	The date and time when the user last attempted rebooking at the favorite station.	Null
8	reservation_id	Integer	Foreign key referencing the reservation made from the favorite station.	Foreign Key (reservations.reservation_id)

Table 11 – Favorite Charging Station & Rebooking Management Module Data Dictionary:

24f Persistent Data management

For the EVCMS (Electric Vehicle Charging Management System) application, persistent data management plays a critical role in ensuring that user data, charging station availability, payment information, and other system data are accurately stored, accessible, and secure across sessions and system restarts. Below is a breakdown of how persistent data management concepts would apply to the key features of the EVCMS application.

- Given the structured nature of the application data (user profiles, charging stations, payment transactions), a SQLITE would be ideal for storing user profiles, reservation data, and transaction history.
- For example, when a user searches for the nearest charging station, the system retrieves station data based on the user's location, availability status, and charging capabilities.
- When scheduling or reserving charging slots, the system retrieves the user’s preferred charging stations and available time slots.
- The system will regularly update charging station status based on IoT sensor data (e.g., availability, maintenance status). This can be done in real-time via API calls or data push mechanisms from charging stations.
- User preferences, favorite stations, and payment methods will be updated dynamically when a user changes their preferences, updates payment methods, or books/reschedules charging slots.
- When users cancel or modify a reservation, the system will update the corresponding data in the database to reflect the new state.
- Real-time updates to energy cost or grid load data will help optimize charging schedules and load distribution.
- Regular backups will be implemented to protect critical data such as user profiles, reservation records, and payment histories.
- Implemented input validation checks and constraints to ensure that data entered by users (e.g., payment information, charging station IDs) is accurate and follows the correct format.

- Any data corruption or system failure will trigger error logs to monitor the system's state and quickly address any issues.

24g Access control and security

To ensure the integrity, privacy, and seamless user experience of the Electric Vehicle Charging Management System (EVCMS) Application, robust access control and security mechanisms must be implemented.

- Users will be required to authenticate using a combination of their username, password, and a secondary factor (e.g., SMS or email OTP, or biometric data like fingerprint or facial recognition). This ensures that only authorized individuals can access their accounts.
- The system will implement Role-Based Access Control (RBAC) to define different user roles (e.g., Regular Users, EV Owners, Charging Station Administrators, and System Administrators). Each role will have predefined permissions, ensuring that users can only access features relevant to their role.
- Sensitive user data (e.g., payment information, vehicle details, charging history) will be encrypted when stored in databases, using industry-standard encryption algorithms such as AES-256.
- The system will maintain detailed logs of user activities, including login attempts, charging session history, and any administrative actions.

24h Global software control

Global software control in the Electric Vehicle Charging Management System (EVCMS) involves managing and coordinating the entire software ecosystem to ensure smooth operation, scalability, security, and compliance with global standards.

- A centralized version control system (VCS) like Git will be used to manage the source code of the entire EVCMS platform. This allows developers globally to collaborate, track changes, and maintain code integrity.

- To support global payments, the app will integrate with multiple payment gateways (e.g.,Razorpay) to enable payments in various currencies and regions, ensuring that users can easily pay for charging sessions in their local currency.
- The software will ensure compliance with energy and environmental regulations in different regions. For example, it may need to adhere to local government incentives for electric vehicles and energy consumption or abide by regulations for grid load management in various countries.
- A governance framework will guide the development of EVCMS, ensuring that all changes are well-documented, secure, and aligned with business objectives. Agile methodologies (e.g., Scrum) will allow for flexible, iterative improvements while adhering to governance controls.

24i Boundary conditions

To define boundary conditions for the EVCMS Application, we need to specify constraints, limitations, assumptions and any restrictions within the system's design that will guide the development, functionality, and usage of the system.

- The availability of charging stations and their services may be restricted based on the user's geographical location (e.g., only available within certain countries or regions).
- Charging stations can only be listed if they are operational and support real-time availability data.
- The accuracy of the charging station's location on the map must be validated to prevent discrepancies in route planning.
- Charging slots can be reserved for a limited duration (e.g., 30 minutes to 2 hours), and slots must be available within a specified time frame (e.g., within the next 24 hours).
- Users can modify or cancel reservations within a predefined time window (e.g., 15 minutes before the reserved time).
- Users should be notified immediately if a payment fails, and the system should allow retry or alternate payment method selection.

- The system's charging point recommendations for long-distance travel should take into account charging time and distances based on vehicle battery capacity and driving conditions.
- Users will receive notifications about the maintenance schedule or unexpected service interruptions for charging stations they've reserved.
- User data (e.g., reservation history, payment data) should be retained for a certain period (e.g., 6 months) but should be deletable upon user request, in compliance with privacy policies.

25 Subsystem services [10]

For the EVCMS Application outlined in the abstract, we can break down the system into various subsystem services that handle specific features and functionalities. Each service would focus on providing a specific aspect of the overall system. Here is a breakdown of possible subsystem services for each feature mentioned:

1. User Registration and Authentication Service

- Handles user registration, login, authentication, and profile management.
- **Key Services**:
  - Secure login/logout, including two-factor authentication (2FA) and password recovery.
  - Creation, updating, and deletion of user profiles.
  - Differentiates between user roles (e.g., regular user, admin, station operator).

2. Charging Station Discovery and Mapping Service

- Helps users search for nearby or specific charging stations and provides information on their availability.
- **Key Services**:
  - Search and filter nearby available charging stations based on location, station type, charging speed, and amenities.

- Displays details such as charging slot availability, station type, and operating hours.
- Provides interactive map integration, displaying charging stations relative to the user's current or planned location.
- Uses GPS or mobile location to show nearby charging stations in real-time.

3. Dynamic Scheduling and Reservation Service

- Manages real-time scheduling, booking, and reservation of charging slots.
- **Key Services**:
  - Provides up-to-date information on available slots at charging stations.
  - Enables users to book, modify, or cancel charging slots based on availability.
  - If a charging station is fully booked, the system should manage waiting queues and notify users about the slot's availability.
  - Sends confirmations for successful reservations and reminders before the scheduled time.

4. Maintenance and Notification Service

- Sends notifications about maintenance schedules, charging station availability, and other system alerts.
- **Key Services**:
  - Alerts users when a charging station is under maintenance or out of service.
  - Provides real-time updates on charging station status (e.g., operational, busy, offline).
  - Alerts users when maintenance or charging completion is expected.
  - Sends push notifications for upcoming reservations and service interruptions.

5. IoT Integration and Energy Optimization Service

- Uses IoT sensors at charging stations to gather data, analyze usage patterns, and optimize energy consumption.
- **Key Services**:
  - Analyzes historical charging patterns to predict peak usage times and demand.
  - Suggests optimal charging times to minimize energy costs and reduce strain on the grid (e.g., off-peak charging).
  - Manages the load on the power grid by distributing demand across stations during peak and off-peak hours.

6. Payment Gateway Service

- Manages the integration of multiple payment methods and processes financial transactions.
- **Key Services**:
  - Supports various payment options, such as credit/debit cards, mobile wallets, UPI and in-app payments.
  - Handles payment authorization, verification, and completion for charging sessions.
  - Provides users with a history of payments and allows them to download invoices.

7. Route Planning and En-Route Charging Points Service

- Supports long-distance travel by suggesting charging stations along the user's route.
- **Key Services**:
  - Calculates the best route from the user's source to destination, considering current charging station availability and vehicle range.
  - Displays recommended charging stations along the route, taking into account user preferences (e.g., charging speed).

- Updates the route in real-time based on charging station status (e.g., a station becomes unavailable).
- Adjusts routes based on real-time traffic conditions, detours, or delays.
- Allows users to save and review past trips and charging Stations used.

8. Favorites and Frequent Station Management Service

- Allows users to save frequently used charging stations for easy future reservations.
- **Key Services**:
  - Stores frequently used routes, charging stations, and preferences for seamless future experiences.
  - Allows quick rebooking at a favorite station with a single tap.

9. Data Analytics and Reporting Service

- Provides analytics related to charging behavior, station usage, and energy consumption patterns.
- **Key Services**:
  - Generates reports on station usage, charging times, and energy consumption trends.
  - Analyzes user charging patterns, including most frequently used stations and peak times.
  - Provides detailed reports on energy consumption and grid load reduction.
  - Provides charging station operators and administrators with dashboards for monitoring station performance and user metrics.

10. System Administration and Management Service

- Manages the administration of the application, including adding/removing charging stations, updating system configurations, and monitoring overall system health.
- **Key Services**:
  - Allows admins to add, remove, or update charging stations in the network.

- Manages system settings, such as pricing models, availability hours, and station statuses.
- Handles user account administration, including managing user roles, permissions, and access.
- Tracks system performance, logs errors, and monitors usage trends.

26 User Interface

Screens already added in 22. System Design Section

For the EVCMS Application (Electric Vehicle Charging Management System), the user interface (UI) will play a crucial role in delivering a seamless and intuitive experience for users.

1. Home Screen

- A Navigation bar that Links to key sections such as Home, Search, Profile, Leaderboard, and Settings.
- Prominent buttons for users to log in or register an account.
- A short, interactive section that highlights the key benefits of the app (e.g., “Find the nearest charging station,” “Plan your long-distance travel,” etc.).
- Buttons for quick actions, such as searching for nearby charging stations or viewing available routes with charging stations.

2. Charging Station Search Screen

- A search field where users can input a location (e.g., city, zip code) to search for nearby charging stations.
- A map interface showing the user’s location and nearby charging stations. Users can zoom in/out and click on individual station markers for more details.
- Options to filter stations by type (e.g., fast charging, standard charging), availability, or other criteria.
- Each station listed in the search results can be displayed as a card with key details (location, status, distance from user, type of charging, and availability).

3. Reservation and Scheduling Screen

- Displays current reservation details such as the station, scheduled time, and remaining time until the slot becomes available.
- Buttons to make a new reservation, cancel, or modify an existing reservation.
- A dynamic section showing charging slots available in real time at a particular station.
- If the user is looking to optimize charging schedules, they can access a list of available slots based on the predicted peak usage times.

4. Profile Screen

- Displays the user's profile information (name, profile picture, email, etc.).
- A section showing the user's previous charging history (stations used, reservation times, energy consumption, costs, etc.).
- Options to edit profile information, such as name, contact information, and preferred payment methods.
- Allows users to set preferences for notifications (e.g., maintenance alerts, reservation reminders) and payment methods.

5. Route Planning and Charging Points Along the Way Screen

- Users input their starting point and destination to plan a route.
- Interactive map showing the user's route along with charging points along the way.
- Option to automatically optimize the route based on current station availability, expected charging times, and user preferences.
- Provides an overview of charging stations along the route, with estimated times to reach each station and the expected charging time.

6. Sensor Data-Display Screen:

- Graphical representation of energy consumed by users at different times (e.g., hourly, daily).

- A dynamic heatmap showing peak usage times across different days or hours of the day.
- Predictive analytics based on past usage patterns, forecasting the busiest times and suggesting optimal times for charging to avoid congestion.
- Shows Visualize patterns (e.g., whether the user charges more during peak times or off-peak times).
- Displays the average time taken to charge the user's vehicle across different sessions.

7. Maintenance & Notifications Screen

- A section showing scheduled maintenance for stations, system updates, and real-time outage alerts.
- Displays any upcoming notifications related to reservations (e.g., "Your reservation is 30 minutes away," "Station under maintenance").
- A log of previous maintenance alerts, with options to check the duration of previous outages and any recurring issues.
- Allows users to manage which notifications they receive (e.g., push notifications, email alerts).

8. Payment and Transaction History Screen

- Displays the user's stored payment methods and allows for easy editing or adding of new methods.
- Lists all past transactions, including charging costs, payments made, and any discounts applied.
- Option to download a transaction receipt or invoice for each charging session.
- Users can set default payment methods or select preferred payment options.

27 Object Design

27a Object Design trade-offs

Object design tradeoffs involve balancing various factors when creating object-oriented designs, focusing on how to best structure objects and their interactions. This includes choosing between different design options to meet system requirements while addressing constraints and goals. Also by Minimizing coupling between classes while maximizing cohesion within classes can simplify debugging, testing, and future modifications.

- **Trade-off in User Registration & Login**:
  Instead of creating separate classes for different types of users (e.g., AdminUser, RegularUser), a single User class can handle both types using attributes like role and methods such as approve().
- **Trade-off in Mapping Services**:
  Instead of creating separate classes for different types of maps (e.g., RouteMap, ChargingStationMap, EnRouteMap), a single Map class with attributes like startPoint, destination, and waypoints can handle all map-related functions.
- **API Endpoint: /process_payment**
  - Processes the payment for a charging session. Integrates with a payment gateway for transaction processing.
  - A developer calls the /process_payment endpoint with user payment details (e.g., payment method, amount, transaction ID) to complete a payment.
  - If a "Payment failed" error occurs, the developer should check the provided payment details (e.g., card number, expiration date). For "Transaction declined," the developer should ensure the payment method is valid and has sufficient funds.
- **API Endpoint: /search_stations**
  - Allows EV owners to search for nearby charging stations based on their location.

- A developer calls the /search_stations endpoint to retrieve a list of stations within a given radius of the user's location, providing latitude and longitude as parameters.
- If a "No stations found" error is returned, the developer should check the provided location or radius. A "Location data not valid" error indicates the user's location data may be missing or incorrectly formatted.

27b Interface Documentation guidelines

The primary goal of interface documentation is to provide clarity to developers, maintainers, and other stakeholders on how system components interact. This ensures smooth collaboration, minimizes the likelihood of errors, and simplifies the process of integrating new features or fixing bugs.

1. User Registration & Login Interface

- Allows users to sign up, log in, and manage their accounts, with an admin reviewing and approving new user registrations.
- **Inputs**:
  - **Registration Details**: User information such as name, email, phone number, and password.
  - **Login Details**: Email and password for authentication.
  - **Admin Approval**: Admin can approve or reject the user's registration.
- **Outputs**:
  - **User ID**: A unique identifier for the user.
  - **Login Response**: Confirmation of successful login or error message.

2. Charging Station Locator Interface

- Users input their location to search for nearby charging stations and view station details such as availability, pricing, and connector types and reserve a slot. Admins can update station information.
- **Inputs**:
  - **User Location**: Latitude and longitude of the user's current location.
  - **Search Radius**: Distance (in kilometers) to search for stations.
- **Outputs**:
  - **Charging Stations List**: A list of nearby charging stations with details such as availability, pricing, and connector types.
  - **Reservation Link**: A link for the user to reserve a charging slot at a selected station.

3. Reservation & Booking Management Interface

- Allows users to book charging slots and admins/operators to manage and update bookings. Users can also manage favorite stations for easy rebooking.
- **Inputs**:
  - **Booking Details**: User ID, station ID, booking slot (date/time), and duration of the session.
  - **Favorite Station**: User can select a station to mark as a favorite for future bookings.
- **Outputs**:
  - **Booking Confirmation**: A confirmation message with the booking details and unique booking ID.
  - **Favorite Station List**: A list of stations marked as favorites by the user.

4. Payment Processing Interface

- Facilitates payments for charging sessions and generates transaction records.
- **Inputs**:
  - **Payment Details**: User ID, booking ID, amount to be paid, and payment method (e.g., credit card, PayPal).
  - **Transaction Information**: Payment gateway integration details and status of the transaction.
- **Outputs**:
  - **Transaction ID**: A unique identifier for the payment transaction.
  - **Payment Status**: Indicates whether the payment was successful, failed, or pending.

5. Maintenance and Notifications Interface

- Sends notifications to users about charging status, maintenance updates/Schedules, and upcoming bookings alerts.
- **Inputs**:
  - **User Notification Preferences**: User's preferred method of receiving notifications (email, SMS, app notifications).
  - **Notification Type**: Type of notification, such as charging completed, maintenance, booking reminder.
- **Outputs**:
  - **Notification Message**: A message sent to the user (e.g., "Charging complete," "Scheduled maintenance").

6. Mapping Services(EN/ON Route) Interface

- Provides turn-by-turn navigation to the nearest charging station and identifies charging stations along a travel route and plan their route with multiple charging stations along the way.

- **Inputs**:
    - **User Location**: Latitude and longitude of the user's current position.
    - **Destination**: User's travel destination or end-point.
    - **Route Details**: Data for identifying charging stations along the route.
- **Outputs**:
    - **Navigation Instructions**: Step-by-step directions to reach the selected charging station.
    - **Route with Charging Stations**: List of charging stations along the travel route.

7. Battery Management System (BMS) and Energy Management Interface

- Monitors real-time charging data, predicts peak usage times, and optimizes charging schedules to minimize costs and reduce grid load.
- **Inputs**:
    - **Charging Data**: Data from sensors at charging stations about current load and energy consumption.
    - **User Preferences**: Desired charge level for the vehicle and any scheduling preferences.
- **Outputs**:
    - **Energy Optimization Plan**: A plan to reduce grid load by scheduling charging times.
    - **Charging Schedule**: A time slot for when the vehicle will be charged to optimize energy use and cost.

27c Packages

Packages are a way to group related classes and objects, making system modular and easier to navigate. Packages make system easier to understand and maintain by logically grouping similar functions. This improves code reuse and allowing for better collaboration among developers working on different modules.

1. com.evcharging.usermanagement

Contains classes related to user registration, authentication, and profile management.

- **User**: Represents a user in the system. Contains properties like userId, username, email, phoneNumber, and role.
- **UserProfile**: Manages user details like address, payment methods, and preferences.
- **AuthenticationService**: Handles the login, registration, and session management, including the process of generating authentication tokens and managing user sessions.
- **AdminApprovalService**: Handles the review and approval/rejection of new user registrations by the admin.

2. com.evcharging.stationlocator

Includes classes for searching and locating charging stations.

- **ChargingStation**: Represents a charging station. Contains details such as stationId, location, availability, pricing, connectorTypes, and reservationLink.
- **StationSearchService**: Handles the logic for searching nearby charging stations based on user location and search radius. Also manages filtering and sorting stations based on availability and pricing.
- **StationReservationService**: Manages the process of reserving charging slots at selected stations, including availability checks and slot booking.

3. com.evcharging.booking

Features classes related to charging slot reservations, booking management, and user preferences.

- **Booking**: Represents a reservation or booking made by a user for a charging session. Contains properties such as bookingId, userId, stationId, slot, duration, and status.
- **BookingManager**: Handles all operations related to bookings, including creating new bookings, canceling, updating booking status, and managing user reservations.
- **FavoriteStationService**: Allows users to save frequently used charging stations for easy rebooking. Provides functionality for adding, removing, and retrieving favorite stations.

4. com.evcharging.payment

Handles all payment processing and transaction management.

- **PaymentTransaction**: Represents a payment made for a charging session. Contains properties like transactionId, userId, bookingId, amount, paymentMethod, and status.
- **PaymentGatewayService**: Integrates with third-party payment gateways (e.g., Stripe, PayPal) to process transactions. Handles payment authorization, processing, and confirmation.
- **InvoiceService**: Generates invoices and receipts for successful transactions. Provides users with downloadable or emailed transaction receipts.

5. com.evcharging.notification

Manages user notifications, including charging updates, maintenance alerts, and reminders.

- **Notification**: Represents a notification message sent to users. Contains notificationId, userId, message, type, and status.
- **NotificationManager**: Handles sending notifications to users for various events like reservation confirmations, charging status updates, or system maintenance.

- **MaintenanceNotificationService**: Specifically sends notifications related to scheduled maintenance of charging stations or vehicles.

6. com.evcharging.mapping

Includes classes for providing navigation and route mapping to users, along with station routing along a travel path.

- **Route**: Represents a user's route, containing start and destination coordinates, and a list of charging stations along the route.
- **NavigationService**: Provides turn-by-turn directions to the nearest charging station based on the user's current location.
- **RoutePlanner**: Helps users plan long-distance routes by identifying charging stations along the way, ensuring that stations are available along the route.

7. com.evcharging.admin

Contains classes and services used by admins to manage users, stations, and overall system maintenance.

- **AdminUserManagement**: Provides admins with tools to manage user registrations, review approval requests, and update user roles.
- **StationManagementService**: Allows admins to add, update, or remove charging stations from the system. Manages station availability and maintenance schedules.
- **BookingManagementService**: Lets admins monitor and manage all bookings in the system, including resolving conflicts and adjusting availability.
- **SystemMaintenanceService**: Manages overall system maintenance tasks and notifies users of any downtime or scheduled maintenance activities.

8. com.evcharging.integration

Includes classes for integrating the system with external services, such as mapping providers, payment gateways, and third-party APIs.

- **PaymentGatewayIntegration**: Integrates with external payment processors to handle payment transactions, including fraud prevention and security checks.
- **MappingServiceIntegration**: Integrates with third-party mapping services like Google Maps or OpenStreetMap to provide real-time navigation and route planning to users.
- **IoTIntegrationService**: Connects to IoT devices for real-time monitoring of charging stations, battery health, and energy consumption.

27d Class Interfaces

Class interfaces promote modularity and abstraction in the system. By defining clear interfaces, developers ensure that objects of different classes can work together seamlessly without needing to know the inner workings of other classes. This results in cleaner code, easier maintenance, and the flexibility to update or replace parts of the system without disrupting the whole.

1. User Class Interface:

Attributes:

- userId: Integer
- name: String
- email: String
- password: String
- role: String (values: EvOwner, Admin, ServiceStaff)

Methods:

- register(): Registers the user in the system.
- login(): Authenticates the user and allows them to log in.
- logout(): Logs the user out of the system.

Use Case:

- The user registers by calling the register() method to create an account.Once registered, they log in using the login() method with their credentials. After logging in, the user can access the system based on their assigned role (e.g., EV Owner, Admin, Service Staff).When done, the user logs out by using the logout() method.

2. Admin Class Interface:

Attributes:

- adminId: Integer

Methods:

- approveUserRegistration(): Approves or denies user registration requests.
- manageChargingStations(): Manages the charging stations (add, modify, or remove stations).
- viewReports(): Views system reports, such as station usage and revenue.
- viewGraphs(): Displays graphical data (e.g., performance metrics or usage statistics).

Use Case:

- The admin logs into the system using their credentials. The admin manages charging stations with the manageChargingStations() method, adding a new station or modifying existing ones. The admin can also generate reports and

view system graphs using the viewReports() and viewGraphs() methods to track system performance and usage.

3. **EvOwner Class Interface:**

   Attributes:

   - vehicleId: String
   - favoriteStations: List<ChargingStation>

   Methods:

   - viewNearestStations(List<ChargingStation>): Views the nearest charging stations.
   - reserveChargingSlot(): Reserves a charging slot at a chosen station.
   - addFavoriteStation(List<ChargingStation>): Adds a station to the user's list of favorite stations.
   - makePayment(Payment): Makes a payment for the reservation.

   Use Case:

   - The EV Owner logs into the system.They use the viewNearestStations() method to find available charging stations near their location. After selecting a station, they call reserveChargingSlot() to reserve a slot at the station. Once the reservation is confirmed, the EV Owner uses the makePayment() method to complete the payment for the reservation. The EV Owner can also add favorite stations to their profile with addFavoriteStation().

4. **ServiceStaff Class Interface:**

   Attributes:

   - staffId: Integer
   - assignedStations: List<ChargingStation>
   - alertId: Integer

Methods:

- performMaintenance(): Performs maintenance tasks on assigned charging stations.
- sendAlert(): Sends maintenance or operational alerts to the system.
- updateAlertStatus(status): Updates the status of maintenance alerts.

Use Case:

- The Service Staff logs into the system. They receive an alert about a malfunctioning charging station using the receiveMaintenanceAlert() method. The staff member reviews the issue and performs the necessary maintenance using the performMaintenance() method. Once maintenance is completed, the Service Staff updates the alert status to "Completed" using the updateAlertStatus() method.

5. **ChargingStation Class Interface:**

Attributes:

- stationId: Integer
- location: String
- availability: Boolean
- pricePerUnit: Float
- connectorType: String
- serviceId: Integer
- userId: Integer
- reservationId: Integer
- reservationTime: DateTime
- status: String (values: Booked, Completed, Canceled)

Methods:

- checkAvailability(): Checks if the charging station is available for booking.

- displayDetails(): Displays information about the station (e.g., location, price, availability).
- updateAvailability(Boolean availability): Updates the availability status of the station.
- createReservation(): Creates a reservation for the station.
- updateReservation(): Modifies an existing reservation.
- cancelReservation(): Cancels an existing reservation.

Use Case:

- The ChargingStation uses the checkAvailability() method to check if the station is available for booking. The user views details of the station using displayDetails(), which shows the location, price per unit, and availability. Once the user decides to proceed, the station creates a reservation with the createReservation() method. If the station's status changes (e.g., a reservation is canceled), the system can update the station's availability via updateAvailability().

6. **IoTSensors Class Interface:**

Attributes:

- sensorId: Integer
- stationId: Integer
- sensorType: String (values: Temperature, Voltage, Load)
- sensorData: Float
- batteryHealth: String
- energyConsumption: Float

Methods:

- sendSensorData(): Sends the data collected by the sensor to the system for processing.

- monitorBatteryHealth(): Monitors and reports the health of the station's battery.
- optimizeEnergyUsage(): Optimizes energy usage based on sensor data to improve efficiency.

Use Case:

- The IoT sensors collect data and send it to the system using sendSensorData(). They also monitor battery health and energy consumption using monitorBatteryHealth() and optimizeEnergyUsage() to optimize station efficiency.

7. **Payment Class Interface:**

Attributes:

- paymentId: Integer
- userId: Integer
- reservationId: Integer
- amount: Float
- paymentMethod: String (values: CreditCard, Wallet, etc.)

Methods:

- processPayment(): Processes the payment for a reservation.
- generateInvoice(): Generates an invoice for the processed payment.

Use Case:

- The EV Owner completes their reservation, and the system generates a payment request. The EV Owner uses the processPayment() method to pay for the reservation via their selected payment method (e.g., Credit Card, Wallet). After payment is successful, the system generates an invoice using the generateInvoice() method.

IV Test Plans

28 Features to be tested / not to be tested

This section is to provide a clear focus on the areas where testing should be prioritized. Testing is often resource-intensive, so focusing on high-impact, business-critical features ensures that the core functionalities are working smoothly and without errors. It also helps stakeholders to manage testing expectations and resources by clearly identifying what is not included in the current test scope, avoiding confusion or assumptions.

To be Tested:

- **User Registration:** Verify that users can register successfully with valid details.
- **Login Process:** Ensure users can log in using valid credentials.
- **Search Functionality:** Verify the ability to search for nearby charging stations based on the user's location.
- **Filters:** Test the application of filters (e.g., pricing, connector types, availability) in the search results.
- **Station Availability:** Ensure accurate display of charging station availability.
- **Reservation Process:** Ensure users can successfully reserve a charging slot.
- **Payment Integration:** Test the payment flow during the reservation process.
- **Booking Confirmation:** Verify users receive a confirmation after booking a slot.
- **Payment Process:** Ensure payments are processed successfully through integrated payment gateways and done securly.
- **Energy Consumption:** Test the system's ability to track and report energy consumption during charging sessions.
- **Location Tracking:** Verify that the system correctly tracks the location of charging stations.
- **Route Mapping:** Ensure the system accurately provides directions to the nearest charging station.

Not to be Tested:

- **Third-Party Authentication:** Testing third-party authentication systems (e.g., Google Login, Facebook Login) is not included.
- **Captcha Functionality:** Captcha validation is assumed to be correctly implemented by the third-party service and will not be tested in this scope.
- **Charging Station Hardware:** Testing of charging station hardware functionality (e.g., cables, ports) is outside the scope.
- **Payment Gateway Failures:** Only the system's interaction with the payment gateway will be tested, not the gateway itself.
- **Hardware Functionality:** Testing of actual battery hardware is not within the scope (e.g., battery cells, BMS sensors).
- **External Mapping APIs:** Testing of the underlying map APIs (e.g., Google Maps) is excluded unless specified.
- **Offline Mapping:** Mapping functionality without an internet connection is out of scope.

29 Pass/Fail Criteria

This section defines the criteria used to determine whether a particular module in EVCMS has passed or failed testing. These criteria help ensure consistency in the testing process and provide clear guidelines for evaluating the success or failure of the tests.[11]

- **Pass:** The system behaves as expected, with no critical issues.
- **Fail:** The system does not meet the expected behavior or results in errors.

1. User Registration & Login

- **Pass Criteria:**
  - The user is able to successfully register, log in, and access their account with valid credentials.
  - The system validates email formats and passwords, rejecting invalid input.

- **Fail Criteria:**
  - The system allows registration/login with invalid or incomplete credentials (e.g., invalid email format or weak passwords).
  - Login fails despite entering correct credentials.

2. Charging Station Locator

- **Pass Criteria:**
  - The system accurately identifies and displays charging stations based on user location.
  - The charging station information (e.g., availability, connector types, pricing) is up-to-date and correctly shown.
  - Users can filter stations by specific criteria such as connector type or availability.
- **Fail Criteria:**
  - Station details are outdated or inaccurate (e.g., showing incorrect availability).
  - The map or filtering options fail to work as expected or are slow to update.
  - Station reservation and selection cannot be performed or have bugs (e.g., wrong data displayed).

3. Reservation & Booking Management

- **Pass Criteria:**
  - Users can reserve charging stations successfully, with accurate details (time, location, type of connector, etc.).
  - Booking confirmation notifications are sent to the user via email or in-app notifications.
- **Fail Criteria:**
  - The system allows double booking of charging slots or fails to update availability.
  - The system fails to show the correct slot availability or errors occur when users attempt to book or modify a slot.

4. Payment Processing

- **Pass Criteria:**
  - Payments are processed securely through multiple payment methods, including credit/debit cards, wallets, etc.
  - Payment amounts are calculated correctly based on usage and pricing policies.
  - Users receive accurate and immediate payment confirmations.
- **Fail Criteria:**
  - Payment gateway errors are not handled or lead to system crashes.
  - Payment options are unavailable, or there is a lack of proper error messaging in case of payment failures.

5. Maintenance and Notifications

- **Pass Criteria:**
  - Maintenance notifications are timely, informative, and easy to understand.
- **Fail Criteria:**
  - Notifications are missing or do not contain relevant information (e.g., wrong dates/times for maintenance).
  - Users experience notification spam, incorrect notifications, or notifications are sent at incorrect times.

6. Mapping Services

- **Pass Criteria:**
  - The map displays the user's current location and nearby charging stations accurately.
  - The system shows real-time updates on station availability (e.g., available or occupied slots).
  - Users can zoom in/out, filter, and search for stations based on various criteria (e.g., location, connector type).
- **Fail Criteria:**
  - The map is not interactive or shows incorrect or outdated data.

- The map or directions lead users to the wrong charging station or incorrect routes.

7. Battery Management System (BMS) Integration

- **Pass Criteria:**
  - The system prevents overcharging by automatically stopping when the battery is fully charged.
  - Integration with the BMS ensures that all battery-related information (e.g., health, temperature) is communicated accurately to the user.
- **Fail Criteria:**
  - The system allows overcharging or fails to stop the charging process when the battery is full.
  - Integration errors occur, causing inaccurate or delayed battery-related data.

30 Approach

This section outlines the testing strategy used to evaluate the system's performance and functionality. It describes the methodologies, tools, and processes that will be employed during testing Process.

1. User Registration & Login

- **Testing Types and Execution Plan:**
  - **Phase 1 (Unit Testing):** Test individual login, registration, and password reset functionality.
  - **Phase 2 (Integration Testing):** Verify registration and login processes integrate with the user management system.
  - **Phase 3 (System Testing):** Validate the complete flow from account creation to successful login.
  - **Phase 4 (User Acceptance Testing (UAT)):** Have real users test account creation and login for ease of use.

2. Charging Station Locator

- **Testing Types and Execution Plan:**
  - **Phase 1 (Unit Testing):** Test the search and filter features for accurate station results.
  - **Phase 2 (Integration Testing):** Ensure the integration of real-time data and map functionality for station availability.
  - **Phase 3 (System Testing):** Test the full flow of searching, filtering, and reserving a charging station.
  - **Phase 4 (User Acceptance Testing (UAT)):** Conduct user testing to confirm the map's ease of use and accuracy in locating stations.

3. Reservation & Booking Management

- **Testing Types and Execution Plan:**
  - **Phase 1 (Unit Testing):** Test individual reservation and booking slot selection.
  - **Phase 2 (Integration Testing):** Ensure bookings sync with the backend system to reflect real-time availability.
  - **Phase 3 (System Testing):** Validate the full end-to-end reservation process.
  - **Phase 4 (User Acceptance Testing (UAT)):** User feedback on ease of reservation, modification, and cancellation processes.

4. Payment Processing

- **Testing Types and Execution Plan:**
  - **Phase 1 (Unit Testing):** Test individual payment gateway integrations and calculations.
  - **Phase 2(Integration Testing):** Ensure payment processing works seamlessly across all modules.
  - **Phase 3 (System Testing):** Validate the entire payment flow, from charging to payment confirmation.
  - **Phase 4 (User Acceptance Testing (UAT)):** Conduct user testing for payment completion and user interface experience.

5. Mapping Services

- **Testing Types and Execution Plan:**
  - **Phase 1 (Unit Testing):** Test the basic map and location services functionality.
  - **Phase 2 (Integration Testing):** Ensure real-time station availability and accurate route calculations.
  - **Phase 3 (System Testing):** Test the complete flow of searching, filtering, and navigating to charging stations.
  - **Phase 4 (User Acceptance Testing (UAT)):** Have users test the map for ease of use and accuracy in providing routes and station info.

6. Battery Management System (BMS) Integration and Energy Management

- **Testing Types and Execution Plan:**
  - **Phase 1 (Unit Testing):** Test individual data collection and reporting of battery status.
  - **Phase 2 (Integration Testing):** Ensure integration with the charging system for real-time energy management.
  - **Phase 3 (System Testing):** Validate the full battery management process, from charging to report generation.
  - **Phase 4 (User Acceptance Testing (UAT)):** Have users test battery status visibility and overall system accuracy.

31 Suspension and resumption

This section specifies the conditions for suspending (pausing) and resumpting continuing) testing activities. Testing may be suspended when critical issues are encountered that hinder further progress. Testing resumes once these issues are resolved or when the conditions necessary for testing readiness are met. Suspension and resumption guidelines allow the testing team to handle issues like critical bugs, hardware failures, or other unforeseen complications in a controlled manner.

1. **Reservation & Booking Management**

    - **Suspend:** If a major bug prevents users from completing or confirming a booking (e.g., the system fails to reserve a slot, displays incorrect availability, or crashes during the reservation process), testing is paused.
    - **Resume:** After the bug is fixed (e.g., reservations are correctly processed, availability is updated in real-time, and the reservation confirmation is successful), testing resumes from the point of slot selection and reservation confirmation to ensure the entire booking flow works as expected.

2. **Mapping Services**

    - **Suspend:** If a major bug causes the mapping service to fail (e.g., routes not being displayed, incorrect directions, or inability to find nearby charging stations), testing is paused.
    - **Resume:** After the bug is fixed (e.g., accurate maps and routes are displayed, and users can see nearby charging stations), testing resumes from the point where the user requests directions or searches for charging stations on the map to ensure seamless navigation functionality.

3. **Payment Processing**

    - **Suspend:** If a critical bug is found where the payment gateway fails to process transactions (e.g., payments are not being processed, or the user is charged but no confirmation is sent), testing is paused.
    - **Resume:** After the payment gateway integration issue is resolved (e.g., payments successfully process and confirmations are sent to the user), testing resumes from the payment submission stage to validate successful transactions and related functionalities.

32 Testing materials (hardware / software requirements)

This section outlines the hardware and software requirements for testing. It covers the necessary devices, operating systems, browsers, testing tools, and any specific environments or configurations needed to support the testing process.

- **Hardware:** A laptop or desktop with at least:
  - 8GB RAM
  - Intel i5 processor (or equivalent)
  - 500GB HDD or SSD storage
  - Charging Stations should have BMS(Battery Management Sensor having temperature, voltage, current ad load sensor)
- **Software:**
  - **Frontend Testing:** Google Chrome, Mozilla Firefox, Microsoft Edge.
  - **Backend Testing:** Sqlite for database management, Postman for API testing and validation, Operating System : Recommended(Windows 10 or 11).
  - **Automation Testing:** Playwright WebDriver for automated UI testing
- **Network:** A stable internet connection with at least 10 Mbps bandwidth required for remote server access (e.g., for frontend deployment on Netlify and backend on Render or any similar cloud platforms).
- **Test Data:** Mock Data for Pre-loaded user accounts (Admin, EV Owners, Service Staff, etc.). Charging stations with mock data for availability, pricing, and booking status. Payment records, reservation history, and maintenance alerts in SQLite.
- **Test Environment:** Test servers configured to mirror the production environment. This ensures that the testing scenarios closely resemble real-world usage conditions.

33 Test cases

A test case is a documented set of conditions, inputs, and actions used to evaluate whether a specific feature of an application behaves as expected. The purpose of test cases is to ensure that the software functions correctly under various conditions and meets the requirements specified. [12]

1. User Registration & Login

Test Case 1: Successful User Registration

- **Test Case ID:** TC-UR-001
- **Description:** Verify successful registration with valid details.
- **Preconditions:** User is not registered in the system.
- **Test Steps:**
  1. Open the registration page.
  2. Enter valid details (Name, Email, Password, etc.).
  3. Click "Submit".
  4. Admin reviews and approves registration.
  5. User receives a confirmation email.
  6. User logs in successfully.
- **Expected Results:** User is successfully registered and can log in.
- **Status:** Pass

Test Case 2: Registration with Missing Required Fields

- **Test Case ID:** TC-UR-002
- **Description:** Test registration when required fields are missing (e.g., Name, Email).
- **Preconditions:** None.
- **Test Steps:**

  1. Open the registration page.
  2. Leave one or more required fields empty (e.g., Name or Email).
  3. Click "Submit".

- **Expected Results:** An error message should appear indicating the missing required fields.
- **Status:** Fail

Test Case 3: Registration with Invalid Email Format

- **Test Case ID:** TC-UR-003
- **Description:** Test registration with an invalid email format.
- **Preconditions:** None.
- **Test Steps:**

  1. Open the registration page.
  2. Enter an invalid email format (e.g., "john.doe.com").
  3. Enter valid Name and Password.
  4. Click "Submit".

- **Expected Results:** An error message should appear indicating that the email format is invalid.
- **Status:** Fail

Test Case 4: Duplicate Email Registration

- **Test Case ID:** TC-UR-004
- **Description:** Test user registration with an email that is already registered in the system.
- **Preconditions:** An email address must already exist in the system.
- **Test Steps:**

  1. Open the registration page.
  2. Enter a previously registered email.
  3. Enter valid Name and Password.
  4. Click "Submit".

- **Expected Results:** An error message should appear stating that the email is already registered.
- **Status:** Fail

2. Charging Station Locator

Test Case 1: Search for Nearby Charging Stations

- **Test Case ID:** TC-CS-001
- **Description:** Test the functionality of searching for nearby charging stations.
- **Preconditions:** User is logged in and location services are enabled.
- **Test Steps:**

  1. Open the app.
  2. Allow location access.
  3. Click "Search Charging Stations".
  4. The app detects the user's location and lists nearby charging stations.

- **Expected Results:** The app should display nearby charging stations with availability and pricing.
- **Status:** Pass

Test Case 2: Search Without Location Services

- **Test Case ID:** TC-CS-002
- **Description:** Test the search functionality when location services are disabled.
- **Preconditions:** User is logged in.
- **Test Steps:**

  1. Open the app and disable location services.
  2. Attempt to search for nearby charging stations.

- **Expected Results:** The app should prompt the user to enable location services.
- **Status:** Fail

Test Case 3: Search for Charging Stations with Filters

- **Test Case ID:** TC-CS-003
- **Description:** Test the filtering option for charging stations (e.g., price, station type).
- **Preconditions:** User is logged in and location services are enabled.
- **Test Steps:**

  1. Open the app and allow location access.
  2. Click "Search Charging Stations".
  3. Apply filters (e.g., Price: Low to High, Connector Type: Type 2).
  4. View the filtered list of stations.

- **Expected Results:** The app should display filtered results based on the selected criteria.
- **Status:** Pass

Test Case 4: Search for Charging Station in Remote Area

- **Test Case ID:** TC-CS-004
- **Description:** Test searching for charging stations in a remote area where no stations are available.
- **Preconditions:** User is logged in.
- **Test Steps:**

  1. Open the app and search for stations in a remote area.
  2. The app should show no results or display a message indicating no stations are available.

- **Expected Results:** The app should inform the user that no stations are available in the selected area.
- **Status:** Pass

3. Reservation & Booking Management

Test Case 1: Successful Reservation of Charging Slot

- **Test Case ID:** TC-RB-001
- **Description:** Verify that the user can successfully reserve a charging slot.
- **Preconditions:** User must be logged in, and charging stations must be available for booking.
- **Test Steps:**

  1. Log into the app.
  2. Search for an available charging station.
  3. Select a station and choose a time slot.
  4. Click "Reserve".

- **Expected Results:** The slot should be successfully reserved, and a confirmation message should appear.
- **Status:** Pass

Test Case 2: Reservation for Unavailable Slot

- **Test Case ID:** TC-RB-002
- **Description:** Test attempting to reserve a charging slot that is already booked.
- **Preconditions:** The selected slot should already be booked by another user.
- **Test Steps:**

  1. Log into the app.
  2. Search for a charging station with a specific slot.
  3. Try to reserve the slot that is already booked.

- **Expected Results:** The system should display an error message indicating that the slot is unavailable.
- **Status:** Fail

Test Case 3: Cancel Reservation

- **Test Case ID:** TC-RB-003
- **Description:** Test the process of canceling a reservation.

- **Preconditions:** The user must have an active reservation.
- **Test Steps:**

  1. Log into the app.
  2. Go to "My Reservations" and select an active reservation.
  3. Click "Cancel Reservation".

- **Expected Results:** The reservation should be successfully canceled, and the user should receive a cancellation confirmation.
- **Status:** Pass

Test Case 4: Attempt to Reserve Slot without Payment

- **Test Case ID:** TC-RB-004
- **Description:** Test the reservation process when the payment is not completed.
- **Preconditions:** User must have an active reservation.
- **Test Steps:**

  1. Log into the app.
  2. Attempt to reserve a slot but do not proceed with payment.
  3. Click "Submit".

- **Expected Results:** The system should prevent the reservation and prompt the user to complete payment.
- **Status:** Fail

4. Payment Processing

Test Case 1: Successful Payment for Charging Session

- **Test Case ID:** TC-PP-001
- **Description:** Verify successful payment processing for a charging session.
- **Preconditions:** User must have an active reservation.
- **Test Steps:**

1. Log into the app.
2. Go to "My Reservations" and select an active reservation.
3. Click "Pay".
4. Enter valid payment details and confirm payment.

- **Expected Results:** Payment is processed successfully, and a confirmation receipt is generated.
- **Status:** Pass

Test Case 2: Payment Failure Due to Invalid Card Details

- **Test Case ID:** TC-PP-002
- **Description:** Test payment failure due to invalid payment details (e.g., expired card).
- **Preconditions:** User must have an active reservation.
- **Test Steps:**

1. Log into the app.
2. Select an active reservation.
3. Enter expired or invalid card details during the payment process.
4. Click "Submit".

- **Expected Results:** An error message should appear indicating that the payment could not be processed due to invalid card details.
- **Status:** Fail

Test Case 3: Payment Gateway Timeout

- **Test Case ID:** TC-PP-003
- **Description:** Verify system behavior when the payment gateway times out.
- **Preconditions:** User must have an active reservation and valid payment details.
- **Test Steps:**

1. Log into the app.
2. Select an active reservation.

3. Attempt to make a payment during a payment gateway timeout.

- **Expected Results:** The system should alert the user about the payment gateway timeout and allow the user to retry.
- **Status:** Fail

Test Case 4: Successful Payment with Wallet Integration

- **Test Case ID:** TC-PP-004
- **Description:** Verify successful payment using an integrated wallet system (e.g., PayPal, Apple Pay).
- **Preconditions:** User must have a linked wallet account.
- **Test Steps:**

1. Log into the app.
2. Select an active reservation.
3. Choose wallet as payment option.
4. Complete the payment through the wallet.

- **Expected Results:** Payment is successfully processed via the wallet system, and a confirmation message is shown.
- **Status:** Pass

5. Maintenance and Notifications

Test Case 1: Notification for Charging Completion

- **Test Case ID:** TC-MN-001
- **Description:** Verify that the user receives a notification when their charging session is complete.
- **Preconditions:** User must have a charging session in progress.
- **Test Steps:**

1. Log into the EVCMS app.

2. Reserve a charging slot and start the session.
3. Wait for the charging session to complete.
4. User receives a "Charging Complete" notification.

- **Expected Results:** User should receive a notification when the charging session is complete.
- **Status:** Pass

Test Case 2: Maintenance Notification for Station

- **Test Case ID:** TC-MN-002
- **Description:** Test the system's ability to send a notification when a charging station is under maintenance.
- **Preconditions:** Charging station must be undergoing maintenance.
- **Test Steps:**

1. Log into the EVCMS app.
2. Search for a charging station undergoing maintenance.
3. Attempt to reserve a slot at the station.
4. User receives a notification indicating the station is under maintenance.

- **Expected Results:** User should receive a notification that the charging station is under maintenance and cannot be booked.

- **Status:** Pass

6. Mapping Services

Test Case 1: Turn-by-Turn Navigation to Charging Station

- **Test Case ID:** TC-MS-001
- **Description:** Test navigation functionality to a nearby charging station using the mapping service.
- **Preconditions:** User must have location services enabled.

- **Test Steps:**

  1. Log into the EVCMS app.
  2. Search for a nearby charging station.
  3. Click on the "Navigate" button.
  4. Follow the turn-by-turn navigation to the station.

- **Expected Results:** The app should provide accurate turn-by-turn directions to the selected charging station.
- **Status:** Pass

Test Case 2: On-Route Charging Station Navigation

- **Test Case ID:** TC-MS-002
- **Description:** Test the ability of the system to identify charging stations along a travel route.
- **Preconditions:** User must have a destination set in the app.
- **Test Steps:**

  1. Log into the EVCMS app.
  2. Set a destination for travel.
  3. Click on the "On-Route Charging Stations" option.
  4. View a list of charging stations along the route.

- **Expected Results:** The system should identify charging stations along the travel route and display their details.

- **Status:** Pass

7. Battery Management System (BMS) Integration and Energy Management

Test Case 1: Energy Usage Monitoring

- **Test Case ID:** TC-BMS-001

- **Description:** Test the monitoring of energy usage at a charging station through BMS integration.
- **Preconditions:** Charging station must have BMS integration.
- **Test Steps:**

  1. Log into the EVCMS app.
  2. Reserve a charging slot at a station with BMS integration.
  3. Monitor energy consumption during the session.
  4. Receive a notification at the end with detailed energy usage stats.

- **Expected Results:** Energy usage should be accurately tracked and displayed in the user's app.
- **Status:** Pass

Test Case 2: BMS Integration Failure

- **Test Case ID:** TC-BMS-002
- **Description:** Test the system's behavior when the BMS integration fails during a charging session.
- **Preconditions:** Charging station has BMS integration.
- **Test Steps:**

  1. Log into the EVCMS app.
  2. Reserve a charging slot at a station.
  3. Start charging while the BMS integration is disconnected or fails.
  4. Monitor system behavior for energy tracking and reporting.

- **Expected Results:** The system should alert the user that energy tracking is unavailable due to BMS failure.
- **Status:** Fail

34 Testing schedule

The testing schedule provides a timeline for all testing activities, ranging from unit testing to system testing and user acceptance testing (UAT). It includes key milestones, deadlines, and phases to help keep the project on track and ensure timely completion of each stage. By organizing testing in a structured manner, the risk of missing critical defects or delaying the project is minimized. The schedule also provides clear visibility into how testing impacts the broader project timeline.

Week	Focus	Features to Test	Key Deliverables
Week 1	Unit Testing	User Authentication, Charge Station Availability, Booking System, Maintenance and Notifications.	Valid registration, login, and booking processes.
Week 2	Integration Testing	User Authentication + Profile, Charge Station + Booking System, Mapping Services ,Battery Management System (BMS) Integration + Notifications	Integration of systems (e.g., booking and payment) and real-time updates.
Week 3	System Testing	Complete User Workflow, Charge Station Management, Booking and Payment Processing	Full end-to-end functionality, consistent charge station updates.
Week 4	User Acceptance Testing (UAT)	User Experience, Booking Flow, Payment Processing, Notifications	Positive user feedback, intuitive booking and complaint process.

Table 12 – Testing Schedule for EVCMS:

V Project Issues

35 Open Issues

Issues that have been raised and do not yet have a conclusion. A statement of factors that are uncertain and might make significant difference to the product.

- **EV Charger Manufacturer Changes**: If any of the charging stations (hardware) involved in the EVCMS are from third-party suppliers, changes in their API standards or firmware updates could affect integration. For example, if the charger manufacturers change their data protocols, this could impact how the charging station status is communicated.
    - **Resolution:** Will Stay in close contact with hardware suppliers for firmware updates and ensure the software is designed to accommodate changes in API or protocols.
- **Payment Gateway Modifications**: Changes to the payment gateway's APIs (e.g., change in payment processors or shift to new PCI compliance standards) could affect the booking and payment functionality within the system.
    - **Resolution:** In progress, expected to be fixed by next release.

- **Energy Tariff Changes**: If energy pricing laws or policies change (e.g., dynamic pricing, peak-hour rates, subsidies for renewable energy), it could affect how the system charges users for EV charging. If dynamic pricing is introduced, the system will need to adjust accordingly.
    - **Resolution:** In progress, expected to be fixed by next release

- **Scalability Challenges**: As the number of users and charging stations increases, the system may experience performance bottlenecks or latency issues
    - **Resolution**: Will try to Conduct performance testing early in the development cycle and design the system with scalability in mind (e.g., cloud-native architecture)

36 Off-the-Shelf Solutions

36a Ready-Made Products

Ready-made products refer to existing software, tools, or services that can be directly utilized in the development of the "EVCMS" application.

Using ready-made products can significantly reduce the time and effort and allows the development team to focus on customizing the application and improving the user experience instead of building everything from scratch.

Likely Products to Buy or Integrate

- **Charging Station Hardware (EV Chargers)**:
    - **Buy**: Existing EV chargers from manufacturers (e.g., Tesla Superchargers, ChargePoint, Siemens, ABB, etc.) can be purchased and integrated with the system.
- **Payment Gateway Integration**:
    - **Buy**: Existing payment gateway solutions like Razorpay can be integrated for processing user payments for charging sessions.
- **Geospatial Mapping Services**:
    - **Upcoming Products**: New satellite imaging or real-time traffic analysis tools could improve routing algorithms and help users find the most efficient charging stations.

Products or Solutions Not to Be Used

- **Proprietary EV Charging Systems**: Charging stations or infrastructure systems that are highly proprietary or locked to specific vendors (e.g., closed charging ecosystems or hardware that doesn’t comply with OCPP standards) should be avoided.

36b Reusable Components

In EVCMS (Electric Vehicle Charging Management System), various reusable components can be leveraged across multiple modules to improve development efficiency, maintainability, and consistency. This includes integrating modules like authentication systems, notification services, payment processing modules, and data visualization tools, which are readily available and widely used.

The use of pre-built, tested, and reusable components increases system reliability and minimizes the need to build every feature from scratch.

- **User Authentication Module**: A reusable module for user sign-up, login, password recovery, and session management. This can be reused across multiple parts of the application wherever user authentication is required (e.g., for admins or EV owners).
- **Booking System**: A reusable module for managing user reservations, including slot availability checks, booking creation, and calendar management. This can be reused across different modules where booking functionality is needed.
- **Payment Gateway Integration**: A modular component for integrating with various payment gateways (e.g., Razorpay) for processing transactions.
- **Notification Service**: A service for sending system-wide or user-specific notifications (e.g., maintenance reminders, booking updates, charging completion alerts).
- **Scheduler:** A reusable component for scheduling maintenance or updates, which can trigger notifications or actions at specific times.
- **Route Planner**: A module that integrates with mapping services to provide turn-by-turn navigation and route planning. This can be reused for both the charging station locator and the navigation of en route charging points.

36c Products That Can Be Copied

Some products or existing open-source projects can serve as inspiration or even be adapted for the development of the "Two Truths and a Lie Game" application. For example, existing trivia or quiz games with similar functionalities could provide code or design patterns that can be modified to fit this project's requirements. Motivation Using products that can be copied or adapted can offer a head start by providing a foundation that can be customized.

For Legal Implications, Ensure that any copied or adapted code complies with copyright and licensing agreements

- **ChargePoint Application**: These platforms can provide the core logic for user registration, session tracking, payment handling, and scheduling. By adapting these systems, EVCMS can save time on backend development and focus on integrating unique features such as Sensor management and real-time charging station availability.
- **Template UI Designs**: Pre-made UI templates for web and mobile applications can be customized to fit EVCMS's branding and user experience needs. These templates often include standard components like forms, tables, buttons, and maps, which can be easily modified to meet the specific requirements of EV owners, charging stations, and admins. Using these templates ensures a polished and professional design while reducing the effort needed for front-end development.

37 New Problems

37a Effects on the Current Environment

Understanding the potential effects on the current environment helps prepare for any resource scaling or technical adjustments that may be required to accommodate the new application

- If the EVCMS is introduced into an environment where older or legacy charging infrastructure exists, there may be compatibility issues. For

example, older charging stations may not be able to integrate with the new software or IoT-enabled features like predictive scheduling and real-time data monitoring.

- If the mapping feature relies on third-party services (e.g., Google Maps), updates or changes in these services could conflict with the way the EVCMS operates, especially if there are frequent updates to API versions or changes to map features.

- If the system automates tasks that were previously handled by human operators (such as manually booking charging slots, managing payments, or handling customer service issues), there could be a risk of job displacement for employees who performed those tasks.
    - For example, customer service representatives may see fewer calls or interactions if the system provides automated notifications and self-service capabilities.
    - Similarly, operators responsible for maintaining station availability might see their roles reduced if the system automates reservation management, booking, and slot allocation.

- The locator book charging slots, particularly during peak hours, especially when users search for nearby stations in real-time or get directions, can cause increased data traffic. High user engagement (especially during long weekends or holidays) may result in a surge of location-based queries.

- Energy management and monitoring systems will require powerful backend resources to process and analyze large volumes of real-time data from EV batteries and charging stations. This may require more robust data storage, analytics, and processing power to optimize energy usage and grid load.

37b Effects on the Installed Systems

Understanding the effects on installed systems is essential to ensuring that the new EVCMS operates seamlessly within the current infrastructure. It reduces the risk of

compatibility issues, data loss, or workflow disruptions. It is important to analyse how the new application might interact with existing systems to avoid conflicts and ensure smooth operation.

The EVCMS introduces several new, modern features that may conflict with older or legacy systems currently in place. Common conflict areas include:

- **Authentication & Security**: Upgrading authentication protocols and ensuring compliance with modern security standards.
- **Database & Data Integration**: Data schema mismatches and the need for real-time data synchronization.
- More reliable, automated notification systems that keep users informed about charging completion, maintenance, and upcoming appointments.
- **Payment Processing & Notification Systems**: Compatibility issues with outdated payment gateways and manual notification processes.

37c Potential User Problems

Potential user problems refer to anticipated challenges users may face when interacting with the System. This includes issues such as difficulty navigating the system, managing login credentials, adapting for booking Charging Slots, and understanding payment processes.

- Simplify navigation and user interfaces to make the system easy to use, even for those less familiar with digital platforms.
- Offering an interactive tutorial or easy-to-follow instructions for first-time users will help them adapt to the online booking process more quickly.
- Some users may also face technical challenges like poor internet connectivity, device compatibility issues, or a lack of experience with digital platforms.
- Some users, especially older adults or those unfamiliar with mobile apps or websites, may find it challenging to navigate the system or perform tasks like booking, payments, or managing settings.

- Addressing these challenges proactively will improve the overall user experience and promote smoother onboarding, reducing frustration and increasing engagement with the system.

37d Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

limitations in the anticipated implementation environment could hinder the performance or adoption of the System. This could include hardware limitations, network connectivity issues, or platform-specific restrictions. By identifying these limitations, the development team can prioritize compatibility solutions, adjust expectations, and recommend any necessary infrastructure enhancements.

- **Limited or Poor Internet Connectivity**: Users in rural or remote areas with unreliable internet access may struggle to search for nearby charging stations or get real-time updates on station availability (e.g., free or occupied slots, charging status). Slow or intermittent connections can cause delays in retrieving information or disrupt the user experience.

- **Offline Capabilities**: To ensure accessibility even in areas with poor connectivity, the EVCMS should include offline functionality that allows users to access the list of nearby charging stations based on cached data (e.g., saved station locations, previous searches). Users can view station details like location and availability, and once the connection is restored, the data can be updated automatically.
- **Mobile-Optimized Design**: Given that some users may access the system through mobile devices with limited connectivity, the system should prioritize mobile optimization. This includes minimizing data usage by using lightweight interfaces, progressive web app (PWA) features for offline caching.

- **Real-time Notifications** across multiple channels (email, SMS, in-app) to keep users informed.

37e Follow-Up Problems

Follow-up problems may arise after the application is launched, such as software bugs, security vulnerabilities, or new feature requests from users. Preparing for follow-up problems ensures that resources are in place for ongoing maintenance, updates, and user support. Addressing potential follow-up problems proactively also helps to streamline post-launch maintenance and improve overall system performance.

- Provides ongoing support via help desks, FAQs, and tutorials, especially for users unfamiliar with digital tools or new features.
- Schedules regular software updates to fix bugs, patch security vulnerabilities, and enhance features.
- Implemented automatic backups and a disaster recovery plan to safeguard user data against loss.
- Established feedback channels (e.g., surveys) to gather insights for continuous improvement and feature updates.

38 Tasks

38a Project Planning

Project planning involves outlining the project's scope, defining objectives, and creating a timeline for development. It includes identifying key milestones, assigning responsibilities, and establishing a budget for the project. The EVCMS project will adopt an Agile methodology with phases: Requirements Gathering, Design, Development, Testing, Deployment, and Maintenance

- **Timeline Estimates**: Estimate time for each phase based on historical data and testing/deployment needs.
- **User Training and Support**: Plan for resources like manuals, tutorials, and support channels to assist users.
- **Risk Assessment**: Early identification of potential risks (e.g., technical failures, adoption challenges) helps in preparing mitigation strategies.

- **Stakeholder Involvement**: Maintain regular updates and open communication with stakeholders to ensure the system meets both business and user expectations.

38b Planning of the Development Phases

Planning the development phases involves breaking down the project into smaller, manageable stages, such as requirements analysis, design, development, testing, and deployment. Each phase has specific goals and deliverables. Phasing also supports continuous feedback from stakeholders, allowing improvements to be made early in the process.

1. Requirements Gathering

- **Required Operational Date**: Month 1
- **Functional Requirements**: User registration, charging station locator, booking system, payment processing.
- **Nonfunctional Requirements**: System performance, security.

2. Design

- **Required Operational Date**: Month 2
- **Functional Requirements**: UI/UX design for booking and payment, mapping integration.
- **Nonfunctional Requirements**: Responsiveness, user accessibility.

3. Development

- **Required Operational Date**: Month 3-5
- **Functional Requirements**: Charging station management, reservation and booking management, notifications.
- **Nonfunctional Requirements**: Security protocols, data encryption.

4. Testing

- **Required Operational Date**: Month 6
- **Functional Requirements**: Testing of all core functionalities (search, reservation, payment, notifications)
- **Nonfunctional Requirements**: Load testing, performance benchmarks.

5. Deployment and Maintenance

- **Required Operational Date**: Month 7
- **Functional Requirements**: Full system launch (all core modules live) Bug fixes, user support, feature updates.
- **Nonfunctional Requirements**: High availability, fault tolerance, real-time data sync, Continuous system monitoring, data backup.

39 Risks

Risks refer to potential problems or challenges that could arise during the development or operation of the EVCMS System. They can affect the project's scope, schedule, budget, quality, or user experience. Identifying risks early in the project helps in developing strategies to mitigate or avoid them. By managing risks proactively, we can minimize disruptions, maintain project timelines, and ensure successful delivery.

- **Risk Identification**[13]

  This step involves identifying any potential risks or uncertainties that could impact the project.

  - **Security vulnerabilities** in user authentication (e.g., data breaches).
  - **Inaccurate station data** for the Charging Station Locator.
  - **Overbooking** in the Reservation Management module.

- **Risk Assessment**[13]

  Once risks are identified, they need to be assessed to determine their likelihood of occurrence and the potential impact they could have on the project.

  - **Likelihood** of security breach: High (because user data and payment information are involved).
  - **Impact** of overbooking in Reservation Management: High (it could lead to user frustration and loss of trust).

- **Risk Management**[13]

  This is the step where mitigation strategies are developed and implemented to either reduce the likelihood of a risk occurring or minimize its impact if it does occur.

  - **Mitigation for security breaches**: Implement strong encryption, multi-factor authentication, and regular security audits.

- **Inaccurate Metrics**: Misleading data on user registrations and activity may affect user onboarding strategies.

- **Silver Bullet Syndrome**: Overdependence on a single mapping API might cause disruptions if it fails.

- **Creeping User Requirements**: Continuous changes in booking features could delay development and cause scope creep.

- **Inaccurate Cost Estimating**: Underestimating transaction or gateway fees could lead to budget overruns.
- **Management Malpractice**: Failing to adhere to payment security standards could lead to breaches and loss of trust.
- **Cancelled Projects**: Payment system failures or delays in implementation could lead to project cancellation.

- **Low Productivity**: Slow integration of mapping services could delay project timelines.

40 Costs

Costs refer to the financial resources required to develop, deploy, and maintain the "EVCMS" System. These costs can be categorized into various types: Development Costs, Operational Costs, Marketing Costs, Support Costs, etc.

- **Input and Output Flows**:
  - Input: User inputs (search, payment, booking), system data inputs (energy usage, station status).
  - Output: Real-time data (availability, energy stats), notifications, transaction receipts.
- **Business Events**:
  - User actions (searching, booking, paying), system events (maintenance scheduling, session updates).
- **Functional Requirements**:
  - User interaction (login, booking), system management (maintenance, energy monitoring for grid load), payment processing.
- **Nonfunctional Requirements**:
  - Security, usability, scalability, reliability, performance.
- **Constraints**:
  - Legal compliance, third-party service dependencies, hardware limitations.
- **Function Points**:
  - Core functions: User registration, station search, reservation management, payment processing, notifications, map services, displaying sensor data.

41 Waiting Room

This section contains features, enhancements, or ideas that are not included in the current version of the "EVCMS" System but may be considered for future updates. These items are kept in a backlog and can be revisited once the initial project goals are met.This approach ensures that innovative suggestions from users, clients, and stakeholders are not lost and that they are evaluated periodically for feasibility in later versions.

- **Low-Hanging Fruit (High Benefit, Low Cost)**:
  These are requirements or features that are easy to implement but can provide significant value to users or operators. These are usually prioritized for the next release because they offer quick wins without a lot of development resources or time.

  - **Favorite Stations for Rebooking**: Allowing users to easily add frequently used charging stations to a favorites list for quick rebooking. It's a simple feature to implement but offers high convenience for regular users.

  - **Gamification and Rewards System** : Introduce a rewards system where users earn points or discounts for frequent charging, reserving charging slots in advance, or following eco-friendly charging practices (like off-peak charging).
- **Longer-Term, High-Impact Features (Strategic Vision)**:
  These are features that could have a large impact on the system's long-term success but are more complex and costly to implement. These might not make it into the next release but are planned for future versions due to their potential to significantly enhance the platform.

  - **AI-Based Energy Optimization**: Integrating AI or machine learning to predict peak usage times and automatically adjust charging schedules to reduce grid load.

42 Ideas for Solutions

This section discusses potential solutions to address identified challenges in the "EVCMS" System. Solutions may involve technical implementations, design changes, or new features to improve user experience, performance.

These ideas are not part of the current development scope but serve as a brainstorming repository to explore ways to improve the system. The solution ideas here are not commitments but are intended to stimulate creative thinking and strategic planning for future enhancements.

- **Voice Assistant Integration:** Integrate the system with voice assistants (e.g., Alexa, Google Assistant) to allow users to easily find and reserve charging stations, check battery status, or get real-time updates on charging progress. Could be especially useful for users who are driving or unable to access the app directly.
- **AI-Powered Personalized Charging Recommendations :** Use machine learning algorithms to make personalized charging recommendations for users based on their driving habits, historical charging patterns, and preferred locations. This could suggest the optimal times to charge, best stations, or recommend charging options based on energy consumption.[14]

43 Project Retrospective

The goal of conducting a retrospective for the development of the Electric Vehicle Charging Management System (EVCMS) is to evaluate the project's success, identify areas for improvement, and document insights that can enhance the future performance.

1. What Went Well

- The user registration and approval process was smooth, with efficient admin oversight.
- The system provides users with accurate location data, station availability, pricing, and connector types. The mapping integration, especially for en-route charging stations, was a highlight, providing users with real-time navigation updates and ensuring ease of use during long-distance travel.

- The booking feature functioned well, allowing users to reserve charging slots in advance.
- The integration with multiple payment gateways enabled a seamless transaction process, supporting a wide range of payment methods.
- The IoT integration for monitoring battery health and By managing peak usage times and reducing grid load, this feature helped achieve the project's energy management goals.

2. What Could Be Improved

- A more robust system for collecting user feedback during the testing phase could have helped identify minor user experience issues earlier.
- The optimization algorithms for energy management could be enhanced further. While the system effectively minimized grid load, there was room for improvement in predicting usage patterns and dynamically adjusting schedules based on real-time energy demand.

3. Lessons Learned

- The integration of multiple systems (e.g., payment gateways, IoT devices, mapping services) required extensive testing to ensure compatibility. Future projects will benefit from a more comprehensive integration and stress testing phase, especially when dealing with external systems.
- User-Centered Design: User feedback is crucial in the development of intuitive interfaces. Simple features like the ability to favorite stations or easily access booking history were highly appreciated by users. A focus on improving user experience should be prioritized in future iterations.

VI Glossary [15]

EVCMS (Electric Vehicle Charging Management System): A software system designed to manage electric vehicle (EV) charging stations, including functionalities like location search, reservations, scheduling, payments, and predictive maintenance.

IoT (Internet of Things): A network of interconnected devices (sensors, machines, or vehicles) that can collect and exchange data. In this project, IoT sensors are used to monitor and manage charging patterns and optimize energy use.

Grid Load: The amount of electrical power required by all users of an electrical grid at any given time. The EVCMS helps reduce this load by managing how and when EVs are charged.

Agile Methodology: A project management approach that emphasizes iterative development, flexibility, and collaboration. In EVCMS, Agile allows continuous feedback and improvement during each phase of the project.

REST API (Representational State Transfer Application Programming Interface): A set of rules for building and interacting with web services. In this context, the API allows the frontend mobile app to communicate with the backend server for data and services.

Geospatial Mapping Services: These are services that provide location-based data, like maps, satellite imagery, traffic information, and routing algorithms, to help users navigate or find places like charging stations.

Proprietary EV Charging Systems: These are specialized charging systems that are owned by a specific company or vendor. These systems may have restrictions or be locked into their brand or platform, which limits interoperability.

OCPP Standards (Open Charge Point Protocol): A standardized communication protocol for EV chargers that ensures interoperability across different EV charging stations and network providers.

Unit Testing: A type of testing where individual components or functions of the system are tested in isolation to verify that each part works as intended.

Integration Testing: Testing the interactions between different components or modules in the system to ensure that they work together as expected.

System Testing: A complete testing phase where the entire system is tested as a whole to ensure it functions properly and meets the requirements.

User Acceptance Testing (UAT): A type of testing where actual users test the system to ensure it meets their needs and is ready for deployment.

Mock Data: Predefined test data that is used during the testing phase to simulate real user data and system interactions without using actual production data.

API (Application Programming Interface): A set of protocols that allows different software systems to communicate with each other. In the context of EVCMS, APIs may be used for integrating third-party services like Google Maps.

Legacy Systems: Older systems, technologies, or software that are still in use but may not be compatible with new applications or technologies (e.g., older charging stations that do not integrate with the latest EVCMS features).

VII References

[1] https://www.geeksforgeeks.org/software-requirement-specification-srs-format/

[2]https://www.researchgate.net/publication/377746400_Battery_Management_System_using_microcontroller

[3] https://www.betsol.com/blog/7-stages-of-sdlc-how-to-keep-development-teams-running/

[4] https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/

[5] https://en.wikipedia.org/wiki/Electrical_safety_standards

[6] https://www.pcisecuritystandards.org/

[7] https://www.geeksforgeeks.org/goals-and-objectives-of-system-design/

[8]https://polygon.io/knowledge-base/article/what-is-the-difference-between-restful-apis-and-websockets

[9]https://support.ptc.com/help/modeler/r9.1/en/index.html#page/Integrity_Modeler/rtsme/dynamic_modeling_overview.html

[10] https://www.cs.fsu.edu/~myers/cop3331/notes/sysdesign.html

[11] https://www.betabreakers.com/testing-to-pass-vs-testing-to-fail/

[12] https://www.javatpoint.com/test-case

[13] https://en.wikipedia.org/wiki/Risk_management

[14] https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=AI+in+EV+Charging&btnG=

[15] https://www.google.com/

VIII Plagiarism Checker Screenshots

I Project Description

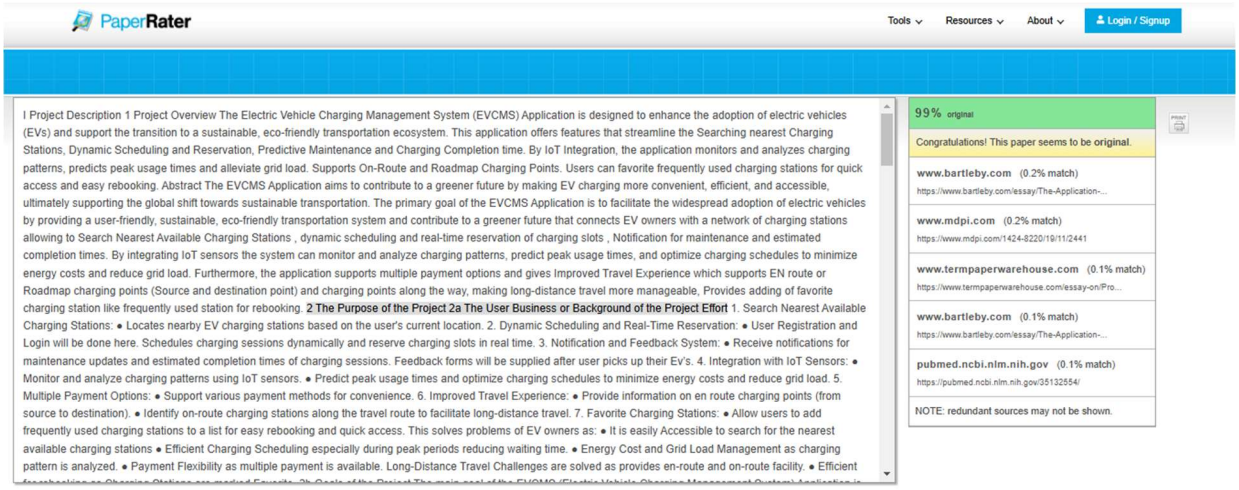


Figure 31 – Plagiarism check for Project Description(1):

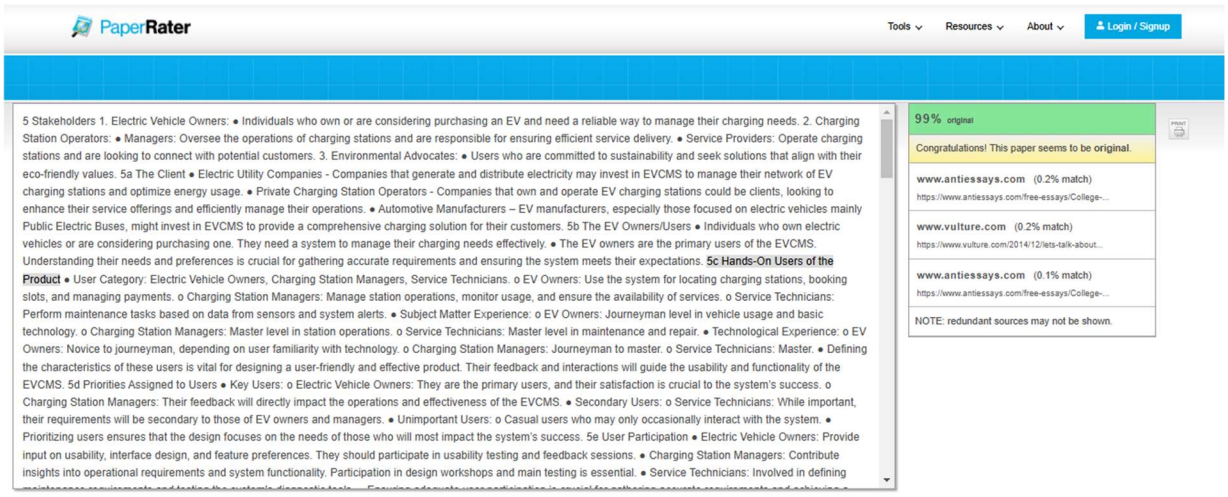


Figure 32 – Plagiarism check for Project Description(2):

II Requirements

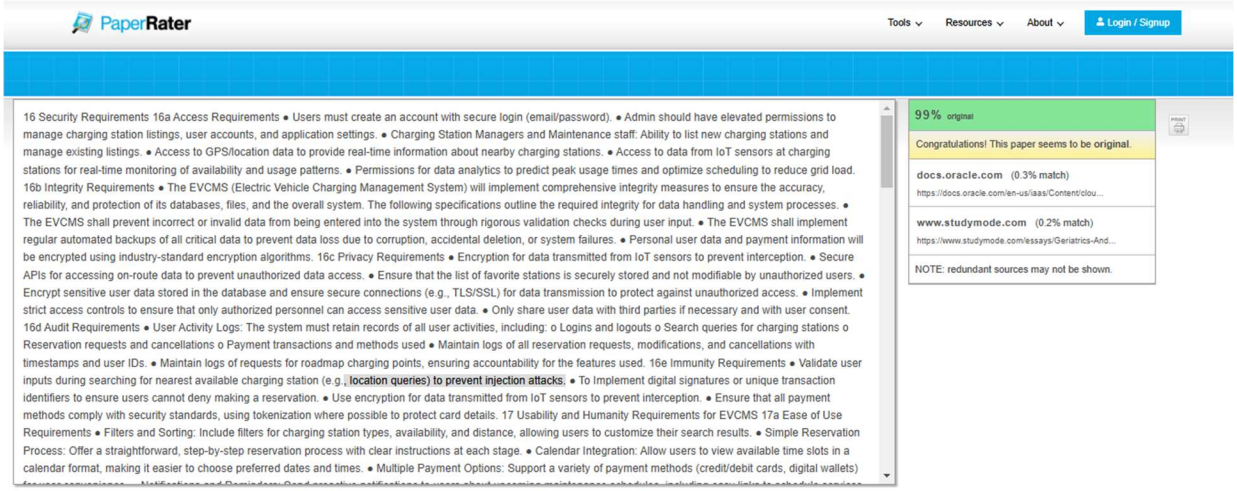


Figure 33 – Plagiarism check for Requirements:

III Diagrams and Design

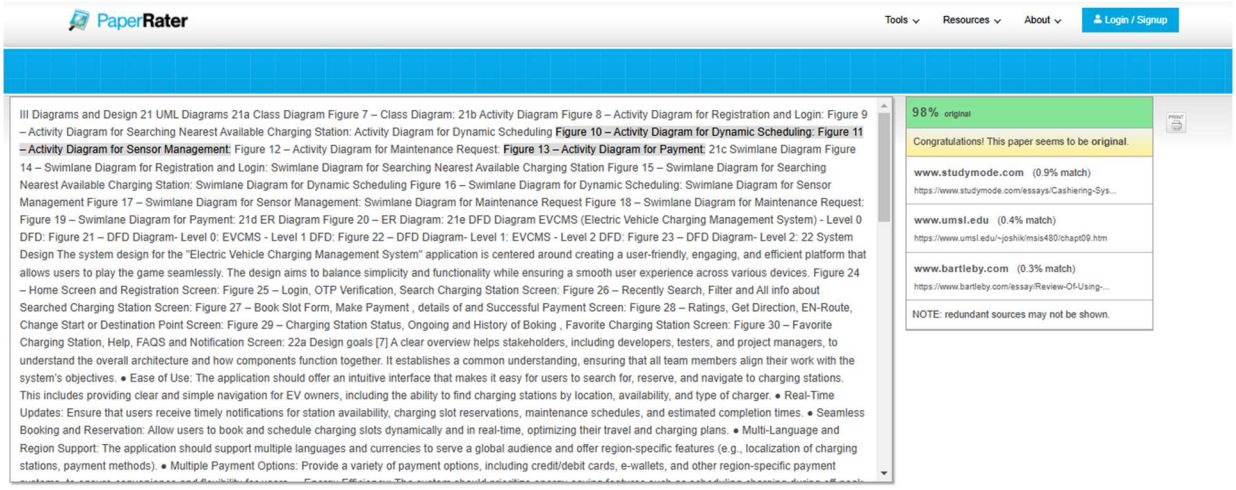


Figure 34 – Plagiarism check for Diagrams and Design(1):

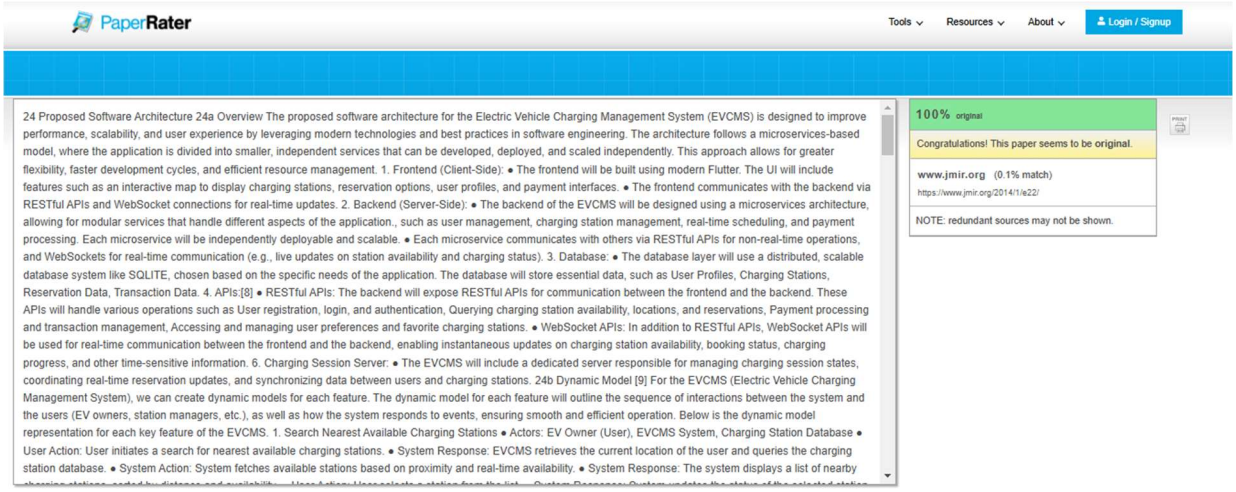


Figure 35 – Plagiarism check for Diagrams and Design(2):

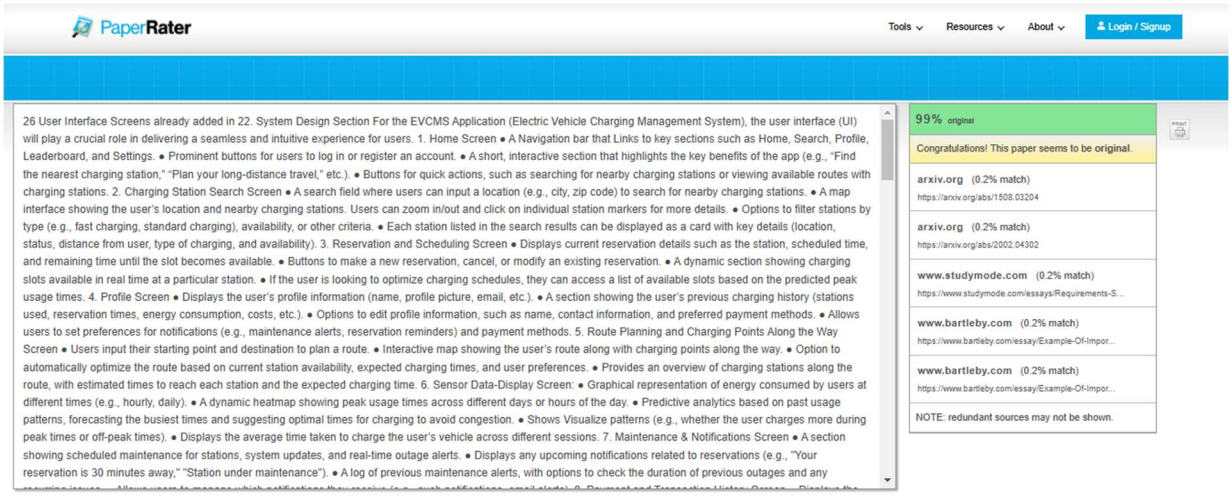


Figure 36 – Plagiarism check for Diagrams and Design(3):

IV Test Plans

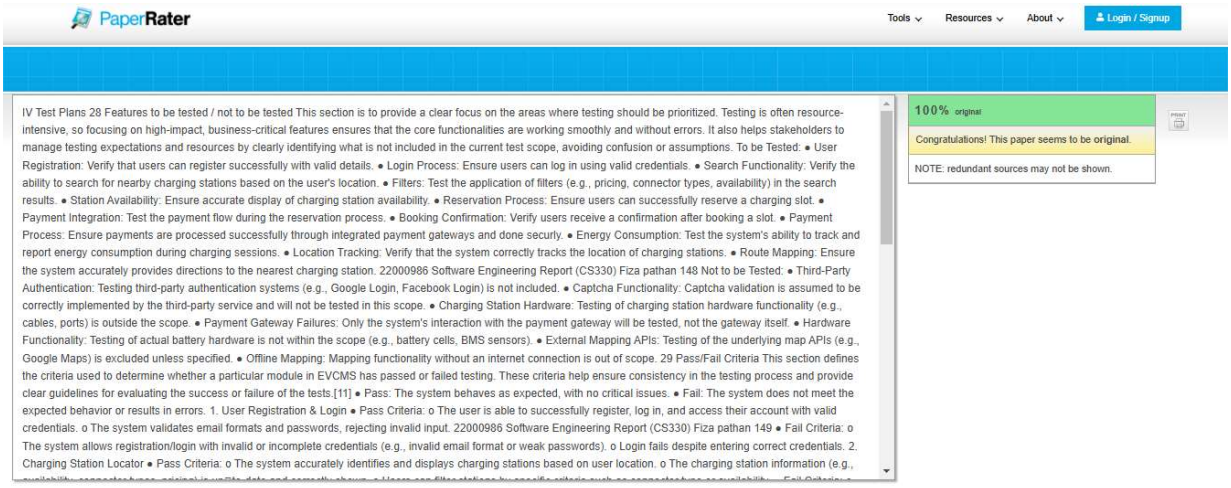


Figure 37 – Plagiarism check for Test Plans(1):

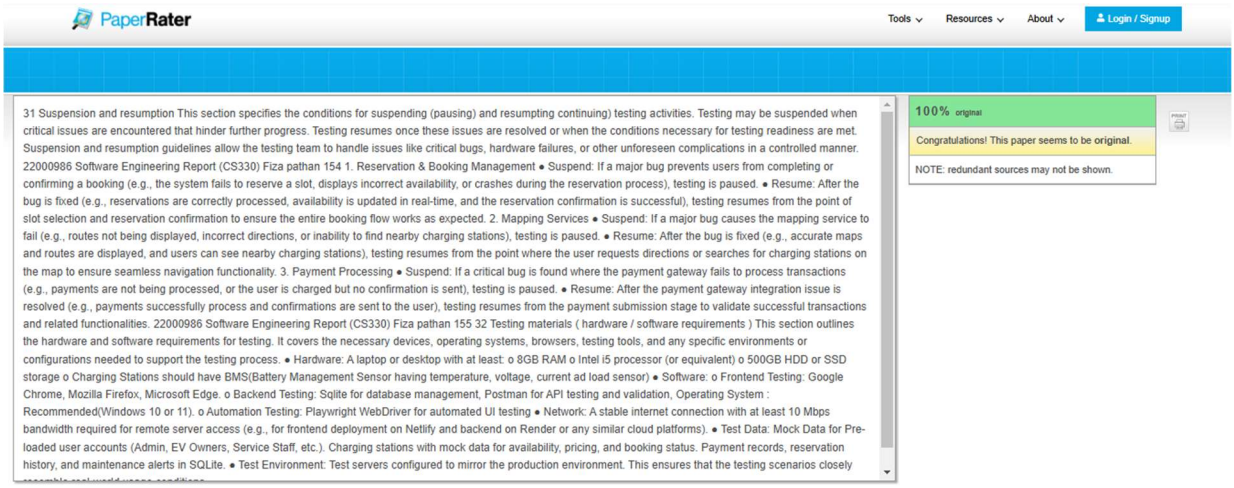


Figure 38 – Plagiarism check for Test Plans(2):

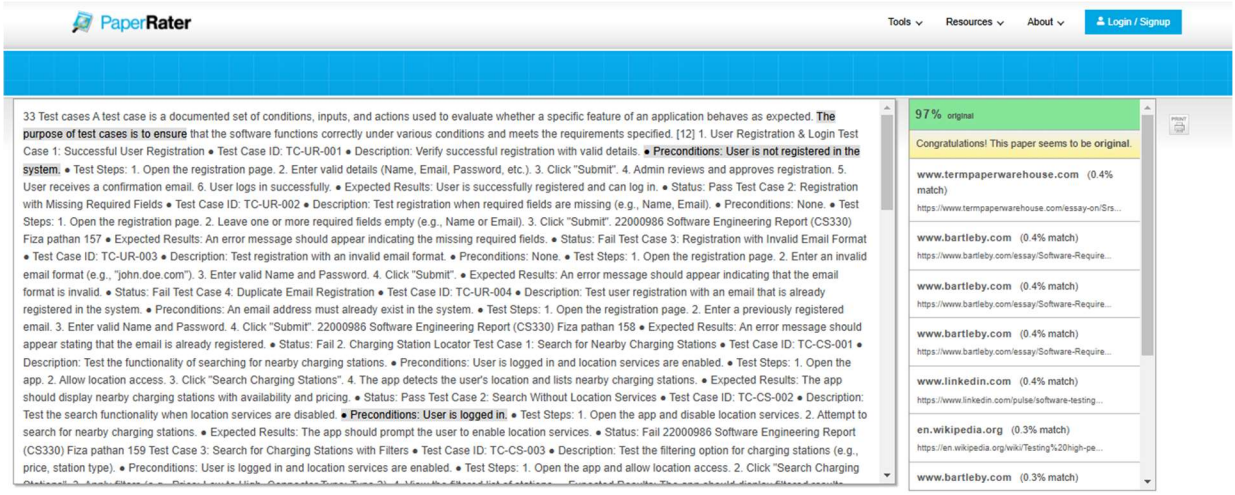


Figure 39 – Plagiarism check for Test Plans(3):

V Project Issues

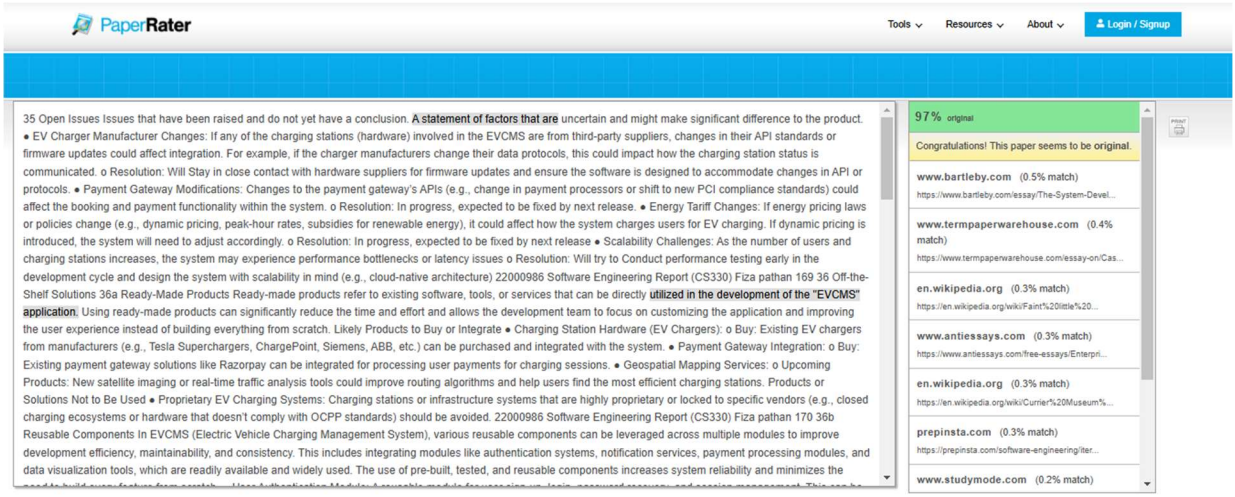


Figure 40 – Plagiarism check for Project Issues(1):

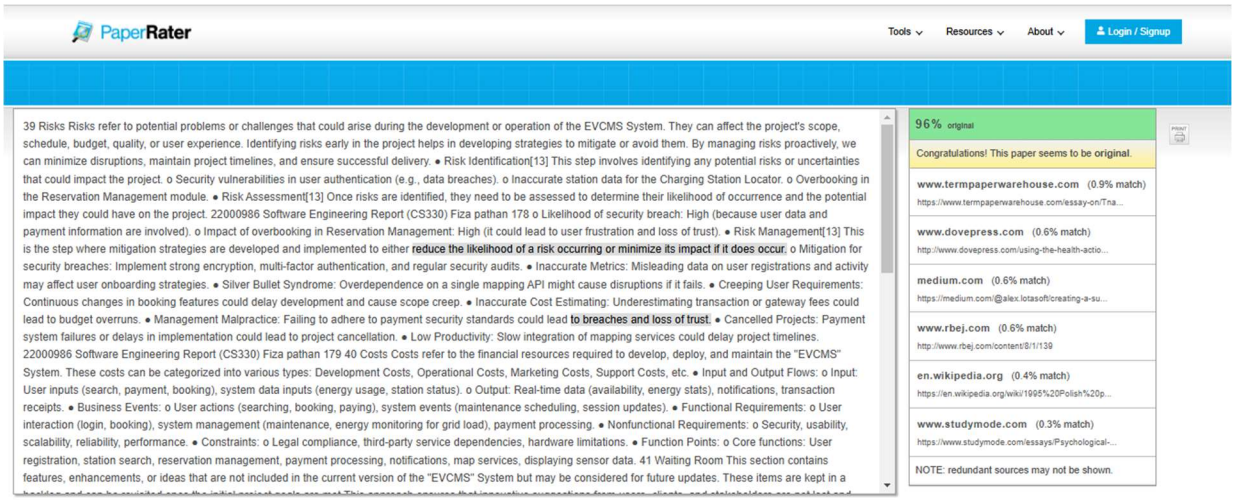


Figure 41 – Plagiarism check for Project Issues(2):

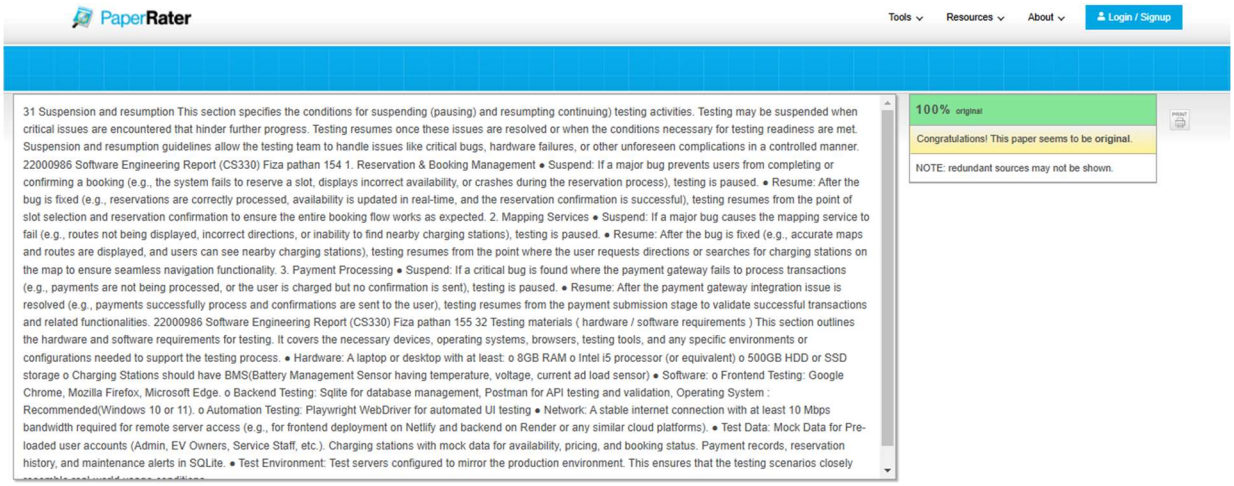


Figure 42 – Plagiarism check for Project Issues(3):