IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, April 2026
Impact Factor: 8.2

A Practical Approach to Electric Vehicle Charging Station Management System Using Object-Oriented Programming Concepts of Java

Sahil Yogesh Londhe[1], Prathamesh Lokhande[2], Gandharv Mamadge[3] and Mrs. Punashri M. Patil[4]
UG Student[1], UG Student[2], UG Student[3], Sr. Assistant Professor[4]
Department of Information Technology
AISSMS Institute of Information Technology, Pune, India
sahilyogeshlondhe@gmail.com

Abstract: *The rapid global transition to electric vehicles (EVs) has increased the demand for efficient and scalable electric vehicle charging station (EVCS) infrastructure. This demand poses management challenges, including optimal station placement, effective scheduling, integration with power grids, and intelligent monitoring systems that support both users and network operations [1][2]. A robust EVCS management system must not only provide reliable access to charging but also address operational scalability, interoperability with communication protocols, and real-time infrastructure monitoring to ensure high utilization and performance.*

Recent studies emphasize that well-planned charging station networks can minimize travel distance and waiting times, improve grid stability, and support smart charging strategies involving bidirectional charging and dynamic energy pricing to enhance sustainability and economic viability.[3] This case study proposes a software-oriented approach to model and implement an EV charging station management system using Object-Oriented Programming concepts in Java. The designed system encapsulates key real-world behaviors such as station registration, vehicle authentication, billing calculation, and resource allocation while illustrating core OOPS principles such as classes, objects, method overloading, constructors, static functionality, and object interaction[1][4. The integration of charging protocols and infrastructure considerations underscores the practical relevance of the study for scalable transportation electrification solutions[5].

Keywords: electric vehicle charging; infrastructure monitoring; smart parking systems; edge computing; Internet of things; predictive analytics; user behavior analysis; energy management systems. [1][3].

I. INTRODUCTION

The global shift toward sustainable transportation has accelerated the adoption of Electric Vehicles (EVs). Governments and industries worldwide are promoting EV usage to reduce carbon emissions, minimize fossil fuel dependency, and combat climate change. However, the rapid growth of EV adoption has created new challenges in charging infrastructure management, energy distribution,An Electric Vehicle Charging Station Management System (EVCSMS) is essential to efficiently manage charging operations, user authentication, energy consumption tracking, billing, and system monitoring. Such systems require modular, scalable, and maintainable software architecture to handle multiple charging stations and vehicles simultaneously billing systems, and operational monitoring[3][5].

Object-Oriented Programming (OOPS) provides a structured approach to model real-world systems using classes and objects. By applying OOPS concepts such as encapsulation, abstraction, method overloading, constructors, and object interaction, complex systems like EV charging networks can be efficiently designed and implemented[1].

DOI: 10.48175/IJARSCT-34029

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, April 2026
Impact Factor: 8.2

This case study focuses on designing and implementing an EV Charging Station Management System using Java to demonstrate key OOPS concepts outlined in Unit-II: Defining Classes and Methods.

1.1 Background of the Study

The global transition toward sustainable and environmentally friendly transportation has significantly accelerated the adoption of Electric Vehicles (EVs) over the past decade. Governments, industries, and environmental agencies are promoting EV usage to reduce greenhouse gas emissions, decrease dependence on fossil fuels, and achieve long-term climate goals. As EV adoption increases, the demand for efficient, reliable, and scalable charging infrastructure has grown rapidly. Electric Vehicle Charging Stations (EVCS) are no longer standalone power outlets; they are intelligent systems integrated with smart grids, digital payment systems, real-time monitoring platforms, and communication networks[3][5]. Managing such infrastructure involves multiple operations, including vehicle authentication, charging slot allocation, energy consumption tracking, billing calculation, and system monitoring. These operations require structured software systems capable of handling multiple interacting components efficiently [1].

Object-Oriented Programming (OOPS) provides an effective approach to designing such complex systems by modeling real-world entities as software objects. Through concepts like encapsulation, abstraction, method overloading, constructors, and object interaction, OOPS enables modular, maintainable, and scalable system development. In the context of EV charging infrastructure, entities such as vehicles, charging stations, batteries, and transactions can be represented as classes and objects, allowing clear separation of responsibilities and controlled data management. Therefore, understanding and implementing an Electric Vehicle Charging Station Management System using OOPS not only addresses a real-world technological need but also strengthens foundational software engineering skills essential for modern system design.

1.2 Problem Statement

The rapid increase in the adoption of Electric Vehicles (EVs) has created significant pressure on existing charging infrastructure. Although many charging stations have been installed, the management of these stations often lacks efficiency, scalability, and proper system integration. Common challenges include inefficient allocation of charging slots, inaccurate billing calculations, limited real-time monitoring of charging status, and difficulty in managing multiple vehicles simultaneously. Additionally, poorly structured software systems can lead to data inconsistency, reduced performance, and difficulty in expanding the infrastructure as demand grows[2].

Most small-scale charging systems operate with minimal automation and lack modular design, making them difficult to maintain and upgrade. Without a well-designed management system, issues such as overloading, billing errors, delayed service, and poor user experience may arise. Therefore, there is a need for a structured, object-oriented approach to design a scalable and maintainable Electric Vehicle Charging Station Management System.

This study addresses the problem of designing a software system that can efficiently manage vehicle charging operations, ensure accurate billing, maintain data security, and support system scalability by applying Object-Oriented Programming principles in Java. The objective is to demonstrate how OOPS concepts can be effectively utilized to model real-world EV charging infrastructure in a systematic and organized manner[1].

II. LITERATURE REVIEW

Elsevier — Motlagh et al. (2025), *"A Review on Electric Vehicle Charging Station Operation Considering Market Dynamics and Grid Interaction"*

Motlagh and colleagues present a comprehensive review of EV charging station operations, focusing on how charging systems interact with electricity markets and power grids. The study evaluates optimization techniques such as predictive models and distributed control for scheduling EV charging under uncertain renewable supply and market pricing. It highlights challenges in dynamic pricing and efficient load management while identifying research gaps related to uncertainty and market design in EV charging infrastructure[2] [4].

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, April 2026
Impact Factor: 8.2

MDPI — Martins & Rodrigues (2025), *"Intelligent Monitoring Systems for Electric Vehicle Charging"*
The growing adoption of electric vehicles (EVs) presents new challenges for managing parking infrastructure, particularly concerning charging station utilization and user behavior patterns. This review examines the current state-of-the-art in intelligent monitoring systems for EV charging stations in parking facilities. We specifically focus on two key inefficiencies: vehicles occupying charging spots beyond the optimal fast-charging range (80% state-of-charge) and remaining connected even after reaching full capacity (100%). We analyze the theoretical and practical foundations of these systems, summarizing existing research on intelligent monitoring architectures and commercial implementations. Building on this analysis, we also propose a novel monitoring framework that integrates Internet of things (IoT) sensors, edge computing, and cloud services to enable real-time monitoring, predictive maintenance, and adaptive control. This framework addresses both the technical aspects of monitoring systems and the behavioral factors influencing charging station management. Based on a comparative analysis and simulation studies, we propose performance benchmarks and outline critical research directions requiring further experimental validation. The proposed architecture aims to offer a scalable, adaptable, and secure solution for optimizing EV charging infrastructure utilization while addressing key research gaps in the field.

III. METHODOLOGY

3.1 Research Approach

The methodology adopted in this study follows an object-oriented system design approach to model and implement an Electric Vehicle Charging Station Management System using Java. The system is developed by analyzing real-world EV charging operations and translating them into software components using Object-Oriented Programming (OOPS) principles. Each real-world entity such as vehicle, charging station, battery, and transaction is represented as a class, and their interactions are implemented through object communication. The methodology emphasizes modular design, data encapsulation, scalability, and structured implementation of Unit-II concepts including constructors, static members, method overloading, recursion, inner classes, and object passing. The development process involves system analysis, class modeling, implementation, and validation to ensure accurate simulation of EV charging operations[1].

3.2 System Analysis

The system analysis phase focuses on understanding the operational requirements and functional components of an Electric Vehicle (EV) Charging Station Management System before implementation. This phase involves studying how real-world EV charging stations operate, identifying user interactions, and determining the data and control mechanisms required for efficient management. The system must support vehicle registration, charging session initiation and termination, monitoring of battery levels, calculation of energy consumption, and generation of billing information. Additionally, it must handle multiple vehicles and charging stations simultaneously while maintaining accurate records of transactions and ensuring data security.

3.3 Object-Oriented System Design

The Object-Oriented System Design translates the EV Charging Station requirements into a structured class-based model using Java. Real-world entities such as EVVehicle, ChargingStation, Battery, and Transaction are represented as individual classes with clearly defined responsibilities. Encapsulation is achieved through private data members and controlled access via public methods. Constructors are used for proper object initialization, while method and constructor overloading provide flexibility in system operations. Static members represent shared attributes such as electricity rates. Object interaction is implemented by passing objects between classes, ensuring modularity, scalability, and maintainability of the system[3][4].

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 14, April 2026
Impact Factor: 8.2

3.4 Detailed Explanation of Methodology

Identification of Real-World Entities

The first step involves identifying real-world components of an EV charging system such as EVVehicle, ChargingStation, Battery, and Transaction. Each entity represents a physical or operational part of the charging infrastructure. These entities are mapped into software classes to accurately model real-world behavior using Object-Oriented Programming principles.

Class Structure Design

After identifying entities, each is converted into a structured class containing attributes and methods. Data members are declared private to ensure encapsulation, while public methods provide controlled access. Responsibilities are clearly defined to maintain modularity and reduce system complexity.

Constructor Implementation

Constructors are implemented to initialize objects when they are created. Default and parameterized constructors are used to allow flexible object creation. Constructor overloading ensures that objects can be initialized with different sets of input values while maintaining valid system states.

Method Design Strategy

Methods are designed to represent system behaviors such as starting charging, calculating energy consumption, and generating bills. Both parameterized and non-parameterized methods are implemented. Method overloading is used to support different charging modes, ensuring flexibility and functional clarity.

Static Members Implementation

Static variables and methods are used to represent attributes common to all objects, such as electricity rate or company policies. This avoids duplication and ensures consistency across the system. Static members support centralized management of shared system parameters.

Object Interaction

Objects interact through method calls and parameter passing. For example, the ChargingStation class accepts an EVVehicle object to initiate charging. This interaction simulates real-world communication between system components and demonstrates the core OOPS concept of object collaboration.

IV. PRACTICAL IMPLEMENTATION

In order to understand OOP concepts its implementation is necessary. Concepts that will be involved are :
Defining a Class.
Access Specifiers.
Introducing Methods.
Method with Parameters and Method without Parameters.
Method with Return Type.
Static Data Members.
Static Methods.
Constructors and Types of Constructors.
Method Overloading.
Using this Keyword.
Using Object as Parameter.
Recursion.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, April 2026 Impact Factor: 8.2

Command Line Arguments.
Inner Classes

4.1 Implementation of Core Classes

EVVehicle Class

This class represents an electric vehicle.

Code :

```
class EVVehicle {
private String vehicleId;
private double batteryLevel;

EVVehicle(String vehicleId, double batteryLevel) {
this.vehicleId = vehicleId;
this.batteryLevel = batteryLevel;
}

void displayVehicle() {
System.out.println("Vehicle ID: " + vehicleId);
System.out.println("Battery Level: " + batteryLevel);
}

void updateBattery(double charge) {
batteryLevel += charge;
}

double getBatteryLevel() {
return batteryLevel;
}
}
```

The implementation ensures encapsulation by using private variables with controlled access through public methods. A parameterized constructor initializes objects with specific data. It also includes both methods that return values and methods that perform actions without returning any value.

ChargingStation Class

This class handles charging operations.

Code:

```
class ChargingStation {
static double electricityRate = 8.5; // Static data
private int stationId;
ChargingStation(int stationId) {
this.stationId = stationId;
}
double chargeVehicle(EVVehicle v, double units) {
v.updateBattery(units);
return units * electricityRate;
}
```

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, April 2026 Impact Factor: 8.2

```
void displayStation() {
System.out.println("Station ID: " + stationId);
}
}
```

The implementation demonstrates the use of a static variable to represent shared data such as the electricity rate across all charging stations. It uses an object as a parameter to allow interaction between classes, such as passing an EVVehicle object to the ChargingStation class for charging operations. The billing amount is calculated using a method with a return type, which returns the computed cost based on energy consumption.

Method Overloading Example

Code :

```
double chargeVehicle(EVVehicle v, double units, double fastChargeRate) {
v.updateBattery(units);
return units * fastChargeRate;
}
```

The implementation uses the same method name with different parameter lists to perform related operations in multiple ways. This allows flexibility in handling different charging scenarios. It demonstrates method overloading, where the method behavior varies based on the number or type of parameters provided.

Inner Class Example

Code :

```
class ChargingStation {
class ChargingPort {
int portNumber;
ChargingPort(int portNumber) {
this.portNumber = portNumber;
}
void displayPort() {
System.out.println("Port: " + portNumber);
}
}
}
```

The ChargingPort class is defined inside the ChargingStation class because a charging port logically belongs to a charging station and cannot exist independently. This structure demonstrates the inner class concept, where one class is nested within another to represent a close relationship between components[2][3].

4.3 Main Class Implementation

Code :

```
public class Main {
public static void main(String args[]) {
EVVehicle v1 = new EVVehicle("EV101", 40);
ChargingStation s1 = new ChargingStation(1);
double bill = s1.chargeVehicle(v1, 20);
v1.displayVehicle();
s1.displayStation();
```

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 14, April 2026 Impact Factor: 8.2

```
System.out.println("Total Bill: " + bill);
System.out.println("Command Line Argument: " + args[0]);
}
}
```

The implementation demonstrates object creation by instantiating classes such as EVVehicle and ChargingStation in the main method. It uses command line arguments to accept input at runtime, making the program dynamic. The interaction between objects, such as passing a vehicle object to the charging station for charging operations, illustrates object collaboration within the system[1][2].

V. CONCLUSION

The Electric Vehicle Charging Station Management System successfully demonstrates the practical application of Object-Oriented Programming concepts in modeling a real-world infrastructure system. By representing core components such as vehicles, charging stations, batteries, and transactions as classes, the system effectively translates real operational processes into structured software design. The implementation highlights essential Unit-II concepts including class definition, object creation, encapsulation, constructors, method overloading, static members, object interaction, recursion, inner classes, and command line arguments.

The modular and object-oriented structure enhances maintainability, scalability, and clarity of the system. Through systematic design and practical simulation, the project not only fulfills academic requirements but also reflects real software engineering practices used in modern EV infrastructure management. Overall, the study demonstrates how OOPS principles provide an efficient and organized approach for developing scalable and reliable management systems.

REFERENCES

[1]. Motlagh, S. G., et al. (2025).*A review on electric vehicle charging station operation considering market dynamics and grid interaction.* This paper reviews smart charging operations, market pricing models, and optimization techniques for EV charging infrastructure.

[2]. Mastoi, M. S., et al. (2022).*An in-depth analysis of electric vehicle charging station infrastructure.* This study analyzes research and developments related to EV charging station infrastructure, challenges, and standardization efforts.

[3]. MDPI (2023).*A Comprehensive Review of Electric Charging Stations with Key Technologies and Protocols.* This review presents key aspects of EVCS technologies, intelligent energy supply systems, and sustainability trends in charging infrastructure.

[4]. Qureshi, K. N. (2021).*Electric Vehicle-Intelligent Energy Management and Charging's Scheduling System.* This paper proposes an integrated EV charging and energy management system with secure communication and scheduling.

[5]. ResearchGate (2025).*Electric vehicle charging stations: Infrastructure review.* This literature review explores future trends including mobile and portable chargers and challenges for next-generation EVCS.