**CHARGING POINT AVAILABILITY FOR ELECTRIC VEHICLES**

**ABSTRACT**

Charging Point Availability for Electric Vehicles is a web-based platform that aims to facilitate the use of electric vehicles by providing information on the availability of charging points in nearby charging stations. Users can enter their area name and see a map of charging stations with the number of available stands. Charging station owners can also use the platform to update the status of their stands and attract more customers. Charging Point Availability for Electric Vehicles is designed to address the challenge of range anxiety and improve the convenience and accessibility of Electric Vehicles charging infrastructure. By leveraging the latest web technologies and data analytics, Charging Point Availability for Electric Vehicles hopes to contribute to the transition towards a sustainable, clean-energy future. It can also help EV drivers plan their trips better, by showing them the number of available stands at each charging station. This can help them avoid waiting in queues or getting stranded due to low battery.

It can provide an incentive for charging station owners to register their stations on your platform, as it can increase their visibility and customer base. This can also help them earn more revenue from charging fees or other services.

It can contribute to the growth and adoption of EVs in India, by improving the accessibility and reliability of charging infrastructure. This can also support the government’s policies and targets for EV promotion and emission reduction12.

It can demonstrate your environmental commitment and social responsibility, by providing a service that helps reduce greenhouse gas emissions and air pollution from transport. This can also enhance your brand value and reputation.

**CHAPTER-1**

**INTRODUCTION**

Electric vehicles (EVs) are becoming more popular and affordable in India, but finding a convenient and reliable charging station can be a challenge for many EV drivers. That is why we have created a web-based platform that helps EV drivers search for nearby available charging stations and shows them the number of available stands at each station. Our platform also helps charging station owners register their stations and increase their visibility and customer base. Our platform aims to improve the accessibility and reliability of EV charging infrastructure in India, and support the growth and adoption of EVs. By using our platform, you can also contribute to reducing greenhouse gas emissions and air pollution from transport, and demonstrate your environmental commitment and social responsibility.

The "Charging Point Availability for Electric Vehicles" project aims to address the growing need for efficient and accessible charging infrastructure for electric vehicles.

* This web-based platform facilitates electric vehicle drivers in locating nearby charging points and provides real-time information on the availability of charging stands.
* Additionally, charging point owners can register their stations on the platform, updating the system with the current number of available stands at their location.

**CHAPTER-2**

**LITERATURE SURVEY**

1. **Title**: Electric Vehicle Charging Guidance Strategy Based on Satisfaction Matching

**Author**: Jiaqi Liao

**Abstract**: The increasing adoption of electric vehicles (EVs) has led to a growing need for efficient and intelligent charging infrastructure. To address the challenges associated with the charging process, this paper proposes a novel Electric Vehicle Charging Guidance Strategy (EVCGS) based on Bilateral Satisfaction Matching (BSM). The primary objective of this research is to enhance user satisfaction by optimizing charging station allocation and scheduling, while also promoting grid stability and minimizing operational costs. By employing a bilateral approach that considers both the EV user’s requirements and the charging station capabilities, the proposed strategy ensures a mutually beneficial charging experience.

1. **Title**: European electric vehicle fleet: driving and charging data analysis

**Author**: C. Corchero

**Abstract**: The electrification of the light vehicle fleets would be a reality in the coming decades. It is being studied through pilot experiences and surveys, how this change in the technology would affect the users driving behaviour. Obviously, new challenges appear, for instance the battery charging, which would be certainly different to the current needs and timings for filling up the internal combustion engine vehicles tanks. Analysis of real electric vehicle usage data is a key point in the development of the electric mobility. A large collection of electric vehicles and charging points have been monitored during three years and the results about the driving and charging patterns are shown in this work. These results may help to develop future charging infrastructure location and to evaluate the electric vehicle integration into the grid.

1. **Title:** Short-term forecast of electric vehicle charging stations occupancy using big data streaming analysis

**Author**: Francesca Soldan

**Abstract**: The widespread diffusion of electric mobility requires a contextual expansion of the charging infrastructure. An extended collection and processing of information regarding charging of electric vehicles may turn each electric vehicle charging station into a valuable source of streaming data. Charging Point Operators may profit from all these data for optimizing their operation and planning activities. In such a scenario, big data and machine learning techniques would allow valorizing real-time data coming from electric vehicle charging stations. This paper presents an architecture able to deal with data streams from a charging infrastructure, with the final aim to forecast electric charging station availability after a set amount of minutes from present time. Both batch data regarding past charges and real-time data streams are used to train a streaming logistic regression model, to take into account recurrent past situations and unexpected actual events. The streaming model performs better than a model trained only using historical data. The results highlight the importance of constantly updating the predictive model parameters in order to adapt to changing conditions and always provide accurate forecasts.

**4.Title**: Control of Electric Vehicles Charging Without Communication Infrastructure

**Author**: Morsy Nour

**Abstract**: This paper presents a study of controlled charging of electric vehicles (EVs) using a fuzzy logic controller. The controller regulates the charging power according to the state of charge (SoC) of EV and the voltage at the point of connection of EV charger. The objective of the controller is to charge EVs while keeping the voltage of different distribution system points within the acceptable limits. Most of the proposed methods for controlled charging of EVs depend on the availability of the communication infrastructure which enables communication between utility operator and EV chargers which is not available in the current distribution networks and requires a huge investment. So, an autonomous controller which does not need communication infrastructure is developed in this study. The controller is designed to charge EVs without violating the voltage limits which can occur in case of uncontrolled charging of EVs. The effectiveness of the controller is tested on a residential low voltage (LV) distribution network and the simulations are executed with MATLAB/SIMULINK. The network performance in case of controlled charging of EVs is compared with the uncontrolled charging of EVs and the base case when no EVs are connected. The results demonstrate the distinction of the proposed controlled charging method in terms of total power demand, transformer loading, cable loading, and voltage profile over uncontrolled charging.

Top of Form

**CHAPTER-3**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

Manual checking of EV charging point availability is a system where EV drivers have to rely on their own knowledge, experience, or guesswork to find a suitable charging station for their vehicles. This system has several drawbacks, such as:

It can be time-consuming and frustrating for EV drivers to search for charging stations, especially in unfamiliar areas or during peak hours.

It can increase the risk of running out of charge or getting stranded due to low battery, as EV drivers may not be able to find a charging station when they need it.

It can create inefficiencies and imbalances in the utilization of charging infrastructure, as some stations may be overcrowded while others may be underused.

It can discourage potential EV buyers from switching to EVs, as they may perceive charging as a hassle or a barrier.

Manual checking of EV charging point availability is an outdated and unreliable system that does not meet the needs and expectations of EV drivers and charging station owners. Your web-based platform can offer a better alternative by providing accurate and real-time information on charging station locations and availability.

**DISADVANTAGES**

**1)Lack of Real - Time Information:**

* Manual checking often relies on periodic updates or static information, leading to a lack of real-time data on charging point availability. This can result in outdated and inaccurate information for electric car users.

**2)Inconvenience for Drivers:**

* Electric car drivers may need to physically visit or contact each charging station to inquire about availability, leading to inconvenience and wasted time. This approach is impractical, especially in areas with numerous charging points.

**3)Limited Accessibility:**

* Without a centralized platform, information about charging point availability may be scattered or not readily available to the public. This lack of accessibility can discourage potential electric car users from relying on electric vehicles for their transportation needs.

**3.2 PROPOSED SYSTEM**

Certainly! The web-based platform "Charging Point Availability for Electric Vehicles" offers several advantages for both electric vehicle drivers and charging point owners. Here are the key benefits of the platform.

**ADVANTAGES:**

1. **Real-Time Availability Information:**

* The platform provides real-time information on the availability of charging points, allowing electric car drivers to make informed decisions about where to charge their vehicles.

1. **Convenience for Electric Car Drivers:**

* Electric car drivers can easily locate nearby charging points and check their availability without the need for manual visits or calls to each charging station, enhancing convenience and saving time.

**3) Optimized Charging Experience:**

* Users can plan their charging sessions in advance by checking the availability of charging points, reducing the likelihood of delays and ensuring a more streamlined and efficient charging experience.

1. **Business Opportunities for Charging Point Owners:**

* Charging point owners can attract more users to their stations through the platform, increasing visibility and potentially attracting new customers. The platform serves as a promotional tool for charging infrastructure businesses.

**CHAPTER-4**

**SYSTEM REQUIRMENTS**

**HARDWARE REQUIREMENTS:**

* PROCESSOR : I3.
* Hard Disk : 40 GB.
* Ram : 2 GB.

**SOFTWARE REQUIREMENTS:**

* Operating system : Windows.
* Coding Language : JAVA/J2EE
* Data Base : MYSQL
* IDE` : Netbeans8.1

**CHAPTER-5**

**SYSTEM STUDY**

**5.1 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**5.2 FEASIBILITY ANALYSIS**

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

**TECHNICAL FEASIBILITY**

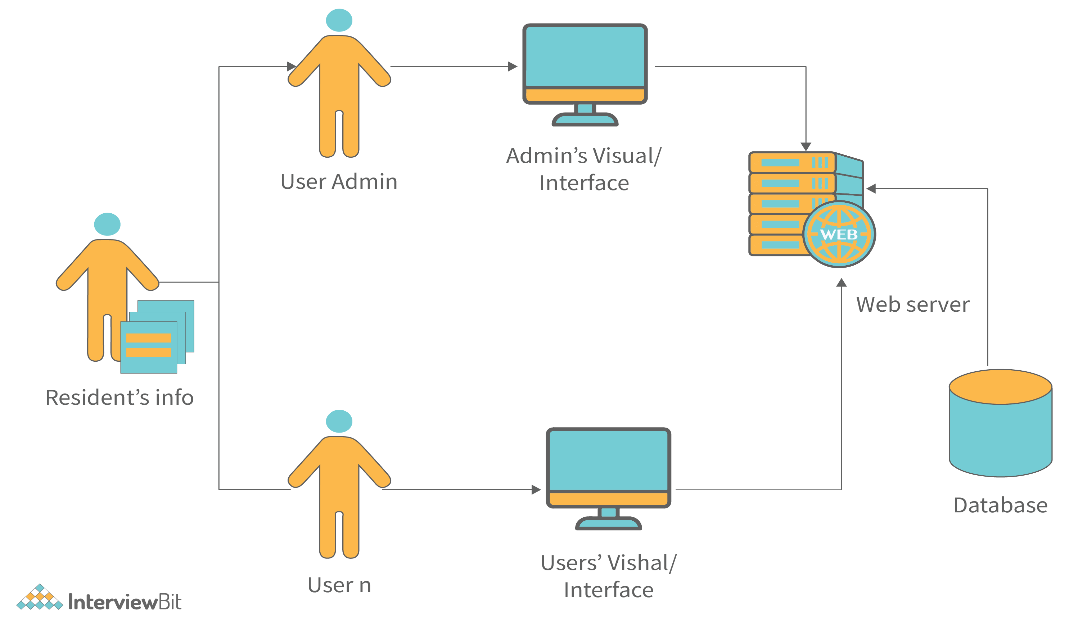
This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

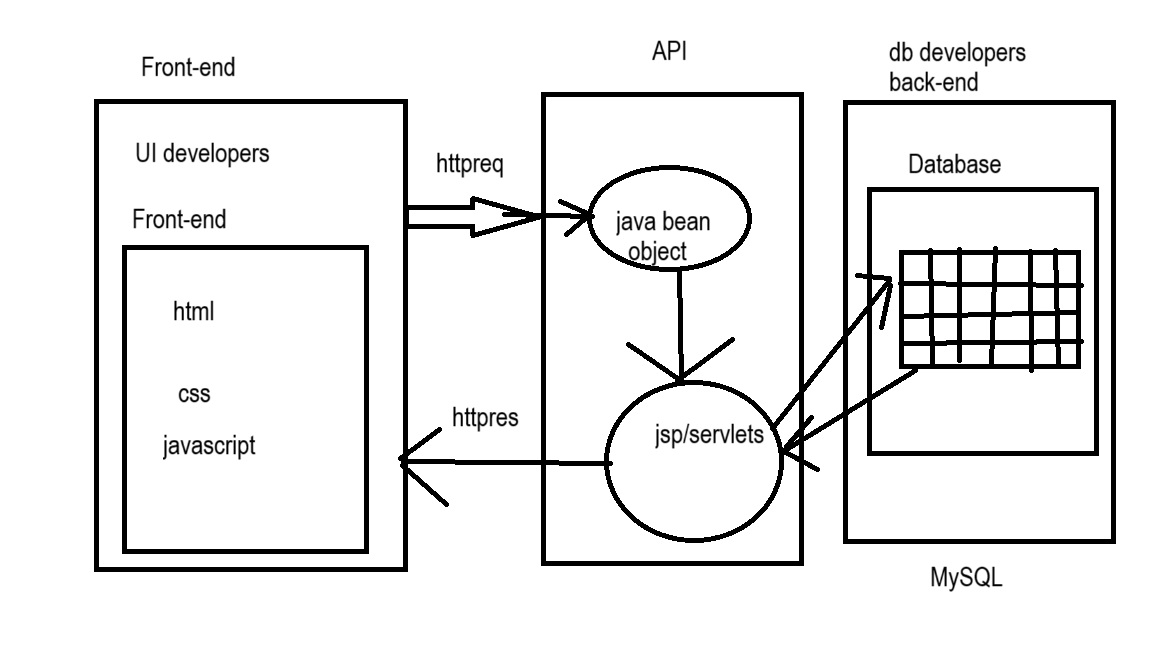
**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**CHAPTER-6**

**SYSTEM DESIGN**

**6.1 SYSTEM ARCHITECTURE:**

****

**6.2 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

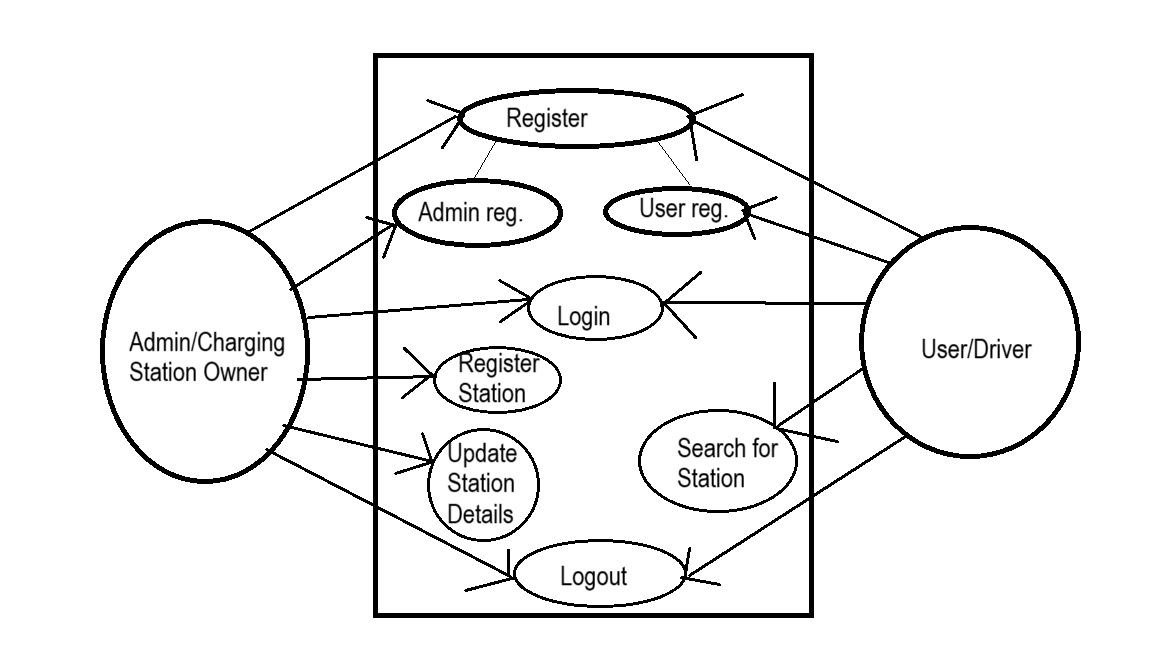
**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

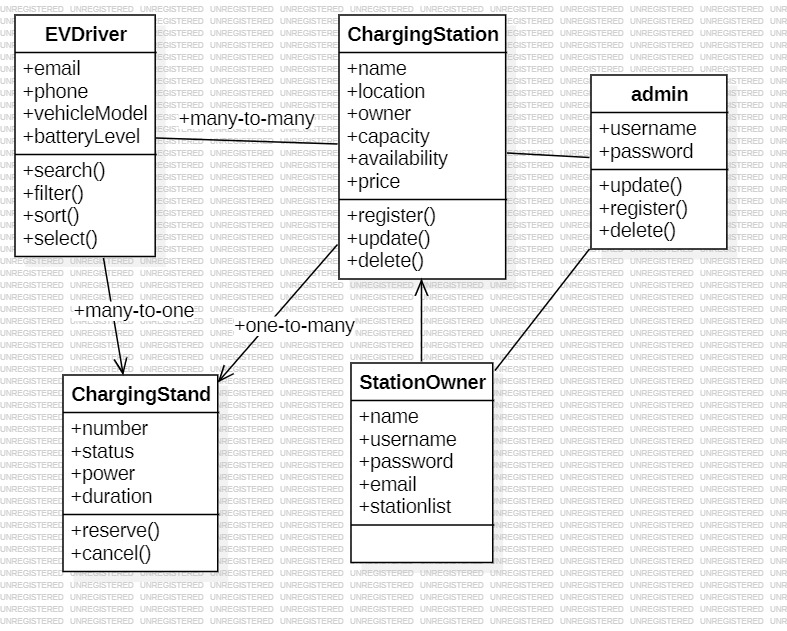
**6.2.1 USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



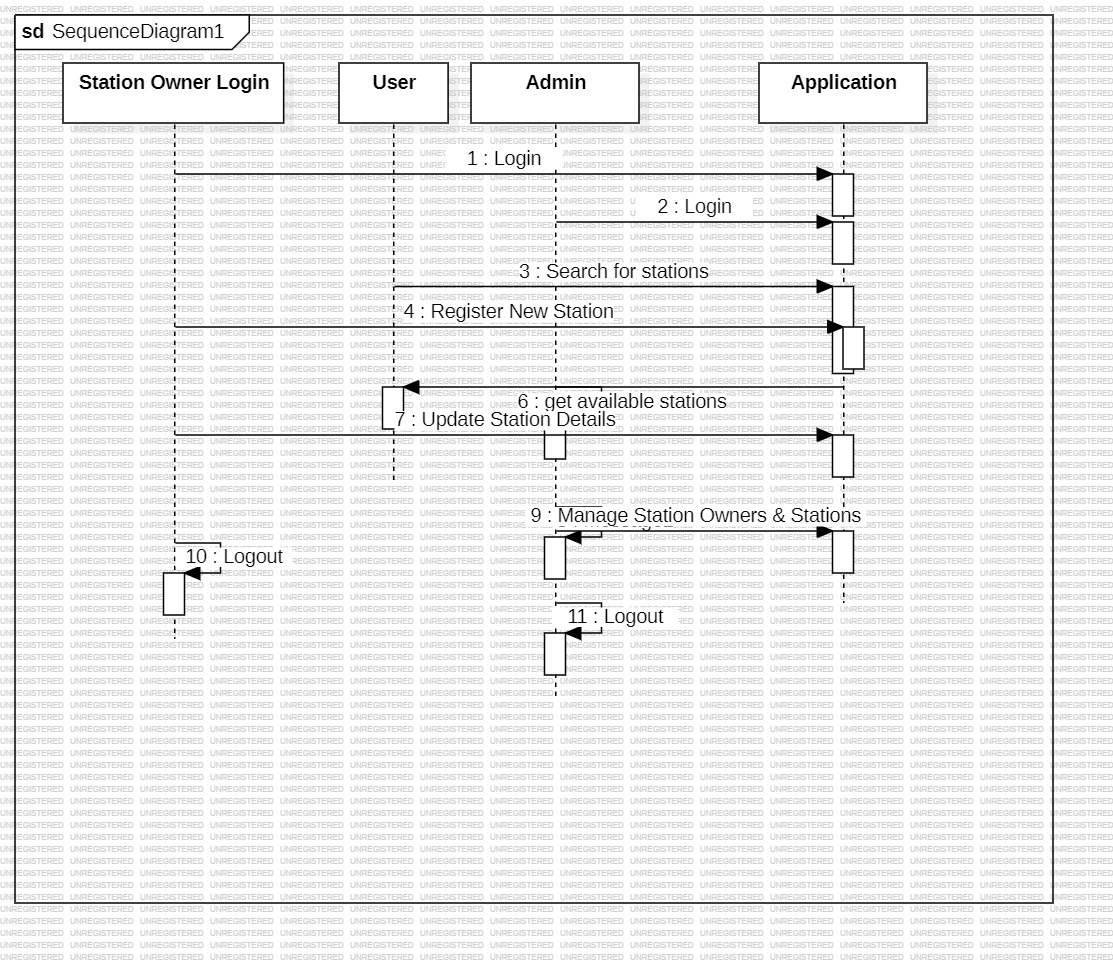
**6.2.2 CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information. Class diagrams are a fundamental tool in object-oriented modeling and are widely used in software development.



**6.2.3 SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



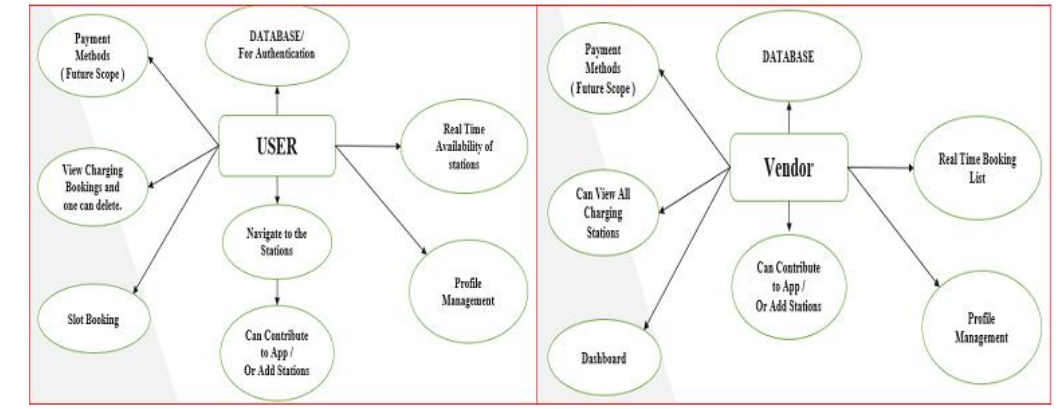
**6.2.4 COLLABARATION DIAGRAM :**

A Collabaration diagram, also known as a Communication diagram, is a type of Unified Modeling Language (UML) diagram used to visually represent the interactions and communication between objects or components in a system or software application. It focuses on showing how various objects or components collaborate to achieve specific functionality or a particular use case. Collaboration diagrams are particularly useful for understanding the dynamic aspects of a system and how objects interact over time.

****

**6.2.5 ACTIVITY DIAGRAM :**

An Activity diagram is a type of Unified Modeling Language (UML) diagram that is used to model the dynamic aspects of a system, particularly the flow of activities and actions within a business process, use case, or other system behaviors. Activity diagrams provide a graphicalrepresentation of how different activities and actions are organized and interact to achieve a specific goal.Activity diagrams help stakeholders, including analysts, designers, and developers, to understand the sequential and parallel flows of activities and make them a valuable tool for system design and documentation.

****

**CHAPTER-7**

**INPUT AND OUTPUT DESIGN**

**7.1 INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**7.1.1 OBJECTIVES**

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

**7.2 OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

**CHAPTER-8**

**IMPLEMENTATION**

A module is a self-contained unit of code or functionality that can be reused or combined with other modules to create a larger system. Modules can help you organize your code, improve readability, and facilitate testing and maintenance.

**8.1 MODULES**

* USER/DRIVER INTERFACE MODULE
* ADMIN/STATION OWNER INTERFACE MODULE
* WEB SERVICE MODULE
* DATABASE MODULE
  + 1. **MODULE DESCRIPTION**

**User Interface Module:**

A user interface module that handles the front-end web development, such as displaying the web page layout, design, and content. This module can use HTML, CSS, and JavaScript to create a responsive and user-friendly website for EV drivers and charging station owners.

**Web Service Module:**

A web service module that handles the back-end web development, such as connecting to the server, database, and other web services. This module can use Java, Spring Boot, and JSON to create a web service that generates and consumes data from the relational database and other sources.

**Database Module:**

A database module that handles the data storage and retrieval, such as storing and querying the information about the charging stations, their locations, availability, and other details. This module can use MySQL or another relational database management system to create and manage the database tables and schemas.

**CHAPTER-9**

**SOFTWARE ENVIRONMENT**

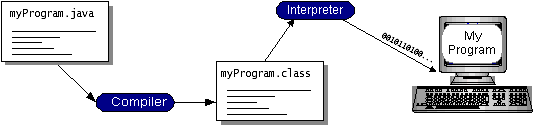
**9.1 JAVA TECHNOLOGY**

Java technology is both a programming language and a platform. The Java Programming Language.

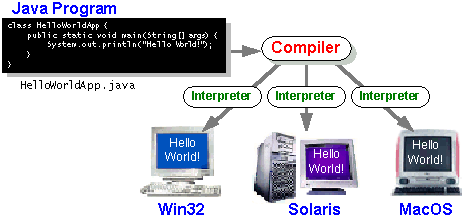
**The Java programming language is a high-level language that can be characterized by all of the following buzzwords:**

* + - Simple
    - Architecture neutral
    - Object oriented
    - Portable
    - Distributed
    - High performance
    - Interpreted
    - Multithreaded
    - Robust
    - Dynamic
    - Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



**The Java Platform**

A *platform* is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other hardware-based platforms.

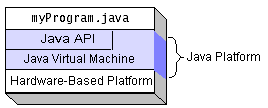
**The Java platform has two components:**

* The *Java Virtual Machine* (Java VM)
* The *Java Application Programming Interface* (Java API)

You’ve already been introduced to the Java VM. It’s the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

**What Can Java Technology Do?**

The most common types of programs written in the Java programming language are *applets* and *applications*. If you’ve surfed the Web, you’re probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

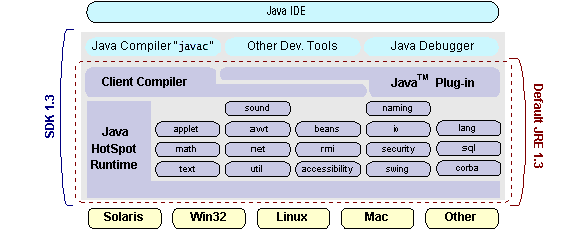
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

* **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
* **Applets**: The set of conventions used by applets.
* **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
* **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
* **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
* **Software components**: Known as JavaBeansTM, can plug into existing component architectures.
* **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
* **Java Database Connectivity (JDBCTM)**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



**How Will Java Technology Change My Life?**

We can’t promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

* **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it’s easy to learn, especially for programmers already familiar with C or C++.
* **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
* **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people’s tested code and introduce fewer bugs.
* **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
* **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure JavaTM Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
* **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
* **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

**ODBC**

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn’t change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn’t as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

**JDBC**

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

**JDBC Goals**

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. ***SQL Level API***

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

1. ***SQL Conformance***

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

1. ***JDBC must be implemental on top of common database interfaces***The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.
2. ***Provide a Java interface that is consistent with the rest of the Java system***

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

1. ***Keep it simple***

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

1. ***Use strong, static typing wherever possible***

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

1. ***Keep the common cases simple***

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally, we decided to proceed the implementation using Java Networking. And for dynamically updating the cache table we go for MS Access database.

Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple Architecture-neutral

Object-oriented Portable

Distributed High-performance

Interpreted multithreaded

Robust Dynamic

Secure

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.

**Java Program**

**Compilers**

**Interpreter**

**My Program**

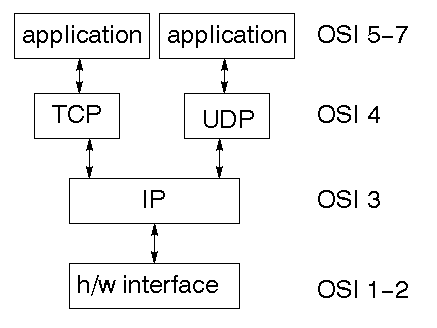
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

**Networking**

**TCP/IP stack**

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

**IP datagram’s**

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

**UDP**

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

**TCP**

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

**Internet addresses**

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

**Network address**

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

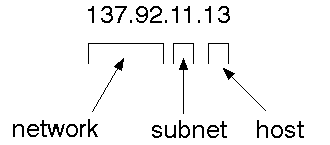
**Subnet address**

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

**Host address**

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

**Total address**



The 32 bit address is usually written as 4 integers separated by dots.

**Port addresses**

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

**Sockets**

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

#include <sys/types.h>

#include <sys/socket.h>

int socket(int family, int type, int protocol);

Here "family" will be AF\_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

**JFree Chart**

JFree Chart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, [free software](http://www.gnu.org/philosophy/free-sw.html). It is distributed under the terms of the [GNU Lesser General Public Licence](http://www.gnu.org/licenses/lgpl.html) (LGPL), which permits use in proprietary applications.

**1. Map Visualizations**

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

**2. Time Series Chart Interactivity**

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

**3. Dashboards**

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

**4. Property Editors**

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

**J2ME (Java 2 Micro edition) :-**

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

**1. General J2ME architecture**



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

**2.Developing J2ME applications**

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

**3.Design considerations for small devices**

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

\* Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.

\* Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.

\* Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

**4.Configurations overview**

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

\* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

\* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

**5.J2ME profiles**

**What is a J2ME profile?**

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

**Profile 1: KJava**

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com.sun.kjava. We'll learn more about the KJava API later in this tutorial when we develop some sample applications.

**Profile 2: MIDP**

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

\* java.lang

\* java.io

\* java.util

\* javax.microedition.io

\* javax.microedition.lcdui

\* javax.microedition.midlet

\* javax.microedition.rms

**9.2 SOURCE CODE: -**

**index.html**

<html>

<head>

<title>Charging Point Availability For Electric Vehicle!</title>

<style>

body {

background:linear-gradient(rgba(255,255,255,1),rgba(123,239,178,1))

}

#header {

height: 100px;

width: 100%;

background: green;

}

#links {

height: 50px;

width: 100%;

background-color: greenyellow;

}

ul li {

float: left;

width: 100px;

list-style: none;

}

a {

text-decoration: none;

font-size: 20px;

color: crimson;

}

#body {

height: 600px;

width: 100%;

color: blue;

}

#box{

height: 180px;

width: 300px;

border: 3px solid black;

border-radius: 15px;

}

img{

margin-top: 40px;

}

p{

color:blue;

}

#footer {

height: 100px;

width: 100%;

}

#search box{

display: flex;

justify-content: center;

margin: 500px 0;

}

#search input{

width: 80%;

padding: 10px;

border: 1px solid black;

border-radius: 5px;

}

#search button{

background-color: green;

color: white;

padding: 10px 20px;

border: none;

border-radius: 5px;

}

</style>

</head>

<body>

<div id="header">

<h1 style="color: white; text-align: center;">Charging Point Availability For Electric Vehicle</h1>

<div id="links">

<ul>

<li><a href="index.html">Home</a></li>

<li><a href="account.html">User</a></li>

<li><a href="admin.html">Admin</a></li>

</ul>

</div>

<div style="width:30%; text-align:left; color: black">

<h2>Find charging point near you </h2>

</div>

<div style="width:30%; text-align:left; color: black">

<h5>Enter your area name and see charging point avability </h5>

</div>

<div id="search box">

<input type="text" id="search-input" placeholder="Location">

<button id ="search button"> Search</button>

</div>

<div id="body">

<img src="electric.jpeg" height="300" width="600" alt="Project Image" >

<p>About Us:Charging Point Availability for Electric Vehicles is a web-based platform that aims to facilitate the use of electric vehicles by providing information on the availability of charging points in nearby charging stations. Users can enter their area name and see a map of charging stations with the number of available stands. Charging station owners can also use the platform to update the status of their stands and attract more customers. Charging Point Availability for Electric Vehicles is designed to address the challenge of range anxiety and improve the convenience and accessibility of Electric Vehicles charging infrastructure. By leveraging the latest web technologies and data analytics, Charging Point Availability for Electric Vehicles hopes to contribute to the transition towards a sustainable, clean-energy future. It can also help EV drivers plan their trips better, by showing them the number of available stands at each charging station. This can help them avoid waiting in queues or getting stranded due to low battery.

It can provide an incentive for charging station owners to register their stations on your platform, as it can increase their visibility and customer base. This can also help them earn more revenue from charging fees or other services.

It can contribute to the growth and adoption of EVs in India, by improving the accessibility and reliability of charging infrastructure. This can also support the government’s policies and targets for EV promotion and emission reduction12.

It can demonstrate your environmental commitment and social responsibility, by providing a service that helps reduce greenhouse gas emissions and air pollution from transport. This can also enhance your brand value and reputation.

</p>

</div>

<div id="footer">

<h3 style="color: blue; text-align: center;">&copy; All rights reserved Mini project 2024</h3>

<a href="https://www.facebook.com/"><img src="face.png" alt="Facebook" height="30" width="40"></a>

<a href="https://www.instagram.com/codewithharry"><img src="insta.png" alt="Instagram" height="30" width="40"></a>

<a href="https://www.twitter.com/"><img src="twitter.png" alt="Instagram" height="30" width="40"></a>

</div>

</body>

</html>

**Customer Login:**

**<html>**

**<head>**

**<title> user login</title>**

**<style>**

**body{**

**background-image: url("back.jpeg");**

**background-repeat: no-repeat;**

**background-size: cover;**

**}**

**#header{**

**height: 100px;**

**width: 100%;**

**background: rgba(255,255,255,0.2);**

**display: flex;**

**justify-content: space-between;**

**}**

**#logo{**

**display: flex;**

**}**

**h1{**

**margin-left: 80px;**

**color: white;**

**}**

**ul li {**

**display: inline-block;**

**width: 70px;**

**}**

**a{**

**color: white;**

**text-decoration: none;**

**font-size: 20px;**

**}**

**a:hover{**

**color:blue;**

**background-color: white;**

**border-radius: 10px;**

**}**

**#body{**

**height: 500px;**

**width: 100%;**

**}**

**#box{**

**height: 180px;**

**width: 300px;**

**border: 3px solid white;**

**margin-top: 50px;**

**border-radius: 15px;**

**}**

**th{**

**color: white;**

**}**

**#footer{**

**height: 50px;**

**width: 100%;**

**background: rgba(255,255,255,0.2);**

**}**

**input[type=text]{**

**border-radius: 10px;**

**}**

**input[type=password]{**

**border-radius: 10px;**

**}**

**input[type=submit]{**

**border-radius: 10px;**

**}**

**</style>**

**</head>**

**<body>**

**<div id="header">**

**<div id ="logo">**

**<img src="logo.jpeg" height="90" width="150">**

**<h1>Charging Point Availability For Electric Vehicle</h1>**

**</div>**

**<div id ="links">**

**<ul>**

**<li><a href="index.html">Home</a></li>**

**<li><a href="account.html">User</a></li>**

**<li><a href="admin.html">Admin</a></li>**

**</ul>**

**</div>**

**</div>**

**<div id="body">**

**<center>**

**<div style="width:30%; text-align: center; color: white">**

**<h2>Welcome back to our community </h2>**

**</div>**

**<div style="width:30%; text-align: center; color: white">**

**<h2>Customer Login</h2>**

**</div>**

**<div id="box">**

**<table>**

**<h2 style="color :white"; User login></h2>**

**<tr>**

**<th> User name:</th>**

**<td><input type ="text" name ="user name" placeholder="enter user name" required></td>**

**</tr>**

**<tr>**

**<th> password:</th>**

**<td><input type ="password" name ="password" placeholder="enter password" required></td>**

**</tr>**

**<tr>**

**<th> </th>**

**<td> <input type ="submit" value="SignIn">**

**</td>**

**</tr>**

**<tr>**

**<th> </th>**

**<td style="color: white;"> Create new account? <a href ="accountsignup.html"> Sign Up</a></td>**

**</tr>**

**</table>**

**</div>**

**<div id ="footer">**

**<h3 style="color: blue; text-align: center;">&COPY all rights reserved Mini project</h3>**

**</div>**

**</center>**

**</div>**

**</body>**

**</html>**

**Customer Signup:**

**<html>**

**<head>**

**<title> user sign up</title>**

**<style>**

**body{**

**background:linear-gradient(rgba(22,160,133,1),rgba(46,204,113,1));**

**}**

**#header{**

**height: 100px;**

**width: 100%;**

**background: rgba(255,255,255,0.2);**

**display: flex;**

**justify-content: space-between;**

**}**

**#logo{**

**display: flex;**

**}**

**h1{**

**margin-left: 80px;**

**color: white;**

**}**

**ul li {**

**display: inline-block;**

**width: 70px;**

**}**

**a{**

**color: white;**

**text-decoration: none;**

**font-size: 20px;**

**}**

**a:hover{**

**color:blue;**

**background-color: white;**

**border-radius: 10px;**

**}**

**#body{**

**height: 500px;**

**width: 100%;**

**}**

**#box{**

**height: 300px;**

**width: 300px;**

**border: 3px solid white;**

**margin-top: 50px;**

**border-radius: 15px;**

**}**

**th{**

**color: white;**

**}**

**#footer{**

**height: 50px;**

**width: 100%;**

**background: rgba(255,255,255,0.2);**

**}**

**input[type=text]{**

**border-radius: 10px;**

**}**

**input[type=email]{**

**border-radius: 10px;**

**}**

**input[type=number]{**

**border-radius: 10px;**

**}**

**input[type=date]{**

**border-radius: 10px;**

**}**

**input[type=password]{**

**border-radius: 10px;**

**}**

**input[type=submit]{**

**border-radius: 10px;**

**}**

**</style>**

**</head>**

**<body>**

**<div id="header">**

**<div id ="logo">**

**<img src="logo.jpeg" height="90" width="150">**

**<h1>Charging Point Availability For Electric Vehicle</h1>**

**</div>**

**<div id ="links">**

**<ul>**

**<li><a href="index.html">Home</a></li>**

**<li><a href="account.html">User</a></li>**

**<li><a href="admin.html">Admin</a></li>**

**</ul>**

**</div>**

**</div>**

**<div id="body">**

**<center>**

**<div style="width:30%; text-align: center; color: white">**

**<h2>Join our community of our family</h2>**

**</div>**

**<div style="width:30%; text-align: center; color: white">**

**<h2>Customer Registration</h2>**

**</div>**

**<div id="box">**

**<form action ="accountSignAction.jsp" method="post">**

**<table>**

**<h2 style="color :white;" User Signup></h2>**

**<tr>**

**<th> Full name:</th>**

**<td><input type ="text" name="fullname" placeholder ="enter fullname" required></td>**

**</tr>**

**<tr>**

**<th> Email:</th>**

**<td><input type ="text" name="email" placeholder ="enter email" required></td>**

**</tr>**

**<tr>**

**<th> DOB:</th>**

**<td><input type ="date" name ="dob"></td>**

**</tr>**

**<tr>**

**<th> User name:</th>**

**<td><input type ="text" name="username" placeholder="enter user name" required></td>**

**</tr>**

**<tr>**

**<th> password:</th>**

**<td><input type ="password" name ="password" placeholder="enter password" required></td>**

**</tr>**

**<tr>**

**<th> </th>**

**<td> <input type ="submit" value="SignUp">**

**</td>**

**</tr>**

**<tr>**

**<th> </th>**

**<td style="color: white;"> Already hava a account? <a href ="account.html"> Sign In</a></td>**

**</tr>**

**</table>**

**</form>**

**</div>**

**<div id ="footer">**

**<h3 style="color: blue; text-align: center;">&COPY all rights reserved Mini project</h2>**

**</div>**

**</center>**

**</div>**

**</body>**

**Admin Login:**

**<html>**

**<head>**

**<title> admin login</title>**

**<style>**

**body{**

**background:linear-gradient(rgba(45,55,255,1),rgba(46,204,113,1));**

**}**

**#header{**

**height: 100px;**

**width: 100%;**

**background: rgba(255,255,255,0.2);**

**display: flex;**

**justify-content: space-between;**

**}**

**#logo{**

**display: flex;**

**}**

**h1{**

**margin-left: 80px;**

**color: white;**

**}**

**ul li {**

**display: inline-block;**

**width: 70px;**

**}**

**a{**

**color: white;**

**text-decoration: none;**

**font-size: 20px;**

**}**

**a:hover{**

**color:blue;**

**background-color: white;**

**border-radius: 10px;**

**}**

**#body{**

**height: 500px;**

**width: 100%;**

**}**

**#box{**

**height: 180px;**

**width: 300px;**

**border: 3px solid white;**

**margin-top: 50px;**

**border-radius: 15px;**

**}**

**th{**

**color: white;**

**}**

**#footer{**

**height: 50px;**

**width: 100%;**

**background: rgba(255,255,255,0.2);**

**}**

**input[type=text]{**

**border-radius: 10px;**

**}**

**input[type=password]{**

**border-radius: 10px;**

**}**

**input[type=submit]{**

**border-radius: 10px;**

**}**

**</style>**

**</head>**

**<body>**

**<div id="header">**

**<div id ="logo">**

**<img src="logo.jpeg" height="90" width="150">**

**<h1>Charging Point Availability For Electric Vehicle</h1>**

**</div>**

**<div id ="links">**

**<ul>**

**<li><a href="index.html">Home</a></li>**

**<li><a href="account.html">User</a></li>**

**<li><a href="admin.html">Admin</a></li>**

**</ul>**

**</div>**

**</div>**

**<div id="body">**

**<center>**

**<div style="width:30%; text-align: center; color: white">**

**<h2>Welcome back to our community </h2>**

**</div>**

**<div style="width:30%; text-align: center; color: white">**

**<h2>Customer Login</h2>**

**</div>**

**<div id="box">**

**<table>**

**<h2 style="color :white"; User login></h2>**

**<tr>**

**<th> User name:</th>**

**<td><input type ="text" name ="user name" placeholder="enter user name" required></td>**

**</tr>**

**<tr>**

**<th> password:</th>**

**<td><input type ="password" name ="password" placeholder="enter password" required></td>**

**</tr>**

**<tr>**

**<th> </th>**

**<td> <input type ="submit" value="SignIn">**

**</td>**

**</tr>**

**<tr>**

**<th> </th>**

**<td style="color: white;"> Create new account? <a href ="accountsignup.html"> Sign Up</a></td>**

**</tr>**

**</table>**

**</div>**

**<div id ="footer">**

**<h3 style="color: blue; text-align: center;">&COPY all rights reserved Mini project</h3>**

**</div>**

**</center>**

**</div>**

**</body>**

**</html>**

**Account Action:**

**<%--**

**Document : accountSignAction**

**Created on : 28 Jun, 2024, 11:16:59 AM**

**Author : DELL**

**--%>**

**<%@page import="java.sql.Statement"%>**

**<%@page import="java.sql.Connection"%>**

**<%@page import="java.sql.DriverManager"%>**

**<%@page contentType="text/html" pageEncoding="UTF-8"%>**

**<%**

**String fullname=request.getParameter("fullname");**

**String email=request.getParameter("email");**

**String dob=request.getParameter("dob");**

**String username=request.getParameter("username");**

**String password=request.getParameter("password");**

**try{**

**Class.forName("com.mysql.jdbc.Driver");**

**Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/chargingpointavaliblityforekectricvehicle" ,"root", "root");**

**Statement st=con.createStatement();**

**int i=st.executeUpdate("insert into user value('"+fullname+"','"+email+"','"+dob+"','"+username+"','"+password+"')");**

**if(i>0){**

**%>**

**<script type="text/javascript">**

**window.alert("Signup Success");**

**window.location="accountsignup.html";**

**</script>**

**<%**

**}else{**

**%>**

**<script type="text/javascript">**

**window.alert("Signup Failed");**

**window.location="accountsignup.html";**

**</script>**

**<%**

**}**

**}catch(Exception e){**

**out.println(e);**

**}**

**%>**

**Account LogAction.html**

**<%--**

**Document : accountlogaction.jsp**

**Created on : 29 Jun, 2024, 2:42:35 PM**

**Author : DELL**

**--%>**

**<%@page import="java.sql.ResultSet"%>**

**<%@page import="java.sql.Statement"%>**

**<%@page import="java.sql.Connection"%>**

**<%@page import="java.sql.DriverManager"%>**

**<%@page contentType="text/html" pageEncoding="UTF-8"%>**

**<%**

**String username=request.getParameter("username");**

**String password=request.getParameter("password");**

**try{**

**Class.forName("com.mysql.jdbc.Driver");**

**Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/chargingpointavaliblityforekectricvehicle","root","root");**

**Statement st=con.createStatement();**

**ResultSet rs=st.executeQuery("select \* from user where username='"+username+"'and password='"+password+"'");**

**if(rs.next()){**

**%>**

**<script type="text/javascript">**

**window.alert("Login Success");**

**window.location="accountHome.jsp";**

**</script>**

**<%**

**}else{**

**%>**

**<script type="text/javascript">**

**window.alert("Login Failed");**

**window.location="account.html";**

**</script>**

**<%**

**}**

**}catch(Exception e){**

**out.println(e);**

**}**

**%>**

**CHAPTER-10**

**RESULTS/DISCUSSIONS**

**10.1 SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

**UNIT TESTING**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**INTEGRATION TESTING**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**FUNCTIONAL TEST**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

**Functional testing is centered on the following items:**

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**SYSTEM TEST**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**WHITE BOX TESTING**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**BLACK BOX TESTING**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**UNIT TESTING**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**TEST STRATEGY AND APPROACH**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**ACCEPTANCE TESTING**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**10.1.1 TEST CASES:**

**Test case1 for Login form:**

|  |  |
| --- | --- |
| **FUNCTION:** | **LOGIN** |
| **EXPECTED RESULTS:** | Should Validate the user and check his existence in database |
| **ACTUAL RESULTS:** | Validate the user and checking the user against the database |
| **LOW PRIORITY** | **No** |
| **HIGH PRIORITY** | **Yes** |

**Test case2:**

**Test case for User Registration form:**

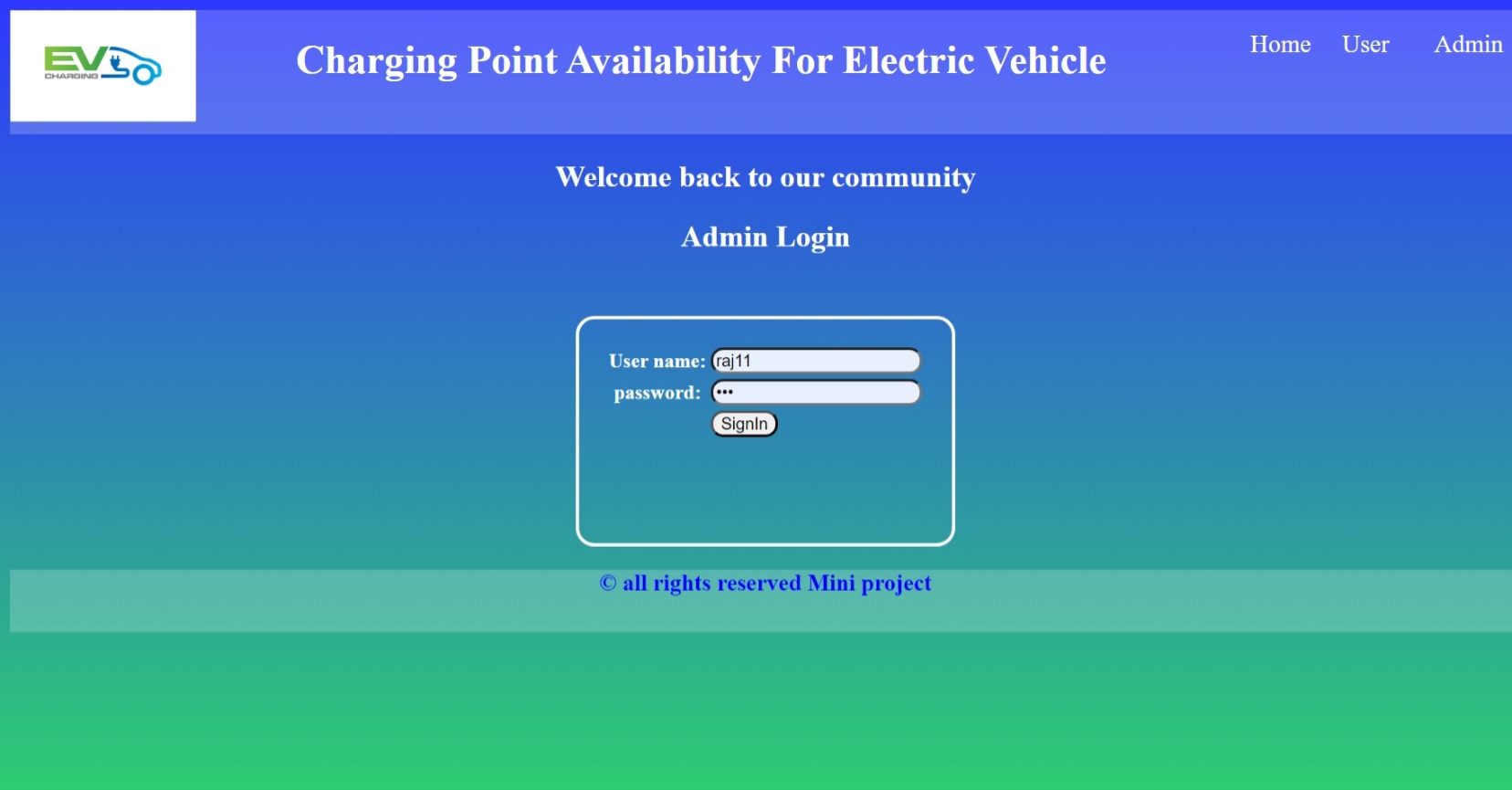
|  |  |
| --- | --- |
| **FUNCTION:** | **USER REGISTRATION** |
| **EXPECTED RESULTS:** | Should check if all the fields are filled by the user and saving the user to database. |
| **ACTUAL RESULTS:** | Checking whether all the fields are field by user or not through validations and saving user. |
| **LOW PRIORITY** | **No** |
| **HIGH PRIORITY** | **Yes** |

**10.2 SCREENSHOTS:**

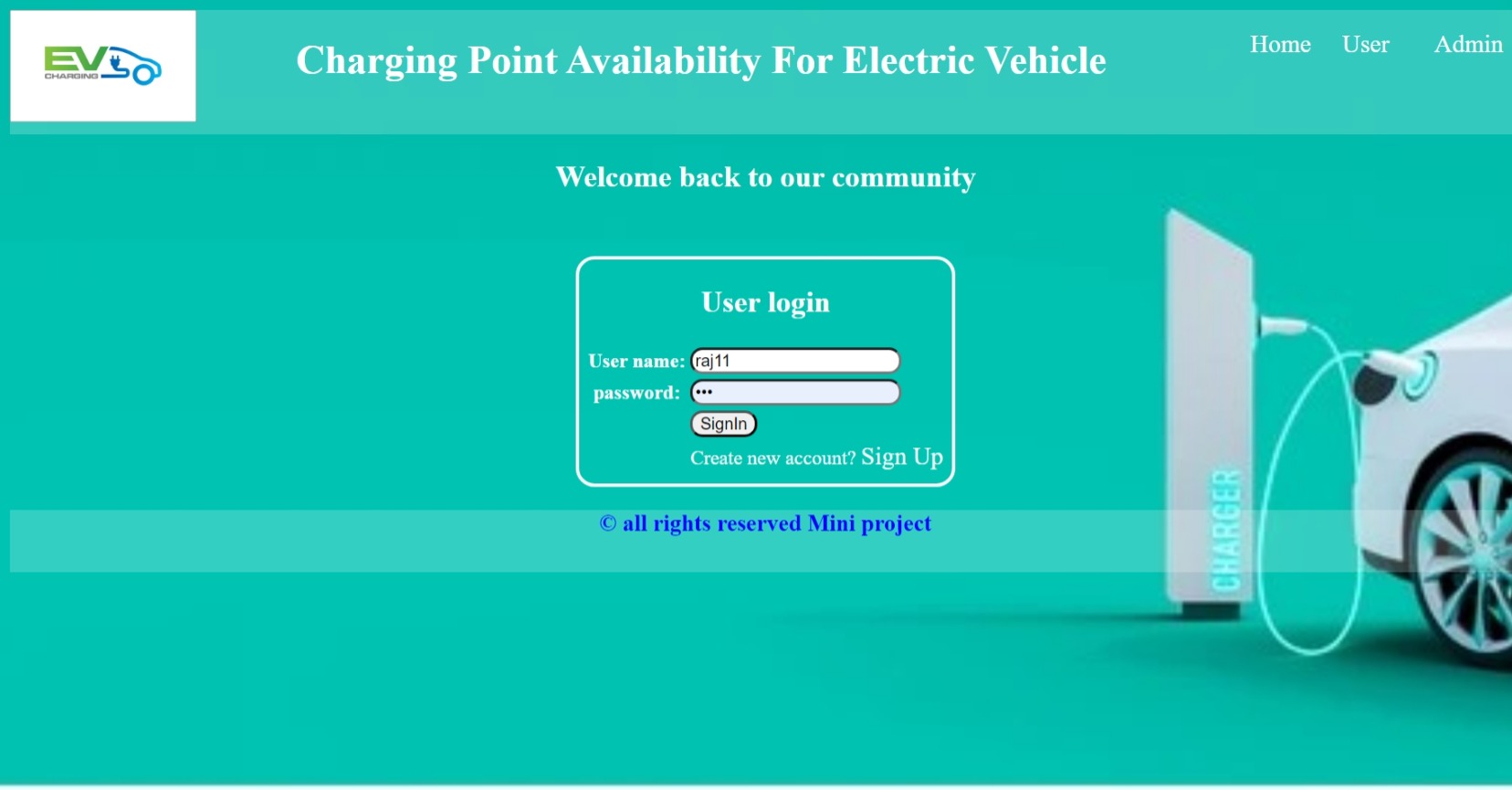
**Home Page:**

****

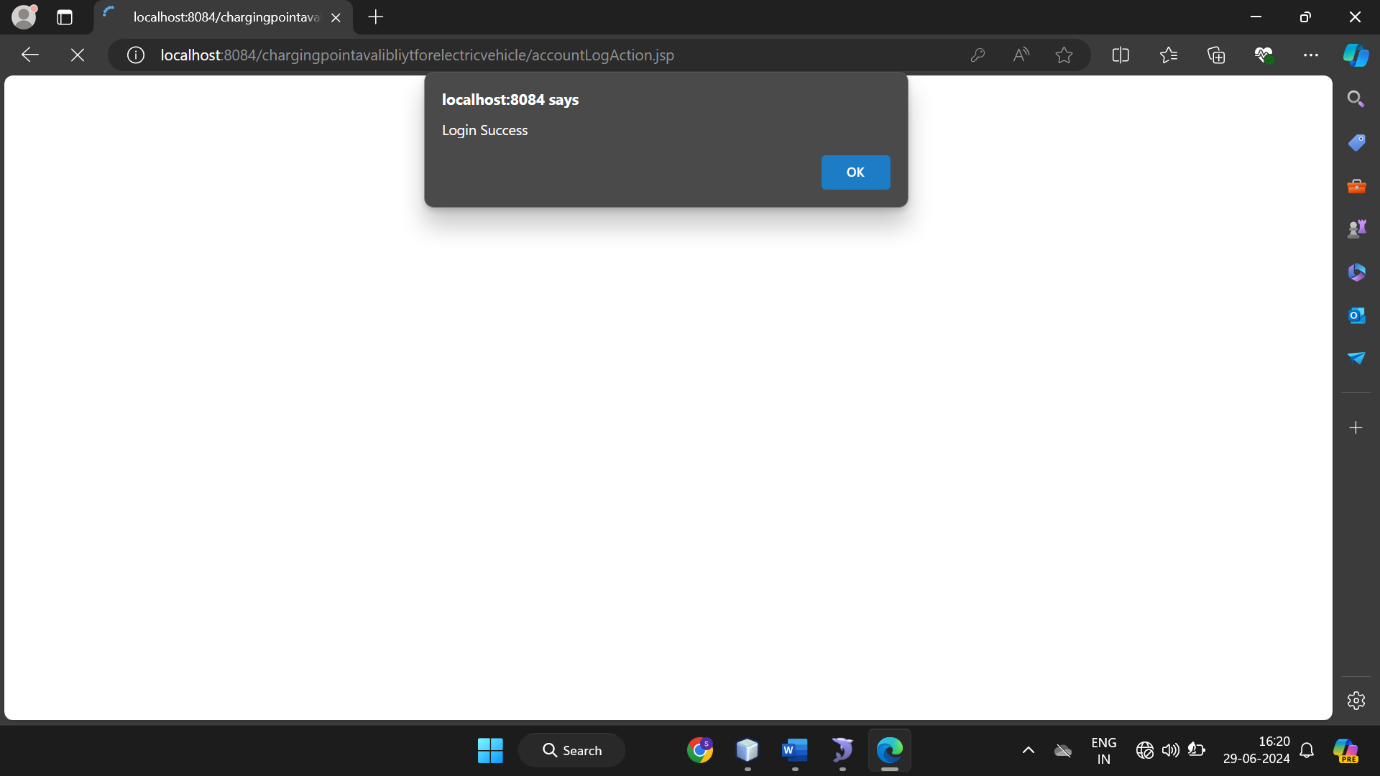
**Admin Login:**

****

**User Login:**

****

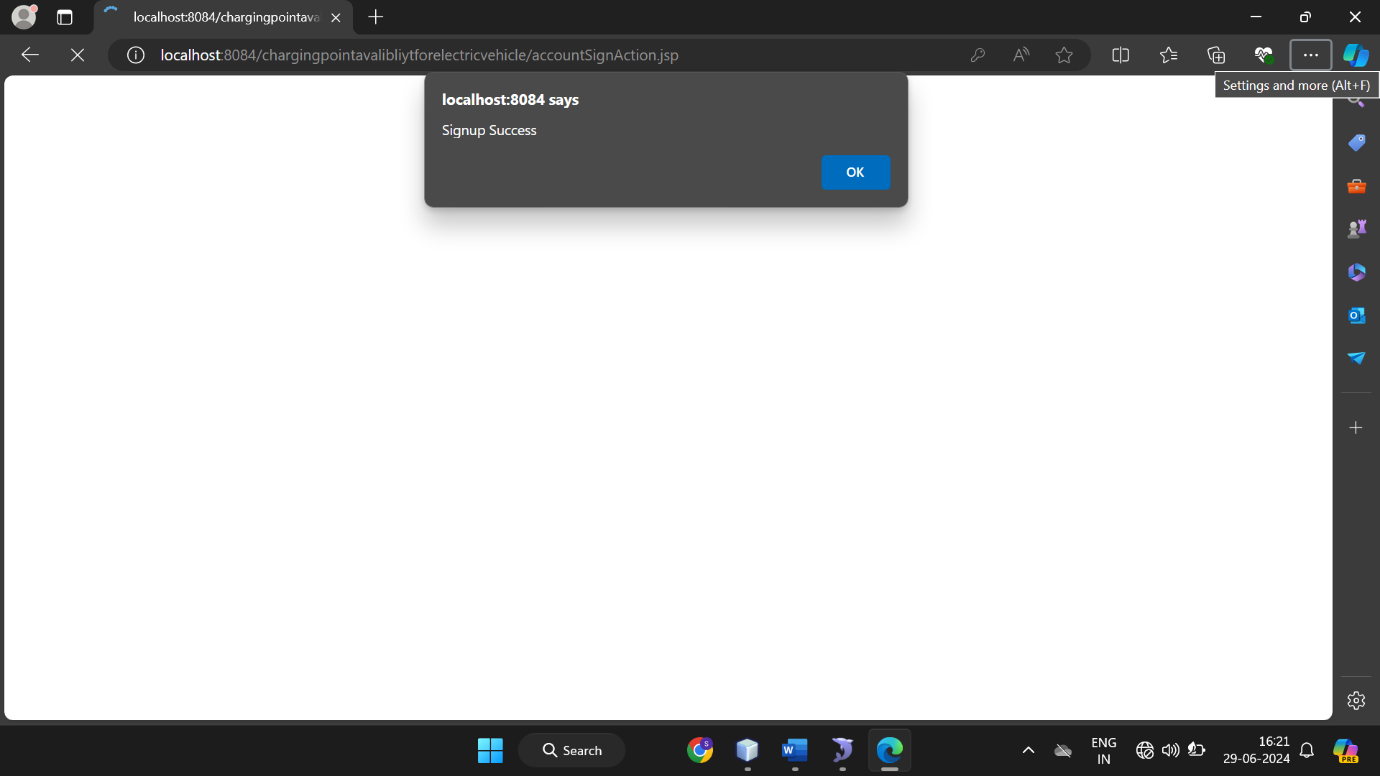
**Login Success:**

****

**User Register:**

****

**Signup Success:**

****

**CHAPTER-11**

**CONCLUSION**

**11.1 CONCLUSION**

In this project, we have developed a web-based platform that helps EV drivers search for nearby available charging stations and shows them the number of available stands at each station. Our platform also helps charging station owners register their stations and increase their visibility and customer base. Our platform addresses the problem of finding a convenient and reliable charging station for EVs, which is a challenge for many EV drivers in India. Our platform also contributes to the promotion of e-mobility, the reduction of greenhouse gas emissions, and the improvement of air quality in India.

Our platform has several advantages over the existing system of manual checking of EV charging point availability, such as saving time and hassle for EV drivers, reducing the risk of running out of charge or getting stranded, creating efficiencies and balances in the utilization of charging infrastructure, and encouraging potential EV buyers to switch to EVs. Our platform also has a promising future scope, as it can expand its coverage and reach, integrate with other web services and applications, leverage data analytics and artificial intelligence, support new technologies and standards, and align with the broader goals and vision of the government and the society.

We hope that our web-based platform for EV charging point availability will provide a useful and innovative service for EV drivers and charging station owners in India, and help them enjoy the benefits of e-mobility. We also hope that our project will inspire other developers and researchers to create more solutions and opportunities for the EV sector in India, and make a positive impact on the environment and the society.

**11.2 FUTURE SCOPE**

This platform can expand its coverage and reach to more regions and cities in India, where EV adoption is growing and charging infrastructure is needed.

This platform can integrate with other web services and applications, such as maps, navigation, payment, booking, etc., to provide a seamless and convenient user experience for EV drivers and charging station owners.

This platform can leverage data analytics and artificial intelligence to optimize the utilization and management of charging infrastructure, such as predicting demand, pricing, load balancing, smart charging, etc.

This platform can support the development and implementation of new technologies and standards for EV charging, such as wireless charging, vehicle-to-grid, vehicle-to-home, etc., to enhance the efficiency and reliability of EV charging.

This platform can contribute to the broader goals and vision of the government and the society for promoting e-mobility, reducing greenhouse gas emissions, and improving air quality.

**CHAPTER-12**

**REFERENCES**

[1] T. Stocker, D. Qin, G. Plattner, M. Tignor, S. Allen, J.Boschung, A. Nauels, Y. Xia, V. Bex, and P. Midgley, Climate change 2013: The physical science basis. Contribution of working group I to the fifth assessment report of the intergovernmental panel on climate change,

(IPCC), Cambridge Univ. Press, Cambridge, U.K., Tech. Rep., 2013.[Online]. Available: https://www.ipcc.ch/report/ar5/wg1/

[2] UNFCCC, Adoption of the Paris agreement, United NationsFramework Conv. Climate Change (UNFCCC), New York, NY,USA, Tech. Rep. FCCC/CP/2015/l.9/rev.1, 2015. [Online]. Available:http://unfccc.int/resource/docs/2015/cop21/eng/l09r01.pdf

[3] Tracking Clean Energy Progress 2017, Int. Energy Agency, Paris,France, 2017. [Online]. Available: http://www.iea.org/reports/trackingclean-energy-progress-2017

[4] M. Weiss, P. Dekker, A. Moro, H. Scholz, and M. K. Patel, On theelectri cation of roadtransportation A review of the environmental,economic, and social performance of electric two-wheelers, Transp.Res. D, Transp. Environ., vol. 41, pp. 348366, Dec. 2015.

[5] Global EV Outlook 2020, Int. Energy Agency, Paris, France, 2021.[Online]. Available: https://www.iea.org/reports/global-ev-outlook-2021

[6] F. Silver and T. Brotherton, CalHEAT research and market transformation roadmap for medium-and heavy-duty trucks, CaliforniaHybrid, Ef cient Adv. Truck Res. Center, Pasedena, CA, USA, Tech.Rep. Draft Publication Rev # 7, Jun. 2013. [Online]. Available:https://calstart.org/wp-content/uploads/2018/10/CalHEAT-Roadmap.pdf

[7] M. Moultak, N. Lutsey, and D. Hall, Transitioning to zeroemission heavy-duty freight vehicles, Int. Council Clean Transp.,Washington, DC, USA, White Paper, 2017. [Online]. Available:https://theicct.org/sites/default/ les/publications/Zero-emission-freighttrucks\_ICCT-white-paper\_26092017\_vF.pdf

[8] E.Mulholland,J.Teter,P.Cazzola,Z.McDonald,andB. P. Ó. Gallachóir,The long haul towards decarbonising road freight Aglobal assessmentto 2050, Appl. Energy, vol. 216, pp. 678693, Apr. 2018.

[9] H. Talebian, O. E. Herrera, M. Tran, and W. Mérida, Electri cation ofroad freight transport: Policy implications in British Columbia, EnergyPolicy, vol. 115, pp. 109118, Apr. 2018.

[10] S. E. Vore, M. Kosowski, M. L. Reid, Z. Wilkins, J. Minicucci, andT. H. Bradley, Measurement of medium-duty plug-in hybrid electricvehicle fuel economy sensitivity to ambient temperature, IEEE Trans.Transport. Electri c., vol. 4, no. 1, pp. 184189, Mar. 2018.

[11] M. Ceraolo, G. Lutzemberger, C. Scarpelli, G. P. Bonelli, and T. Piazza,Hybridisation of forklift trucks, IET Electr. Syst. Transp., vol. 10, no. 1,pp. 116123, Mar. 2020.

[12] C. Jin, T. Yi, Y. Shen, A. Khajepour, and Q. Meng, Comparative studyon the economy of hybrid mining trucks for open-pit mining, IET Intell.Transp. Syst., vol. 13, no. 1, pp. 201208, Jan. 2019.

[13] W. Lhomme, A. Bouscayrol, S. A. Syed, S. Roy, F. Gailly, and O. Pape,Energy savings of a hybrid truck using a ravigneaux gear train, IEEETrans. Veh. Technol., vol. 66, no. 10, pp. 86828692, Oct. 2017.

[14] M. Fries, M. Kruttschnitt, and M. Lienkamp, Operational strategy ofhybrid heavy-duty trucks by utilizing a genetic algorithm to optimize thefuel economy multiobjective criteria, IEEE Trans. Ind. Appl., vol. 54,no. 4, pp. 36683675, Jul. 2018.

[15] D. Wang, P. Hu, J. Du, P. Zhou, T. Deng, and M. Hu, Routing andscheduling for hybrid truck-drone collaborative parcel delivery with independent and truck-carried drones, IEEE Internet Things J., vol. 6, no. 6,pp. 1048310495, Dec. 2019.

[16] T. H. Pham, J. T. B. A. Kessels, P. P. J. van den Bosch, andR. G. M. Huisman, Analytical solution to energy management guaranteeing battery life for hybrid trucks, IEEE Trans. Veh. Techn., vol. 65,no. 10, pp. 79567971, Oct. 2016.

[17] K. Forrest, M. M. Kinnon, B. Tarroja, and S. Samuelsen, Estimatingthe technical feasibility of fuel cell and battery electric vehicles for themedium and heavy duty sectors in California, Appl. Energy, vol. 276,Oct. 2020, Art. no. 115439.

[18] The Future of Trucks, Int. Energy Agency, Paris, France, 2017. [Online].Available: http://www.iea.org/reports/the-future-of-trucks

[19] Global EV Outlook 2019, Int. Energy Agency, Paris, France, 2019.[Online]. Available: http://www.iea.org/reports/global-ev-outlook-2019

[20] J. Shah, M. Nielsen, A. Reid, C. Shane, K. Mathews, D. Doerge, R. Piel,R. Anderson,A.Boulanger,L.Wu,V.Bhandari,A.Gagneja,A.Kressner,X.Li, andS.Sarkar, Cost-optimal, robust charging of electrically-fueledcommercial vehicle eets via machine learning, in Proc. IEEE Int. Syst.Conf., Mar. 2014, pp. 6571.

[21] T. Zhou, M. J. Roorda, H. L. MacLean, and J. Luk, Life cycle GHGemissions and lifetime costs of medium-duty diesel and battery electrictrucks in Toronto, Canada, Transp. Res. D, Transp. Environ., vol. 55,pp. 9198, Aug. 2017.

[22] I. Mareev, J. Becker, and D. Sauer, Battery dimensioning and life cyclecosts analysis for a heavy-duty truck considering the requirements oflong-haul transportation, Energies, vol. 11, no. 1, p. 55, 2017.

[23] B. Sen, T. Ercan, and O. Tatari, Does a battery-electric truck makea difference? Life cycle emissions, costs, and externality analysis ofalternative fuel-powered class 8 heavy-duty trucks in the united states,J. Cleaner Prod., vol. 141, pp. 110121, Jan. 2017.

[24] H. Liimatainen, O. van Vliet, and D. Aplyn, The potential of electric trucks An international commodity-level analysis, Appl. Energy,vol. 236, pp. 804814, Feb. 2019.

[25] B. A. Davis and M. A. Figliozzi, A methodology to evaluate the competitiveness of electric delivery trucks, Transp. Res. E, Logistics Transp.Rev., vol. 49, no. 1, pp. 823, Jan. 2013.

[26] W. Feng and M. Figliozzi, An economic and technological analysisof the key factors affecting the competitiveness of electric commercialvehicles: A case study from the USA market, Transp. Res. C, Emerg.Technol., vol. 26, pp. 135145, Jan. 2013