A PROJECT REPORT

ON

**Electrical Vehicle charging slot booking application**

Submitted in the fulfillment of the requirements for the Award of

### BACHELOR’S DEGREE IN COMPUTER ENGINEERING

SUBMITTED BY

1)Gadge Vyankatesh Narsing Exam No,

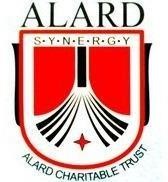
2)Yadav Rahul Indradev Exam No.

3)Vatane Sanket Namdev Exam No.

4)Sahare Aneesh Vijay Exam No.

**Under the Guidance of**

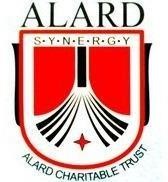
Prof. Sujata Jogdand Gaikwad



**DEPARTMENT OF COMPUTER ENGINEERING** **ALARD COLLEGE OF ENGINEERING AND MANAGEMENT**

**MARUNJI, PUNE-57**

**(AFFILIATED TO SAVITRIBAI PHULE PUNE UNIVERSITY)** **2024 – 2025**



**CERTIFICATE**

This is to certify that the project report entitles

**Electrical Vehicle charging slot booking application**

Submitted by

1)Gadge Vyankatesh Narsing Exam No,

2)Yadav Rahul Indradev Exam No.

3)Vatane Sanket Namdev Exam No.

4)Sahare Aneesh Vijay Exam No.

Is a Bonafide student at this institutes and the work has been carried out by her under the supervision of Prof. Sujata Jogdand Gaikwad and it is approved for the fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of Bachelor of Computer Engineering.

**Prof. Sujata Jogdand Gaikwad Prof. Ganesh Wayal**

Project Guide Head of Department

Dept. of Computer Engg. Dept. of Computer Engg.

**Dr. C.P Jawahar**

External Examiner Principal

**Acknowledgements**

We would like to express our deepest gratitude to our project guide, **Prof. Sujata Jogdand Gaikwad and Prof. Ganesh Goyal** for their unwavering support, insightful guidance, and continuous encouragement throughout the development of this project. Their expertise in the field, constructive feedback, and timely suggestions significantly contributed to the successful implementation of our system. We are truly thankful for their patience and the time they devoted to mentoring us, especially during critical phases of the project.

We would also like to acknowledge the faculty members and administrative staff of the **Department of Computer Engineering** at **Alard college of engineering and mannagement** for providing us with the infrastructure, resources, and academic environment necessary to carry out this project. The knowledge and skills gained through our coursework and lab sessions were instrumental in the realization of this application.

Furthermore, we are grateful to our peers for their collaborative spirit, motivation, and insightful discussions which often led to innovative ideas and solutions. Lastly, we extend our heartfelt thanks to our families for their emotional support and understanding, which kept us motivated during challenging times. This project was a collective effort, and we are proud to present the result of months of dedication and teamwork.

**NAME OF STUDENTS**

1)Gadge Vyankatesh Narsing

2)Yadav Rahul Indradev

3)Vatane Sanket Namdev

4)Sahare Aneesh Vijay

**Abstract**

With the rising global interest in sustainable and eco-friendly transportation, electric vehicles (EVs) are becoming increasingly popular. However, the availability and accessibility of charging infrastructure have not kept pace with the rapid adoption of EVs. One of the major issues faced by EV users is the lack of a centralized system to locate nearby charging stations and book charging slots in real time. This often leads to uncertainty, unnecessary delays, and inefficient use of resources at charging stations.

To address these challenges, we have developed a comprehensive Electric Vehicle Charging Slot Booking Application. This platform enables users to view a list of nearby charging stations, check real-time slot availability, and make reservations in advance. In addition to booking functionalities, the system also offers features such as payment integration, booking history, user authentication, and administrative access for charging station operators.

The application has been designed with a focus on simplicity, responsiveness, and operational efficiency. It provides a seamless user experience across multiple devices and supports both web and mobile platforms. By ensuring better time management, improved resource allocation, and increased transparency, our application contributes to the overall advancement of smart mobility systems. This project lays the foundation for a scalable and adaptive system capable of growing alongside the evolving needs of the electric vehicle ecosystem.

**Indexed Terms**- Electric Vehicles (EV), Charging Slot Booking, Smart Mobility, Real- Time Sc-

-heduling, Sustainable Transportation, Location-Based Services.

# INDEX

1. **[INTRODUCTION](#_bookmark10) 1**
   1. [Introduction](#_bookmark0) 2
   2. [Motivation](#_bookmark11) 2
   3. [Problem Statement](#_bookmark2) 3
   4. [Objective](#_bookmark7) 3
2. **[LITERATURE SURVEY](#_bookmark10) 4**
   1. [Introduction](#_bookmark0) 5
   2. [Literature Survey](#_bookmark5) 5
3. **[SOFTWARE REQUIREMENTS SPECIFICATIONS](#_bookmark9) 8**
   1. [Introduction](#_bookmark0) 9
      1. [Project Scope](#_bookmark7) 9
      2. User Classes and Characteristics 9
      3. [Assumptions and Dependencies](#_bookmark5) 10
   2. [Functional Requirements](#_bookmark12) 10
   3. [External Interface Requirements](#_bookmark3) 11
      1. [User Interfaces](#_bookmark3) 11
      2. [Hardware Interfaces](#_bookmark6) 11
      3. [Software Interfaces](#_bookmark10) 11
      4. [Communication Interfaces](#_bookmark12) 11
   4. [Nonfunctional Requirements](#_bookmark14) 11

|  |  |  |  |
| --- | --- | --- | --- |
|  | [3.4.1](#_bookmark15) | [Performance Requirements.](#_bookmark15) . . . . . . . . . . . . . . .. | . 11 |
|  | [3.4.2](#_bookmark3) | [Safety Requirements](#_bookmark3) . . . . . . . . . . . . . . . . . .. . | . 12 |
|  | [3.4.3](#_bookmark7) | [Security Requirements](#_bookmark7) . . . . . . . . . . . . . . . . . .. | . 12 |
|  | [3.4.4](#_bookmark11) | [Software Quality Attributes](#_bookmark11) . . . . . . . . . . . . . . .. | . 12 |
| [3.5](#_bookmark12) | [System](#_bookmark12) | [Requirements](#_bookmark12) . . . . . . . . . . . . . . . . . . . . . . . . | . 12 |
|  | [3.5.1](#_bookmark13) | [Database Requirements.](#_bookmark13) . . . . . . . . . . . . . . . . . . | . 12 |
|  | [3.5.2](#_bookmark13) | [Software Requirements (Platform Choice)](#_bookmark13) . . . . . . . . . | . 13 |
|  | [3.5.3](#_bookmark2) | [Hardware Requirements](#_bookmark2) . . . . . . . . . . . . . . . . . . | . 13 |

* 1. [Analysis Models: SDLC Model to be applied](#_bookmark8) 13
  2. [System Implementation Plan](#_bookmark4) 14

**[SYSTEM DESIGN](#_bookmark10) 15**

* 1. [System Architecture](#_bookmark0) 16
  2. [Data Flow (DFD) Diagrams](#_bookmark0) 18

[DFD Level 0 Diagram](#_bookmark4) 18

[DFD Level 1 Diagram](#_bookmark0) 19

* 1. [UML Diagrams](#_bookmark0) 20

[Activity diagram](#_bookmark1) 20

[Sequence Diagram](#_bookmark0) 21

[Use case Diagram](#_bookmark0) 22

[Class Diagram](#_bookmark0) 23

* 1. [Entity Relationship Diagrams](#_bookmark0) 24
  2. [Mathematical Model](#_bookmark0) 25

**5** **[IMPLEMENTATION DETAILS](#_bookmark10) 26**

5.1 [Description of Tools](#_bookmark0) 27

5.2 [Programming Language Description](#_bookmark0) 28

5.3 [Implementation Details](#_bookmark0) 29

5.4 Algorithm Details 30

5.5 [Output of Implementation (In form of Screen Shots)](#_bookmark0) 32

**6** **[TESTING](#_bookmark0) 35**

6.1 [Introduction](#_bookmark0) 36

6.2 [Type of Testing](#_bookmark8) 36

6.3 [Test Cases and Results](#_bookmark7) 38

**7** **[CONCLUSION](#_bookmark10) 40**

7**.1** [Conclusion](#_bookmark0) 41

7.2 [Future Scope](#_bookmark9) 41

**8** **[APPENDIX A](#_bookmark10) 42**

8.1 [Journal Paper](#_bookmark0) 43

**9** **[APPENDIX B](#_bookmark10) 45**

9.1 [Plagiarism Report of Project Report](#_bookmark0) 46

1. **APPENDIX C**

10.1 Copyright Certificate…………………………………………………………………………………. 47

**11 [REFERENCES](#_bookmark10)**

# List of Figures

* 1. Waterfall Model 13
  2. System Architecture 17
  3. [DFD Level 0 Diagram](#_bookmark15) 18
  4. DFD Multi Level Diagram 19
  5. Activity Diagram 20
  6. Sequence Diagram 21
  7. Use case Diagram 22
  8. [Class Diagram](#_bookmark12) 23
  9. [ER Diagram](#_bookmark3) 24
  10. Charging station search 32
  11. Slot booking 33
  12. Login 33
  13. Slot allocation 33
  14. Slot booking database 34

6.1 Test case table………………………………………………………………………………………………48

* 1. [Acceptance](#_bookmark12) letter…………………………………………………………………………………. 43
  2. [Acceptance](#_bookmark5) letter 44

8.3 [Plagiarism Report](#_bookmark9) 46

# List of Tables

* 1. System Implementation Plan 14
  2. Summary of performance of Algorithms 31
  3. [Test cases of system](#_bookmark0) 39

# CHAPTER 1: INTRODUCTION

* 1. **INTRODUCTION**

With the increasing popularity of electric vehicles, there is a parallel demand for an organized and reliable charging infrastructure. Despite the expansion of EV usage, users often face delays and uncertainty due to the lack of real-time information on charger availability. This project proposes a digital platform where EV owners can search for stations, view live slot status, and make reservations in advance. Our aim is to bridge the gap between user convenience and operational efficiency, contributing to the development of smart and sustainable transportation systems.

* 1. **MOTIVATION**

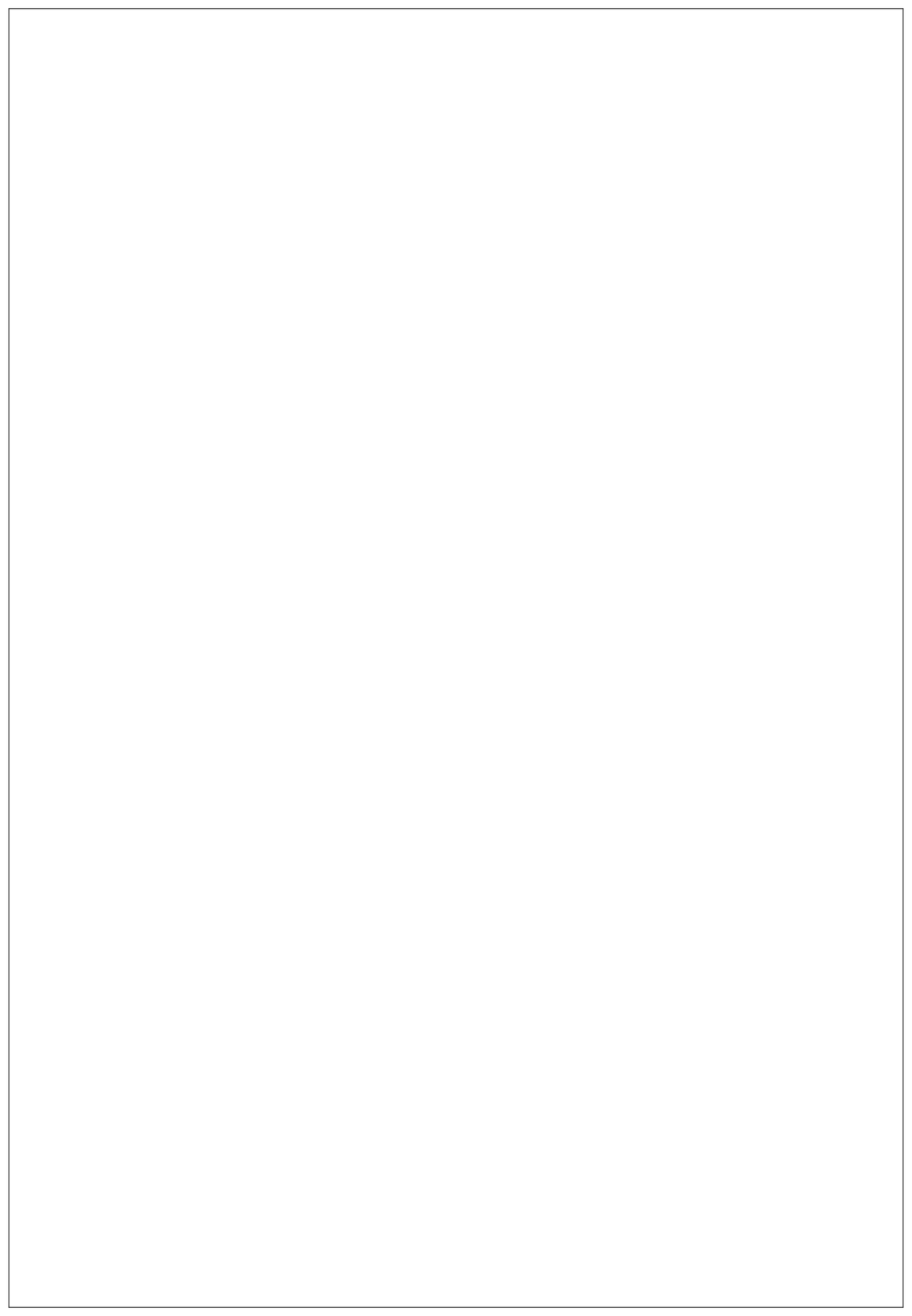
The rapid advancement in electric vehicle (EV) technology has been a major step toward reducing carbon emissions and achieving global sustainability goals. However, the growth of EV adoption has not been matched with an equally efficient charging infrastructure, especially in terms of accessibility and time management. One of the most common challenges faced by EV users is the unpredictability of slot availability at charging stations, which often leads to long waiting times, energy waste, and user dissatisfaction.

This gap in the EV ecosystem inspired us to develop a solution that bridges the disconnect between users and charging infrastructure through technology. We observed that while several EV charging stations are being installed, the lack of a digital system for slot management and pre-booking results in inefficient utilization of resources. By leveraging real-time data and user-friendly design, we were motivated to build a system that simplifies the process of locating charging stations and booking available slots in advance.

* 1. **PROBLEM STATEMENT**

Electric vehicle owners frequently experience inconvenience when searching for available charging stations. The absence of a centralized system for booking and managing charging slots leads to long queues and time wastage. Charging station operators also struggle with unorganized traffic and underutilization of resources. To resolve these issues, a structured platform is required that allows real-time booking, availability tracking, and administrative control. Our project addresses this problem by offering an automated, easy-to-use solution for both users and station managers.

* 1. **OBJECTIVE**
     + To develop a user-friendly web/mobile application that allows EV owners to locate nearby charging stations.
     + To implement a real-time slot booking system that enables users to check availability and reserve charging slots in advance.
     + To reduce waiting times and improve the efficiency of EV charging station usage.
     + To provide charging station administrators with an interface to manage slot bookings, monitor availability, and update station status.
     + To enhance the user experience by integrating features like booking history, status notifications, and secure login functionality.



## **CHAPTER 2 :LITERATURE SURVEY**

* 1. **INTRODUCTION**

The surge in electric vehicle (EV) adoption has underscored the necessity for efficient charging infrastructure. A critical challenge faced by EV users is the lack of real-time information on charging station availability, leading to prolonged waiting times and inefficient resource utilization. To address these issues, various digital solutions have been proposed and implemented.

* 1. **LITERATURE SURVEY**

**S.L. Sreedevi and B.T. Geetha (2023)** introduced a priority-based EV slot booking system to address the inefficiencies in traditional first-come-first-serve charging setups. Their work emphasizes serving emergency vehicles by assigning charging priorities, thereby minimizing waiting time during high-demand periods. While our system draws inspiration from their queue management, it diverges by focusing on public usage with real-time updates and a user-first design, offering flexibility and equal accessibility to all users.

**Ghegade Mayuri et al. (2023)** presented a basic web-based slot booking application that allowed users to reserve charging slots ahead of arrival. Their work laid the groundwork for centralized EV slot management. Our system enhances this concept by adding mobile responsiveness, real-time station updates, and a secure payment gateway — offering a more complete and scalable user experience for widespread public adoption.

**Rukmani Devi D. et al. (2024)** emphasized emergency vehicle prioritization in EV slot booking, considering use cases like ambulances and public service vehicles. Our project, although acknowledging such use cases, is designed for everyday consumer utility. It ensures smooth, efficient booking for individual EV owners while maintaining station availability and administrative control.

**Gowtham R. et al. (2023)** developed an IoT-integrated EV charging architecture where smart devices updated the system with real-time availability. Their model focused on sensor-based hardware implementation. Our work aligns with this approach on the data sync level but is built as a software-driven solution, utilizing available APIs to display live availability without additional hardware constraints.

# CHAPTER 3: SOFTWARE REQUIREMENTS

# SPECIFICATIONS

* 1. **INTRODUCTION**

This Software Requirements Specification (SRS) document outlines the functional and non-functional requirements for the Electric Vehicle Charging Slot Booking Application. This system will provide electric vehicle (EV) users with a platform to search, book, and manage charging slots in real time. The application will aim to streamline the booking process, reduce waiting times, and optimize the utilization of charging stations. This document serves as a reference for developers, testers, and stakeholders involved in the project.

* + 1. **Project Scope**
* Real-time availability checking of charging slots.
* Online booking system for EV charging stations.
* User authentication and authorization for both end-users and administrators.
* Payment gateway integration for charging session payments.
  + 1. **User Classes and Characteristics**

**User classes**

**User Registration and Login:**

The application allows new users to register by providing essential information

such as their name, email address, phone number, and vehicle details. Users can

set a password for secure access. After registration, they can log in using their

credentials. Authentication ensures that only registered users can access the main

features like searching for charging stations, booking slots, and managing their

bookings. The system may also include options like email or SMS verification

during registration for added security.

**Search for Charging Stations**:

Once logged in, users can search for available charging stations based on their

location or by entering the name of a city. The system uses location services

to show nearby stations on a map or in a list format. It displays vital details

such as station distance, available charger types (fast, slow, normal), operating

hours, and user ratings. This helps EV owners quickly identify suitable stations

according to their immediate needs.

**Book a Charging Slot:**

The core functionality of the application is to enable users to book a charging slot

at their chosen station. After selecting a station, users pick a specific date, time, a

-nd charger type based on availability. They confirm the booking through a simple

checkout process, with the option to prepay online if the station requires it. A book

-ing confirmation with details like booking ID, slot time, and station address is sent

to the user via email or SMS.

**Characteristics:**

* + Tech-Savviness: The users may range from tech-savvy individuals to those with limited technical knowledge, so the interface must be intuitive and easy to navigate. Features like location tracking, slot reservations, and payment gateways should be simple and user-friendly.
  + Data Input: Users will input various parameters, such as their vehicle model, battery percentage, and preferred charging time. They may also provide their location to find nearby charging stations.
  + Decision-Making: EV owners rely on the application to help them find the nearest and most convenient charging stations.

**Assumptions and Dependencies Data Availability:**

Assumption: Sufficient and accurate data on charging station locations, availability, and user preferences is available for the system to function effectively. Dependency: The system's accuracy in slot availability, booking status, and optimal station recommendations depends on the real-time accuracy and completeness of the data provided by charging stations and users.

**Model Training:**

Assumption: Any machine learning models used for predictions, such as predicting charging slot demand or estimating user preferences, are trained on a diverse and representative dataset, including factors like location, time of day, and user behavior. Dependency: The effectiveness and accuracy of the predictions depend on the quality and diversity of the training data and the model’s ability to generalize to new data, such as changes in user patterns or station usage.

**Market Conditions:**

Assumption: The platform assumes that the demand for EV charging slots follows a reasonably predictable pattern based on user habits, geographic location, and vehicle usage.

Dependency: Changes in market conditions, such as an increase in the number of EV users, the introduction of new charging stations, or changes in user preferences, can significantly affect the accuracy of demand predictions, as well as the platform’s overall performance and resource allocation.

* 1. **FUNCTIONAL REQUIREMENTS**

Charging Station Data Collection Data Processing Slot Filtering and Availability

Slot Booking and Management Notifications and Confirmation Generation.

* 1. **EXTERNAL INTERFACE REQUIREMENTS**
     1. **User Interfaces**

The system will provide both a web-based interface and a mobile application to facilitate seamless

interaction for users. The web interface will enable users to search for nearby charging stations, view

available time slots, and make bookings easily through a clean and user-friendly design. It will include

clear instructions and real-time feedback during the booking process to enhance the overall user experience.

**3.3.2 Hardware Interfaces**

The system will rely on appropriate hardware to ensure smooth operation and accessibility. Server hardware is essential to host the application and support backend operations, including data processing, booking management, and real- time communication. These servers must be capable of handling high volumes of concurrent requests and supporting database operations efficiently.

**3.3.3 Software Interfaces**

The backend of the application will be developed using robust web frameworks such as Django or Flask, which will handle server-side logic, user authentication, and data routing. For the frontend, technologies like HTML5, CSS3, and JavaScript will be employed, potentially enhanced by libraries such as React.js to deliver an interactive and dynamic user experience.

**3.3.4 Communication Interfaces**

The system will utilize HTTP or HTTPS protocols for secure communication between the client-side applications and the server. This will ensure encrypted data transfer and secure access to booking and user information. Additionally, the platform may integrate external APIs to retrieve auxiliary data, such as location services or charger specifications, and to connect with other services as needed.

**3.4 NONFUNCTIONAL REQUIREMENTS**

**3.4.1 Performance Requirements**

* The system should deliver fast and seamless interactions to ensure a smooth user experience.
* Minimal Response Time.
* High Scalability: The system should handle a large number of concurrent users and bookings across multiple locations Optimal Resource Usages.

**3.4.2 Safety Requirements**

The safety of user data and system operations must be a top priority. The system should ensure data integrity, protecting all user and booking data from unauthorized access, accidental modifications, or deletions. In the event of any operational failure, the system should respond gracefully, providing users with informative error messages while internally logging these issues for further analysis and troubleshooting.

**3.4.3 Security Requirements**

Security is critical for protecting both user data and operational functionality. The system must include secure authentication mechanisms that verify the identity of users and restrict access to sensitive features and data to authorized users only. All critical data, especially user credentials and booking information, should be encrypted during storage.

**3.4.4 Software Quality Attributes**

* **Maintainability:** The system should be designed and implemented in a way that allows for easy maintenance and updates in the future.
* **Reliability:** The system should be reliable, with minimal downtime and errors.
* **Usability:** The system should be user-friendly, with a clear and intuitive interface that allows users to easily input property features and view prediction results.
* **Portability:** The system should be portable, allowing it to run on different platforms and environments .

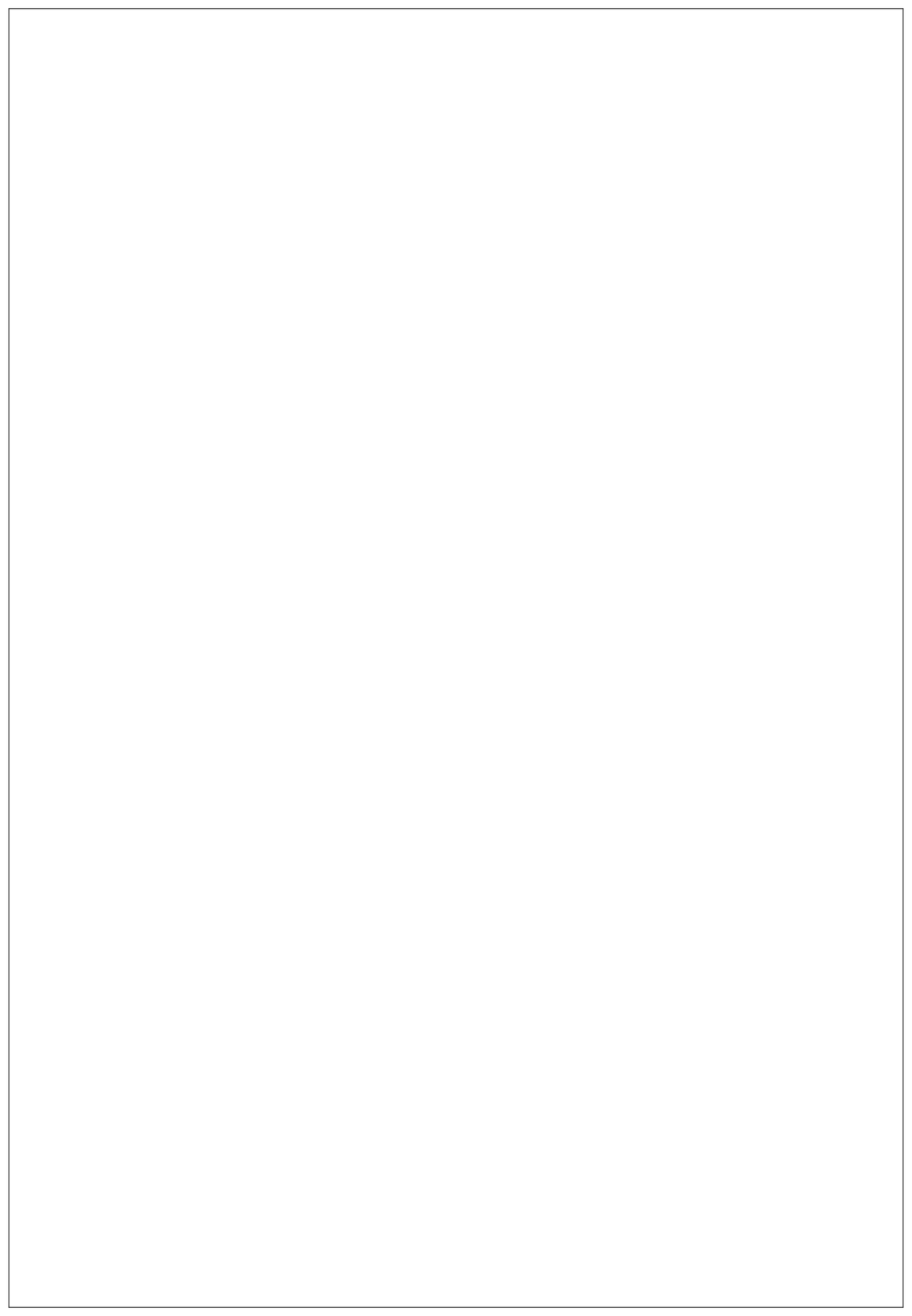
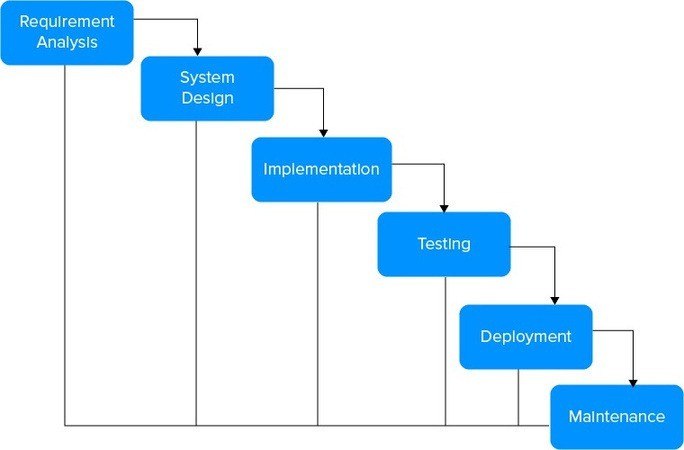
**3.5 SYSTEM REQUIREMENTS**

**Software Requirements**

* **Operating system:** Windows, macOS, and Linux. **Programming Language:** Python and JavaScript.
* **Tools:** Visual Studio Code or PyCharm.
* **Frameworks and Libraries:** Python libraries such as Flask for the backend.

**3.5.1 Hardware Requirements**

* **Processor:** The system should run on a standard processor, such as Intel Core i5 or higher, to handle computations efficiently.
* **Memory:** A minimum of 8 GB RAM is recommended for smooth performance.
* **Storage:** The system should have sufficient storage capacity to store property data, user information, and prediction results. A minimum of 500 GB hard drive space is recommended.
* **Network:** The system should have access to a stable internet connection for real-time data processing and updates.



**3.5.2 ANALYSIS MODELS: SDLC MODEL TO BE APPLIED**

The project plan provided delineates our project’s execution process, following the Software Development Life Cycle (SDLC) phases. It will undergo continuous updates to accurately mirror the project’s real progress and upcoming plans. In the initial stages of requirement gathering and analysis, our primary emphasis was on understanding the problem statement thoroughly and guaranteeing the system’s dependability. This phase also encompassed gathering information about the essential software and hardware components required for the project.

The Waterfall model is a linear and sequential SDLC model where each phase must be completed before moving to the next. This model is suitable for projects with clear and well-defined requirements from the beginning. For your EV charging slot booking system, this model can be used if all features and functionalities (such as user registration, station listing, and booking system) are planned in advance. It ensures systematic development but lacks flexibility for mid-project changes. The Agile model emphasizes iterative development, where requirements and solutions evolve through collaboration

Figure 3.1: Waterfall Model

Our system adheres to the SDLC Waterfall model, comprising several distinct stages:

* Requirement Analysis Stage: This initial phase involves gathering the necessary data to serve as input for the system.
* Design Stage: In this stage, the collected data is formatted into a specific matrix. Scores are assigned to elements in the matrix to facilitate further processing.
* Coding Stage: Here, the system’s main functionality is implemented, including class mapping and the creation of an exact prototype.

**3.6 SYSTEM IMPLEMENTATION PLAN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. NO | Task Name | Begin Date | End Date | Remarks |
| 1 | Selecting project domain | 1 Aug 2024 | 10Aug 2024 | Done |
| 2 | Understanding project need | 11 Aug 2024 | 17 Aug 2024 | Done |
| 3 | Understanding requisites | 18 Aug 2024 | 19 Aug 2024 | Done |
| 4 | Information Gathering | 21 Aug 2024 | 31 Aug 2024 | Done |
| 5 | Literature survey | 1 Sept 2024 | 15 Sept 2024 | Done |
| 6 | Refine project scope | 16 Sept 2024 | 18 Sept 2024 | Done |
| 7 | Concept development | 20 Sept 2024 | 21 Sept 2024 | Done |
| 8 | Planning and scheduling | 22 Sept. 2024 | 23 Sept 2024 | Done |
| 9 | Requirement Analysis, risk identification and monitoring | 25 Sept 2024 | 26 Sept 2024 | Done |
| 10 | Design review and refinement design and modeling | 27 Sept 2024 | 28 Sept 2024 | Done |
| 11 | Implementation | 29 Sept 2024 | 16 Oct 2024 | Done |
| 12 | Review and suggestion | 17 Oct 2024 | 20 Oct 2024 | Done |
| 13 | Outcome assessment | 21 Oct 2024 | 20 Nov 2024 | Done |
| 14 | Testing and QA assistance | 21 Nov 2024 | 15 Feb 2025 | Done |
| 15 | Review and suggestion QA | 16 Feb 2025 | 15 Mar 2025 | Done |

# CHAPTER 4:

# SYSTEM DESIGN

* 1. **SYSTEM ARCHITECTURE**

The system architecture of our Electric Vehicle (EV) Charging Slot Booking Application is designed to ensure efficient interaction between users and charging infrastructure. It follows a three-tier architecture comprising the presentation layer, application layer, and data layer. The front-end (presentation layer) is developed using HTML, CSS, and JavaScript, providing an intuitive interface for users to search, book, and manage charging slots. The application layer, built with Python and Flask, handles business logic, user authentication, and slot booking operations. MySQL is used as the backend database to store user data, station details, and booking information. RESTful APIs ensure smooth communication between the client and server. This architecture ensures scalability, security, and real-time data processing for a seamless user experience.

The system architecture of our Electric Vehicle (EV) Charging Slot Booking Application is structured to provide a reliable, scalable, and user-friendly platform for EV users to locate and reserve charging slots. The architecture follows a three-tier model consisting of the presentation layer, application layer, and data layer.

The **presentation layer** is responsible for user interaction and is developed using HTML, CSS, and JavaScript.

The **application layer** serves as the core of the system and is built using Python and the Flask framework. It manages business logic, user authentication, booking validation, and routing of user requests.

The **data layer** uses MySQL to handle all persistent data, including user profiles, station information, city data, slot availability, and booking history. It supports fast query processing and maintains data integrity.

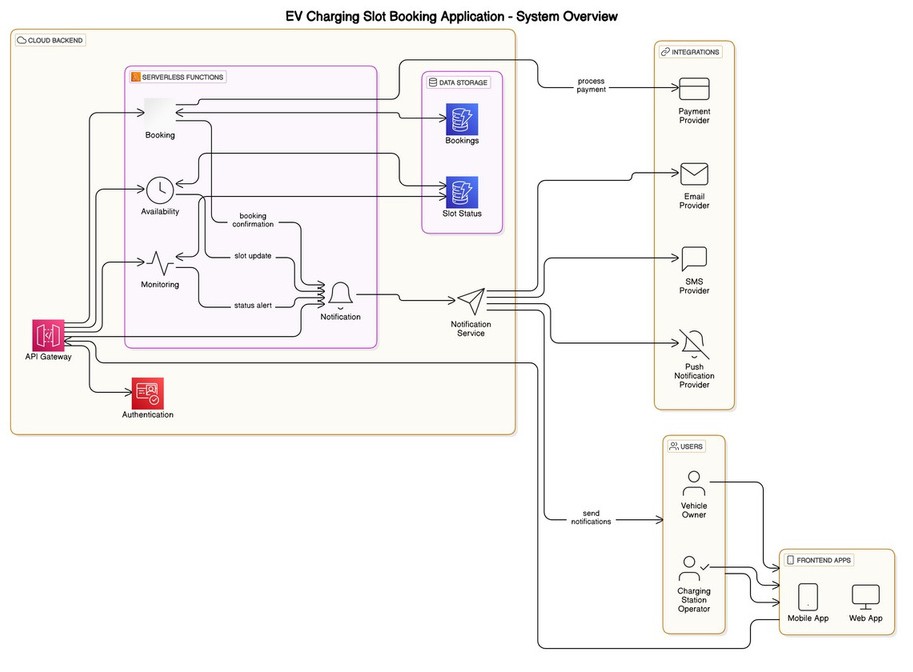
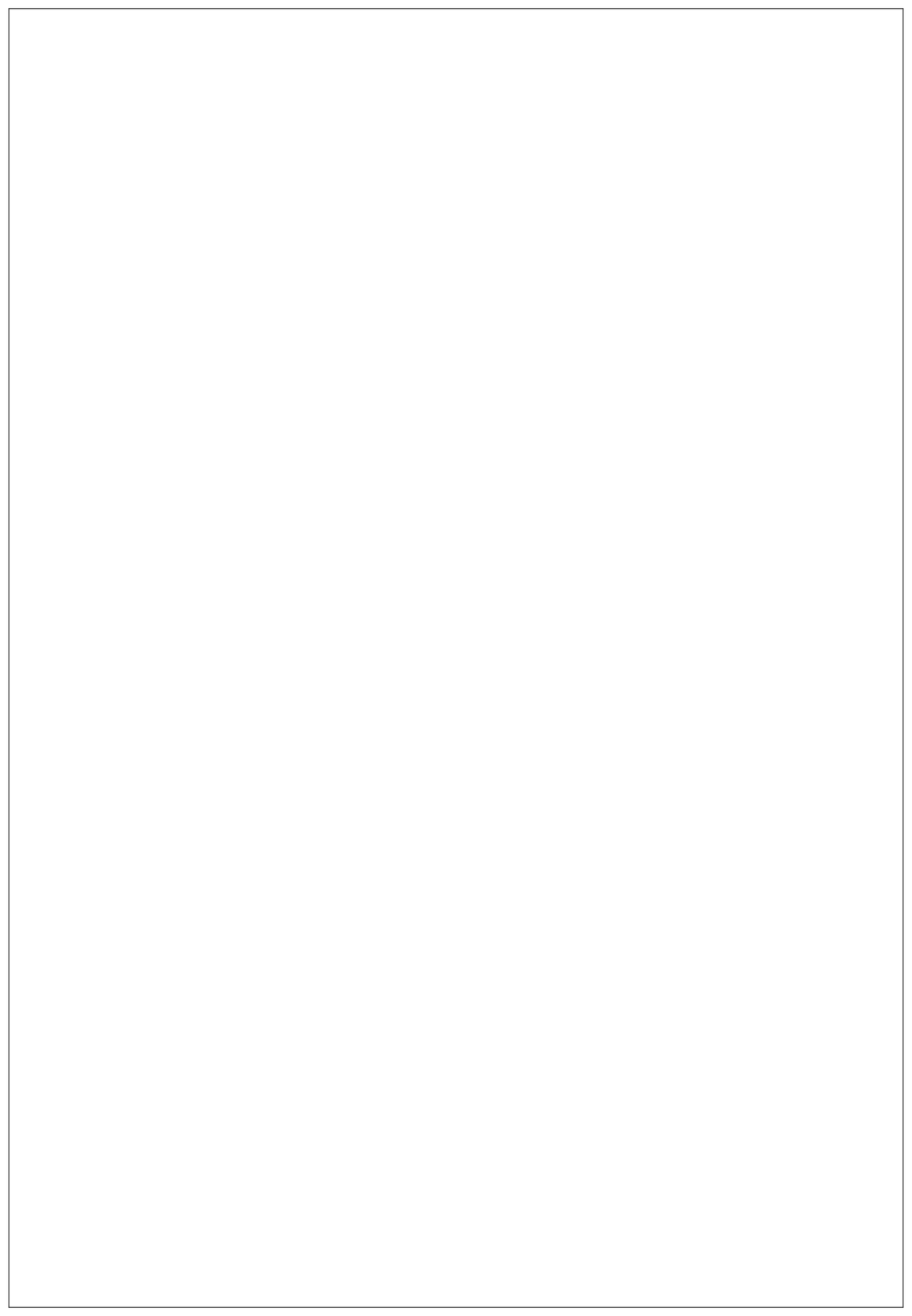
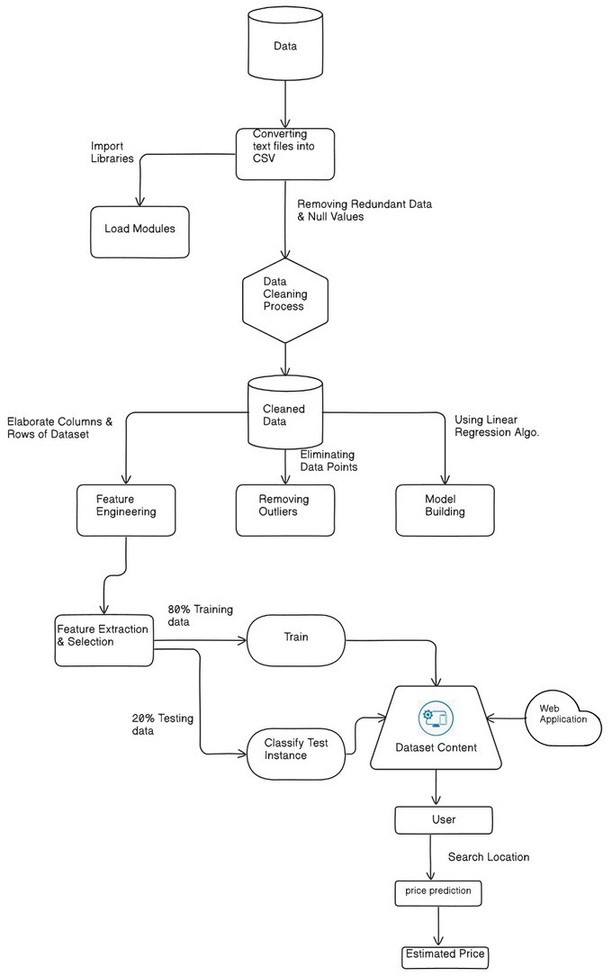
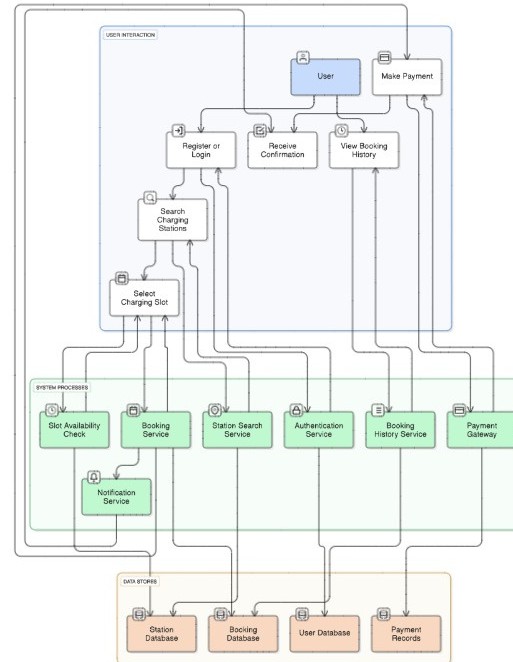
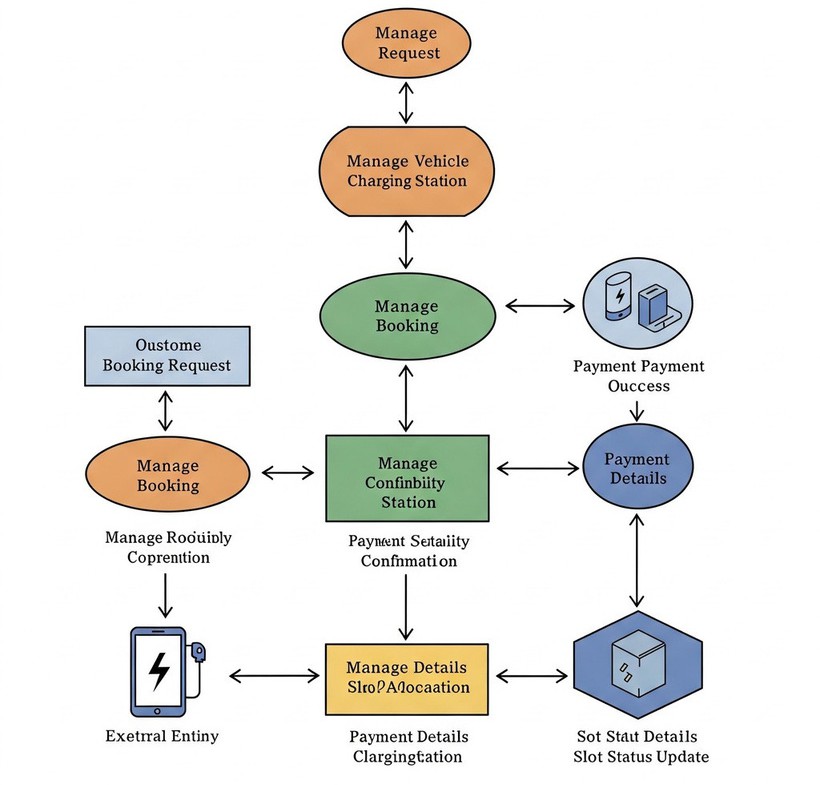
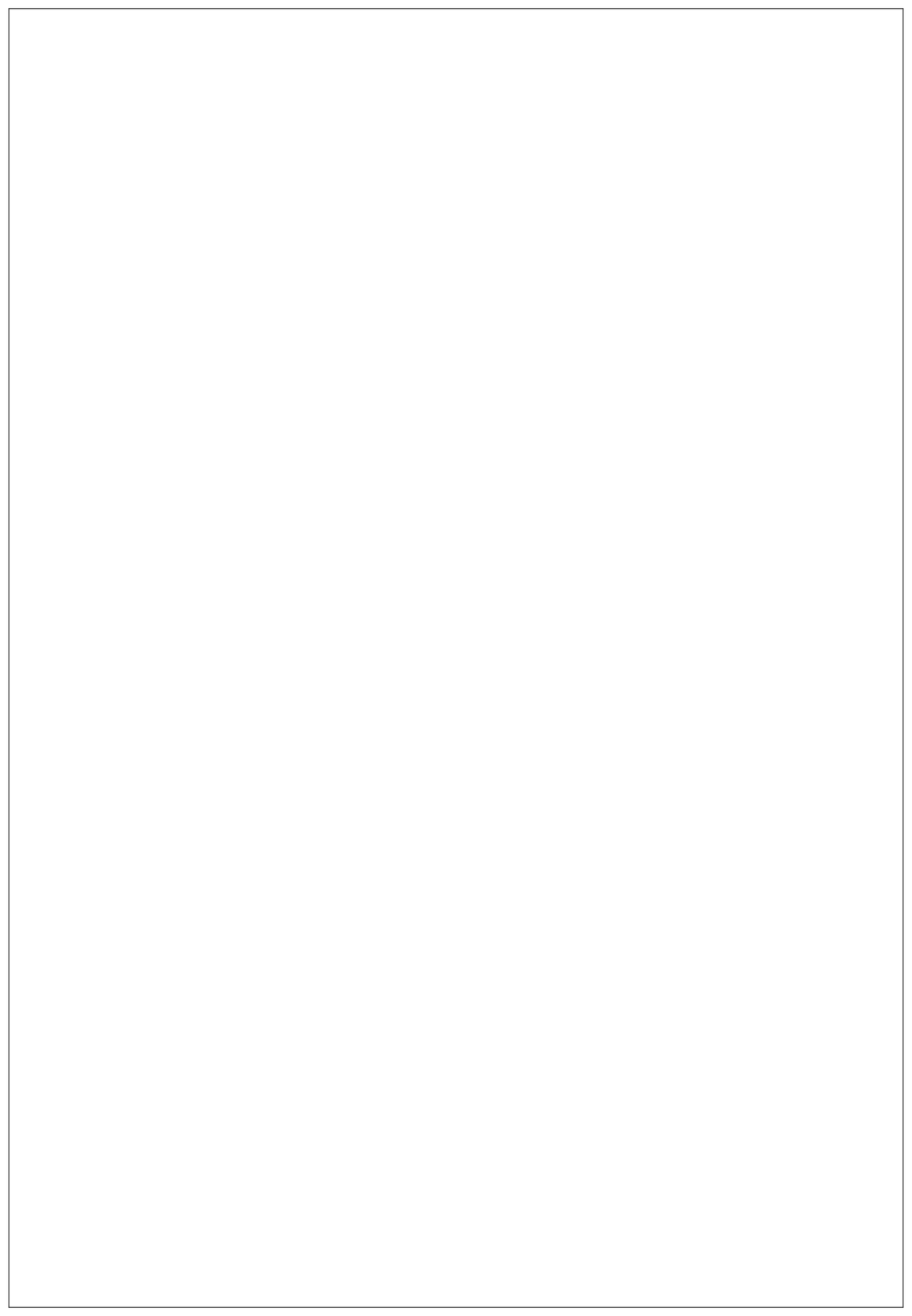
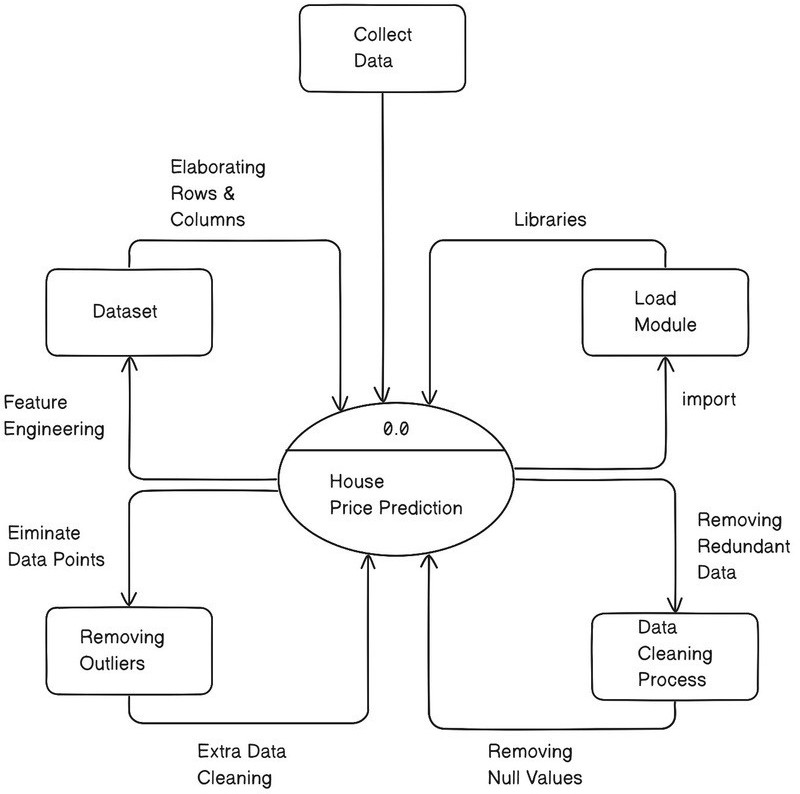


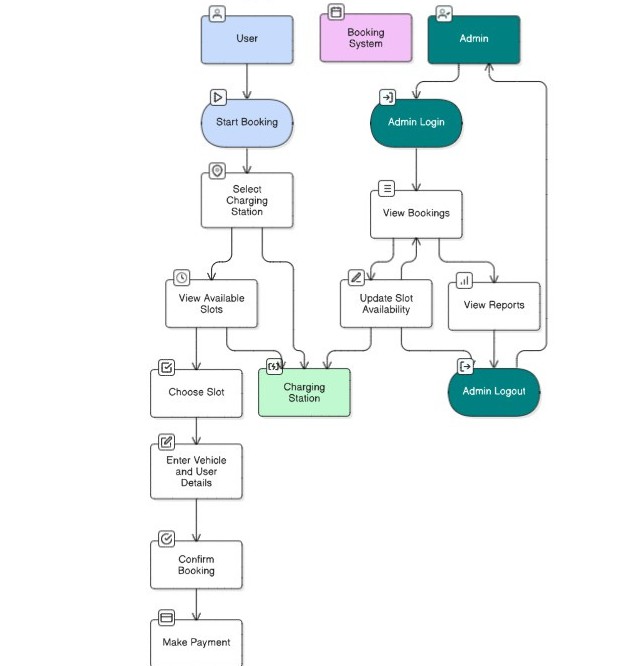
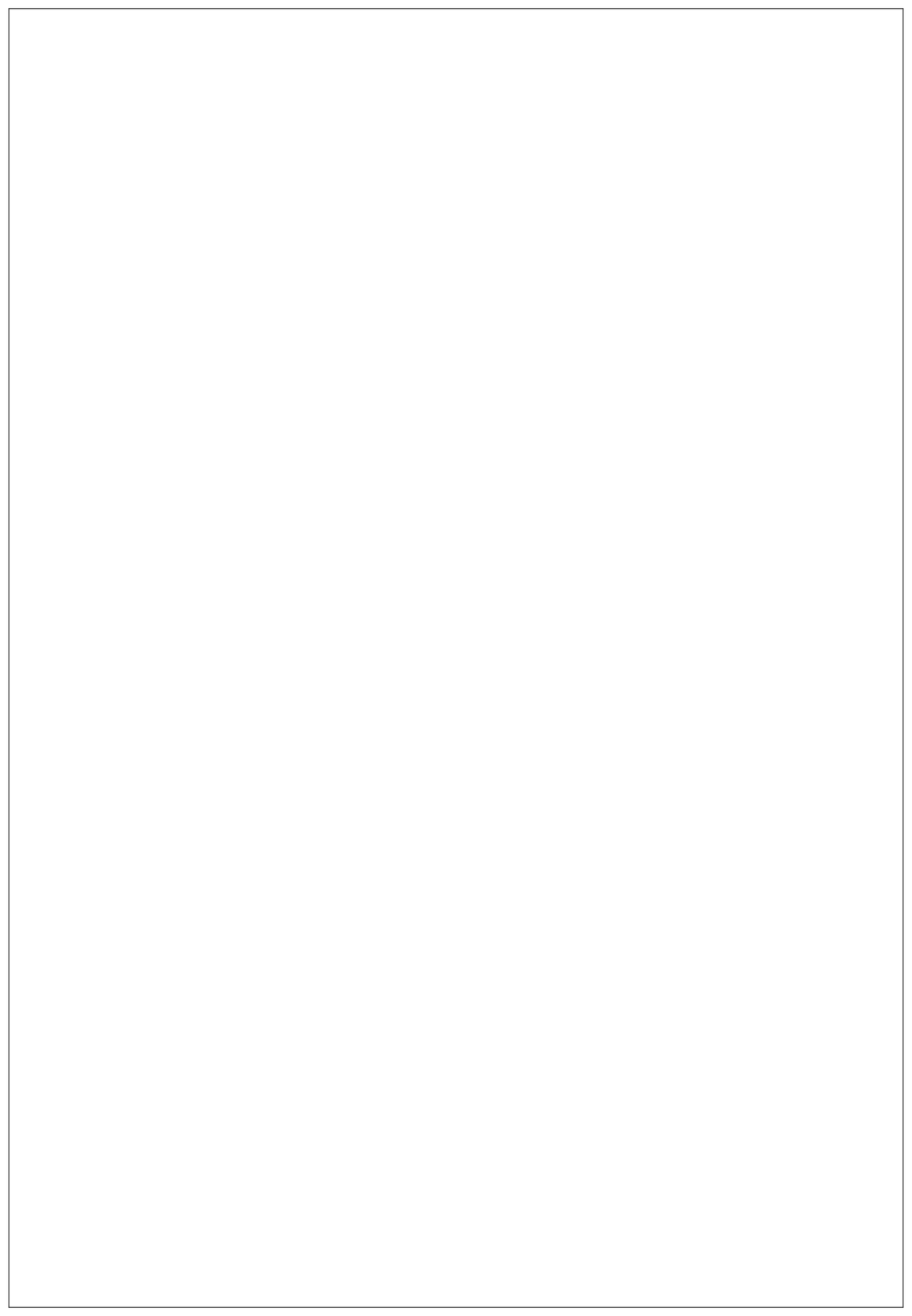
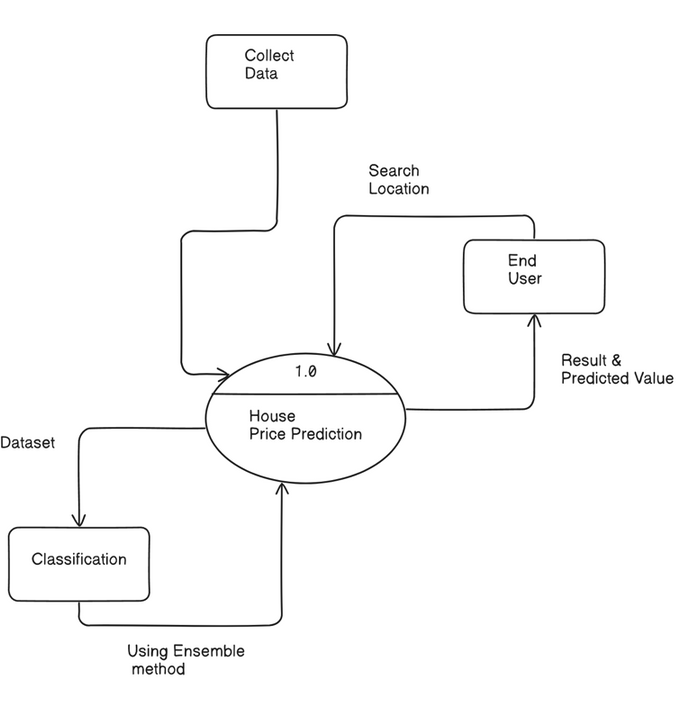
Figure 4.1: System Architecture



* 1. **DATA FLOW (DFD) DIAGRAMS**

A Data Flow Diagram (DFD) visually represents how information flows through a system and undergoes transformations as it moves from input to output. It illustrates the path of information and the processes or transformations that occur at each stage. This graphical technique is essential for understanding how data is processed within a system, from its initial input to the final output.

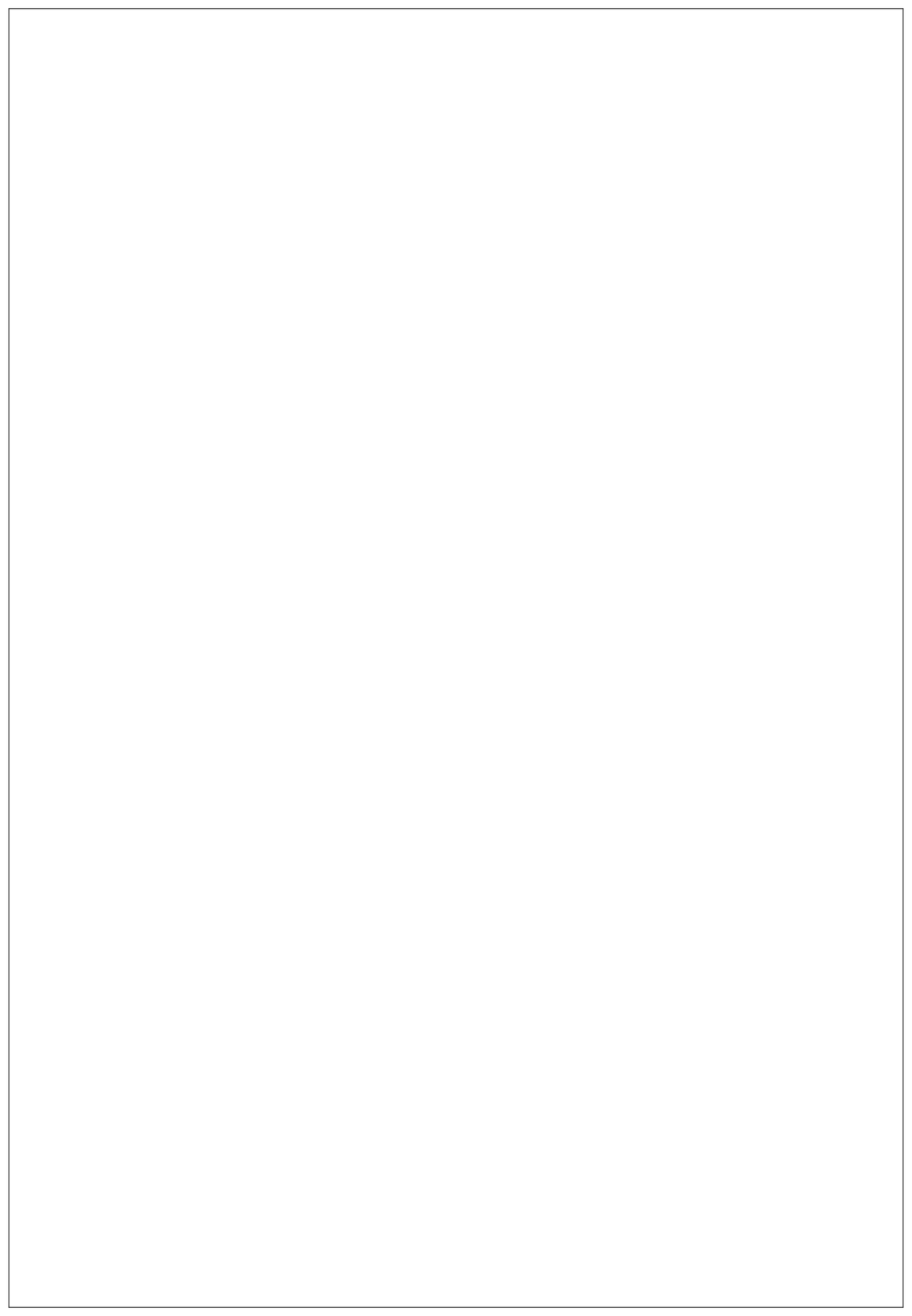
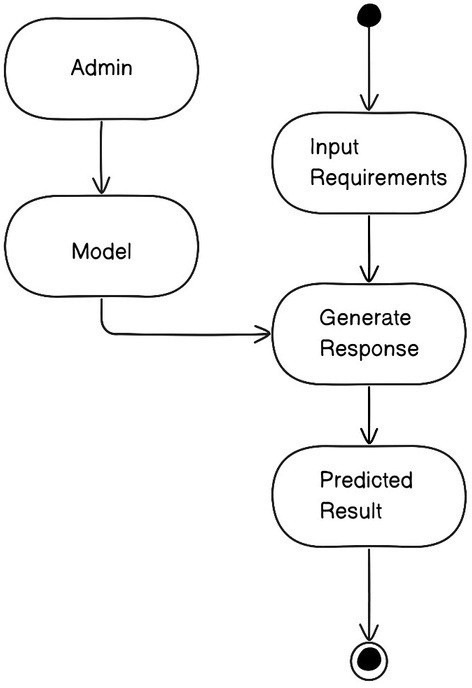
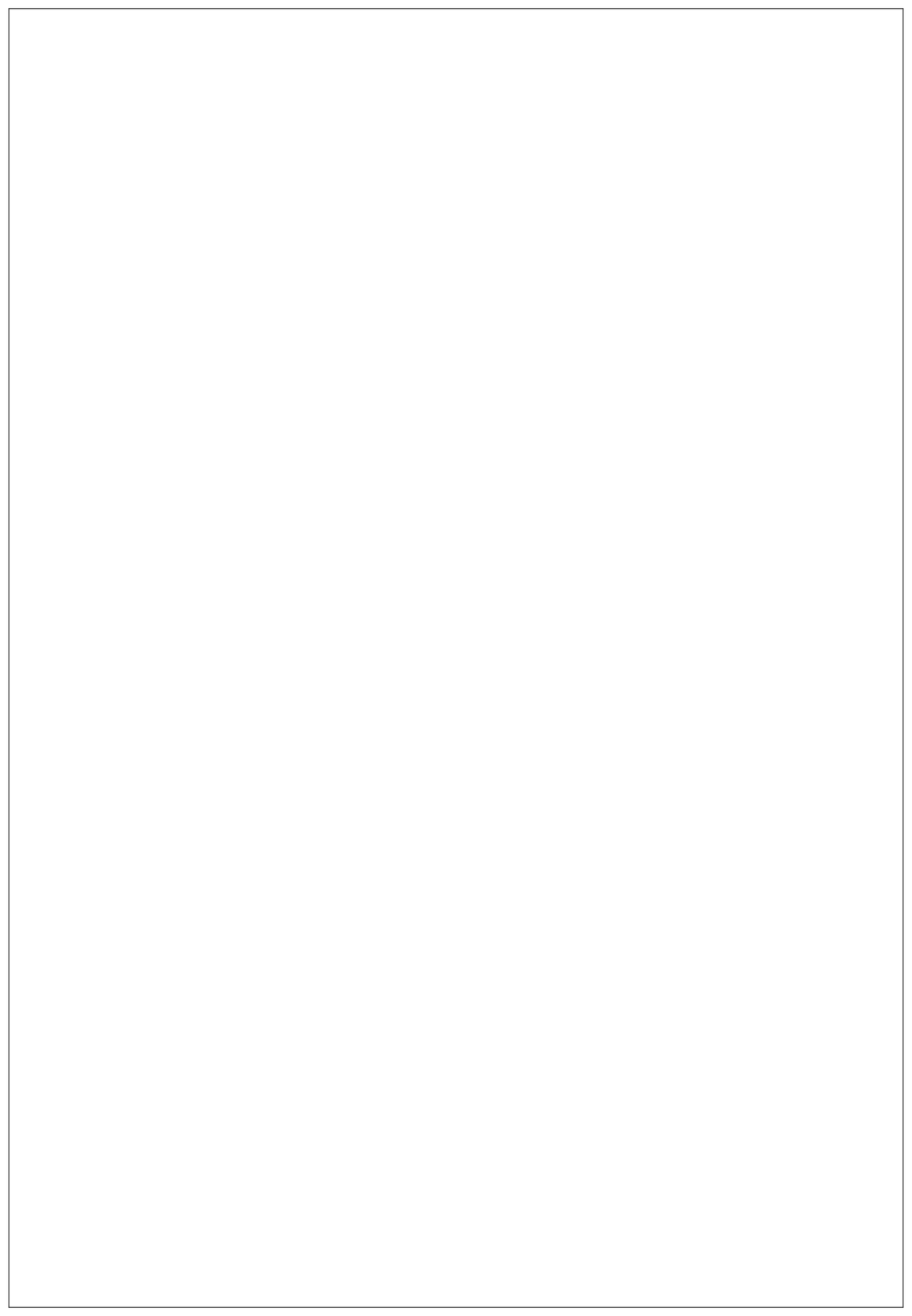
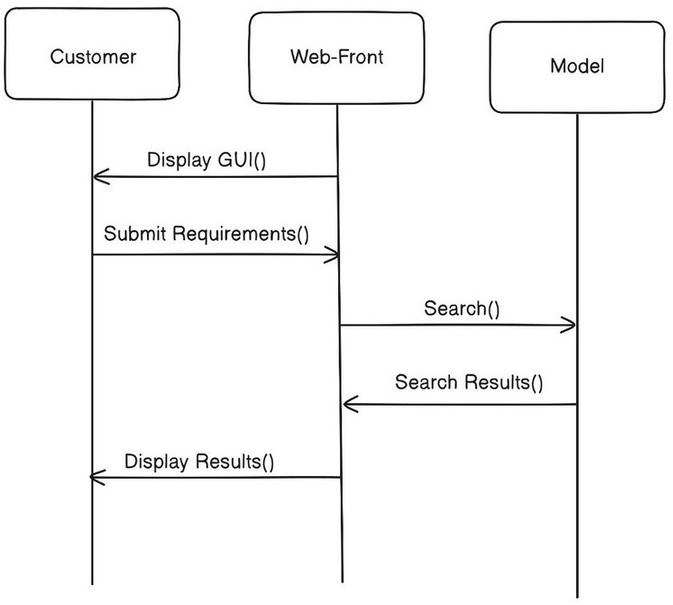
Figure 4.2: DFD Level 0 Diagram



* + 1. **DFD Level 1 Diagram**

The data flow diagram (DFD) is a crucial modeling tool that helps represent various system components. These components typically include the system process itself, the data utilized by that process, external entities that interact with the system, and the pathways through which information flows within the system. DFDs provide a clear visual representation of how data moves through a system and how different elements interact with each other, aiding in system analysis and design.

Figure 4.3: DFD Multi Level Diagram



* 1. **UML DIAGRAMS**
     1. **Activity diagram**

Activity diagrams are visual representations that depict workflows involving step-by-step activities and actions. They support various programming constructs like choice, iteration, and concurrency. In the Unified Modeling Language (UML), activity diagrams serve to model both computational processes and organizational workflows, showcasing the flow of control within a system. These diagrams are constructed using a set of specific shapes connected by arrows to illustrate the sequence and dependencies of activities and actions, making them a powerful tool for system analysis and

design.

Figure 4.4: Activity Diagram

* + 1. **Sequence Diagram**

A sequence diagram is an interaction diagram that illustrates the interactions and order of operations between processes. It is based on the concept of a message sequence chart. The diagram visually represents the sequence of events and object interactions over time. It specifically shows the objects and classes involved in a scenario and the messages exchanged between them to achieve the desired functionality within that scenario.

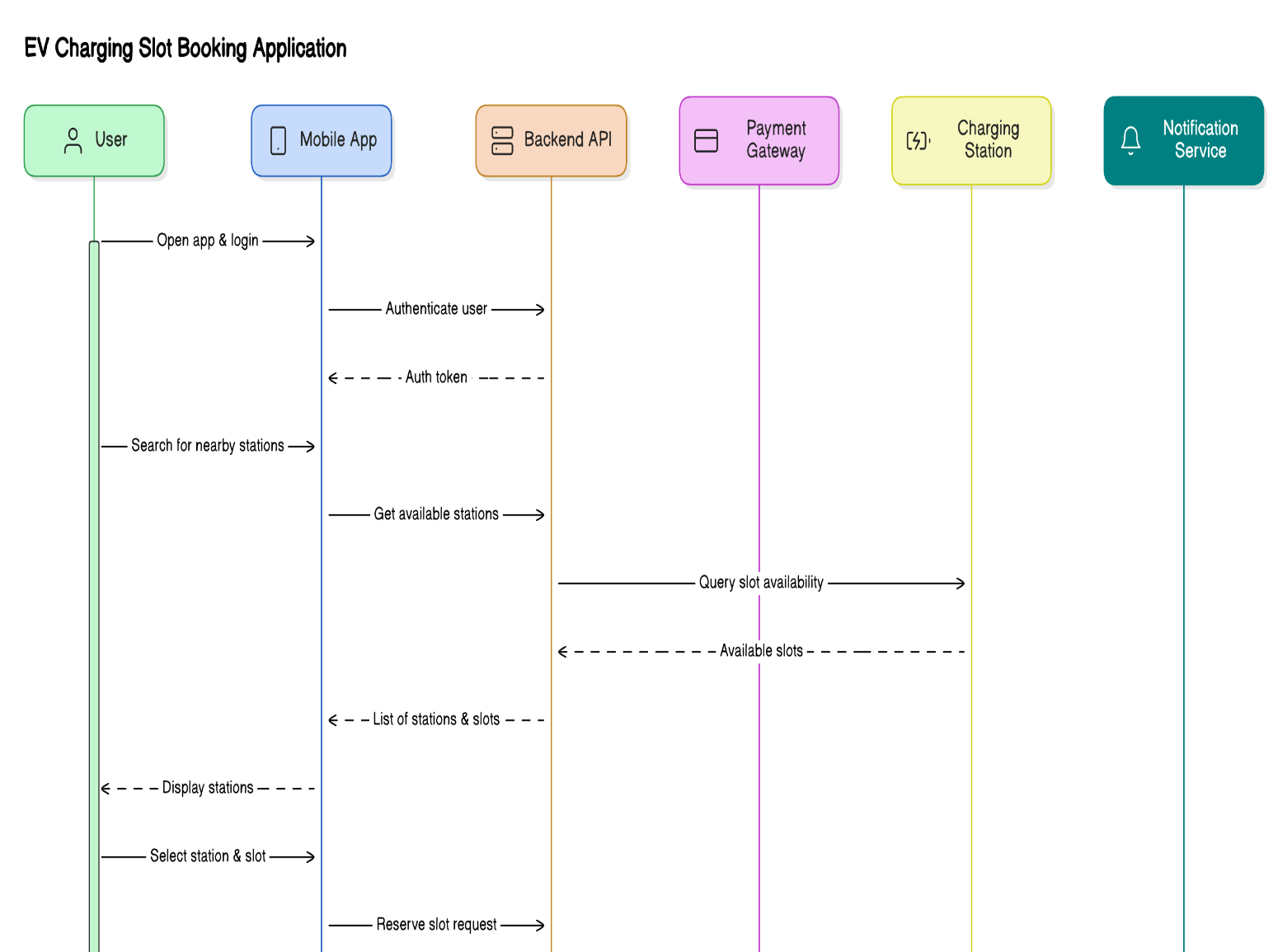
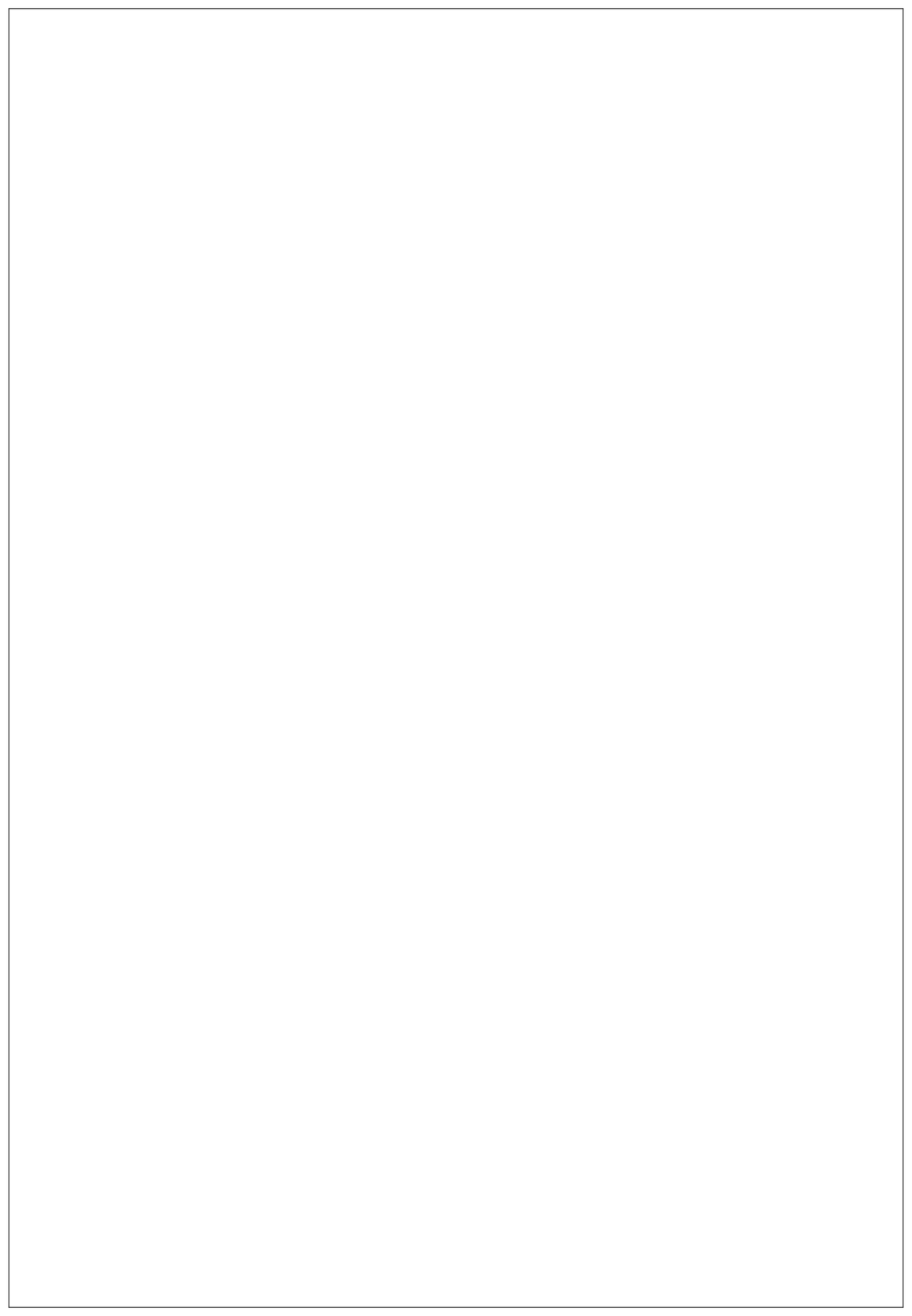
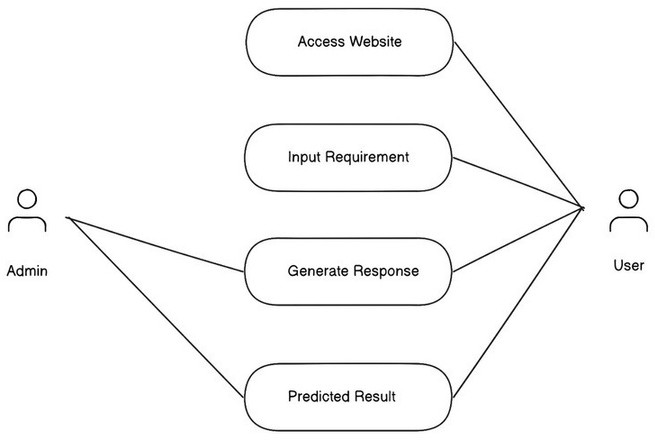


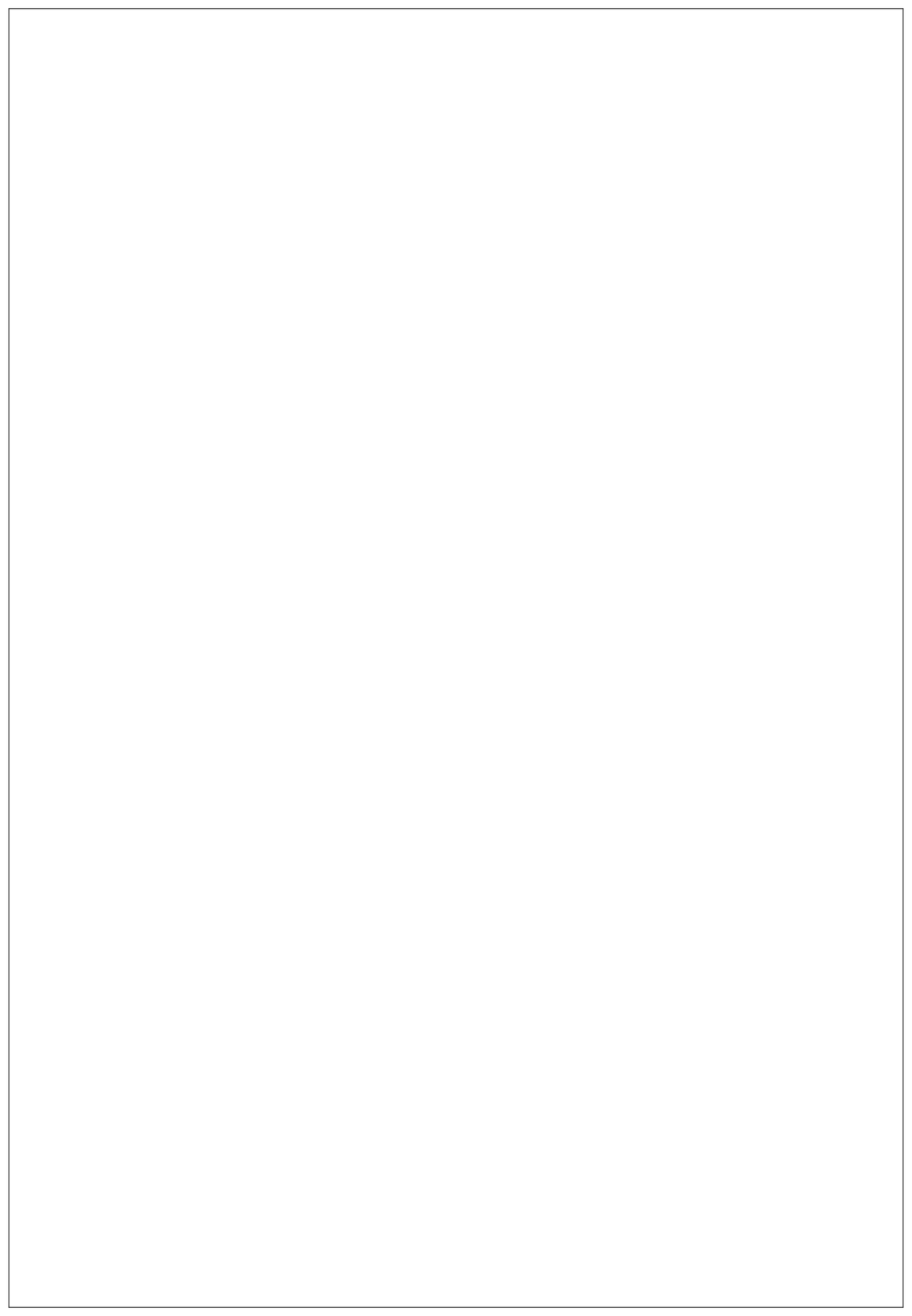
Figure 4.5: Sequence Diagram



* + 1. **Use case Diagram**

Figure 4.6: Use case Diagram

**4.3.4 Class Diagram**



The class diagram is a static representation in Unified Modeling Language (UML), providing a snapshot of an

application's structure. It offers a visual depiction of the system's static view, focusing on the classes, their

attributes, operations, and the relationships between classes. While primarily used for visualizing and documenting

system aspects, the class diagram also plays a crucial role in software development

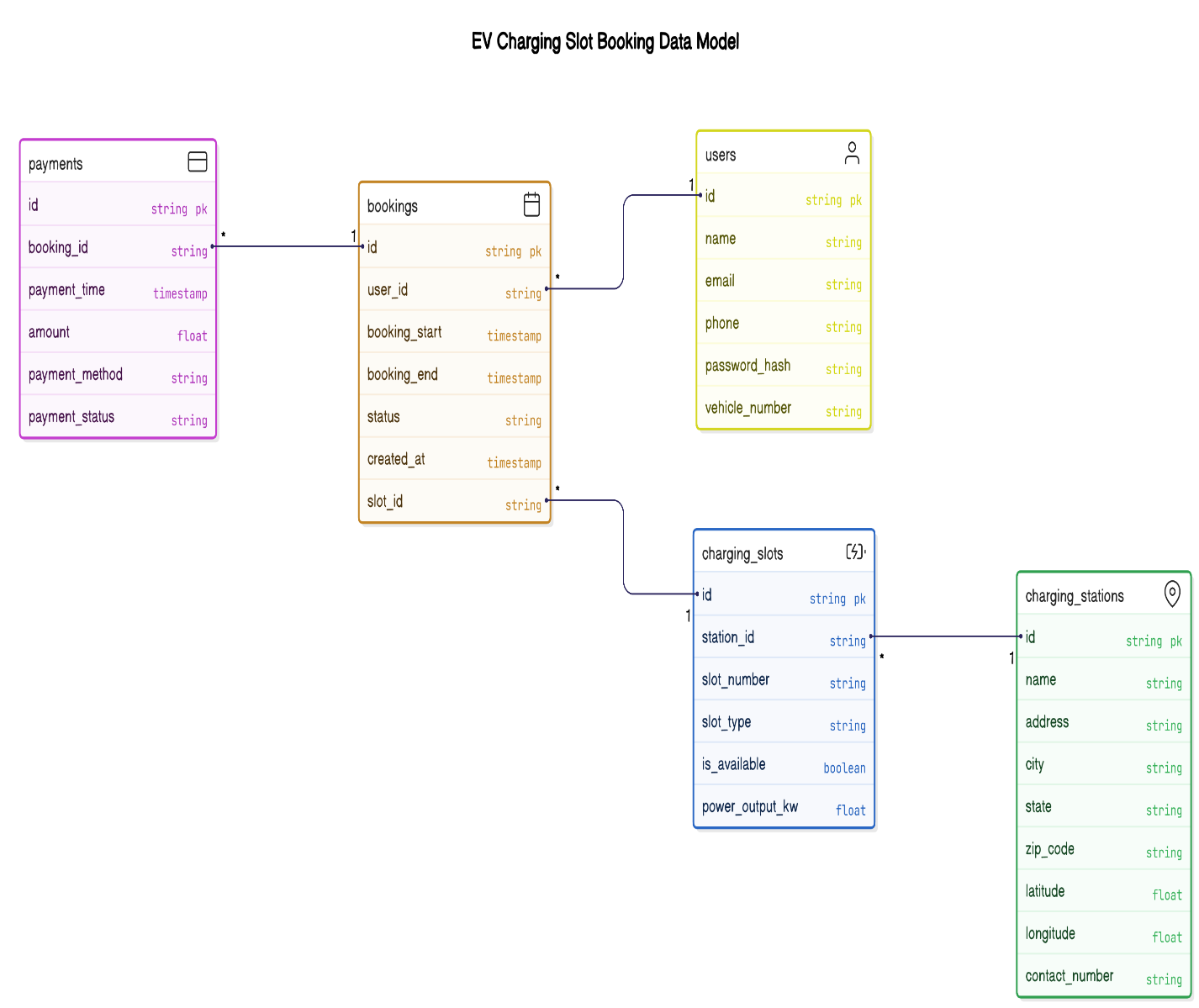
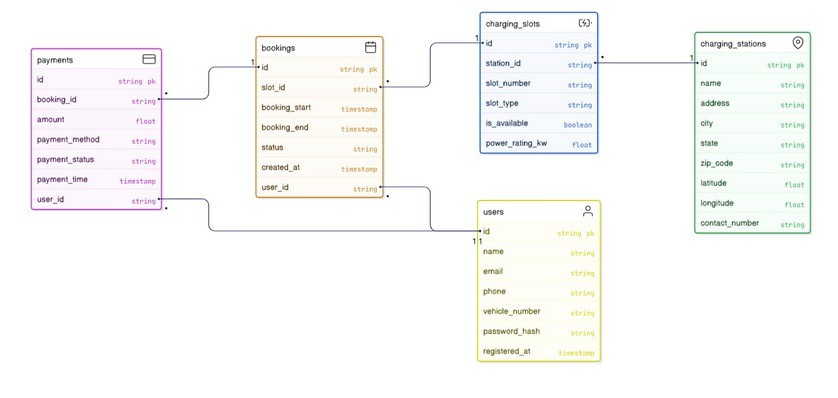
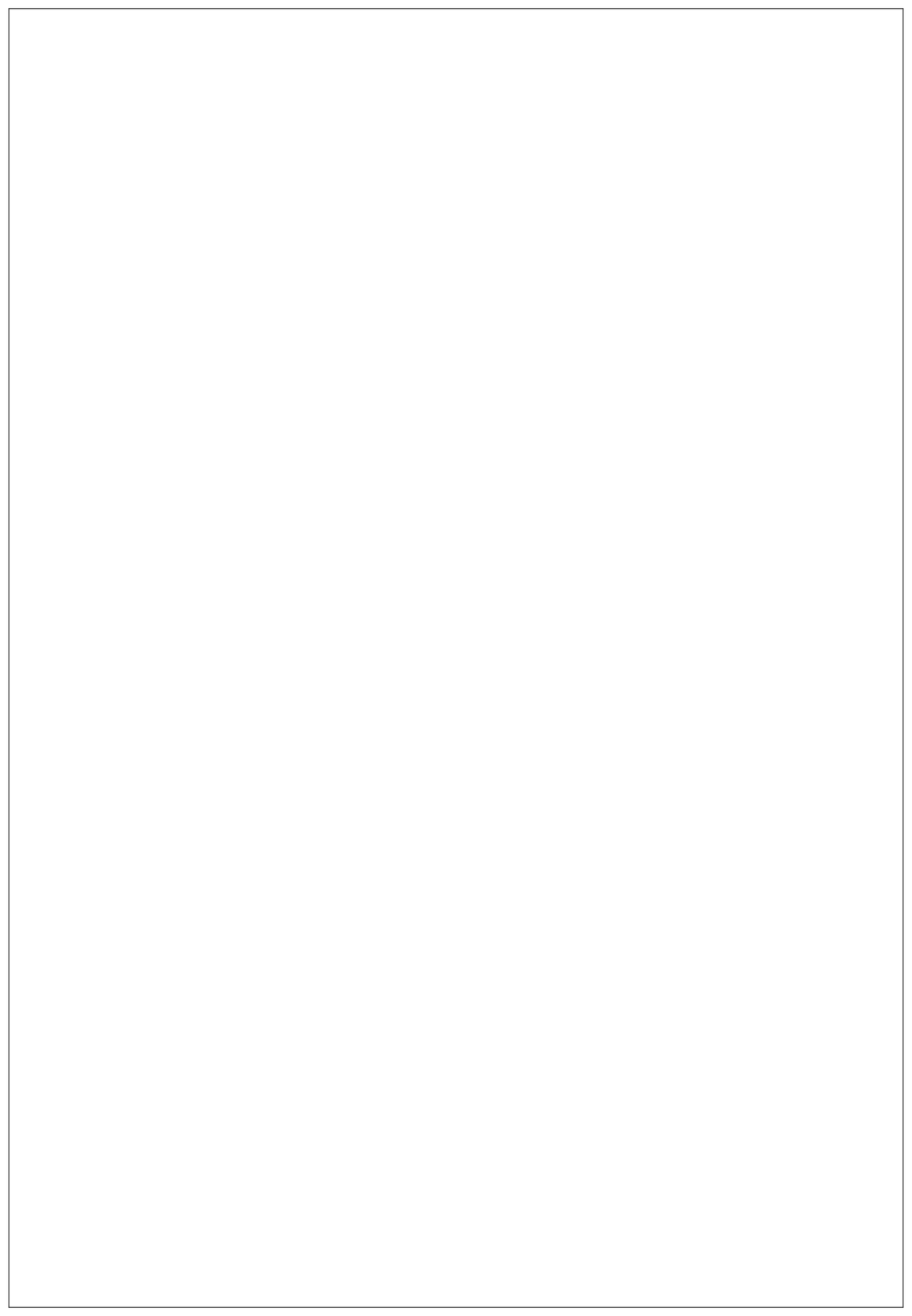
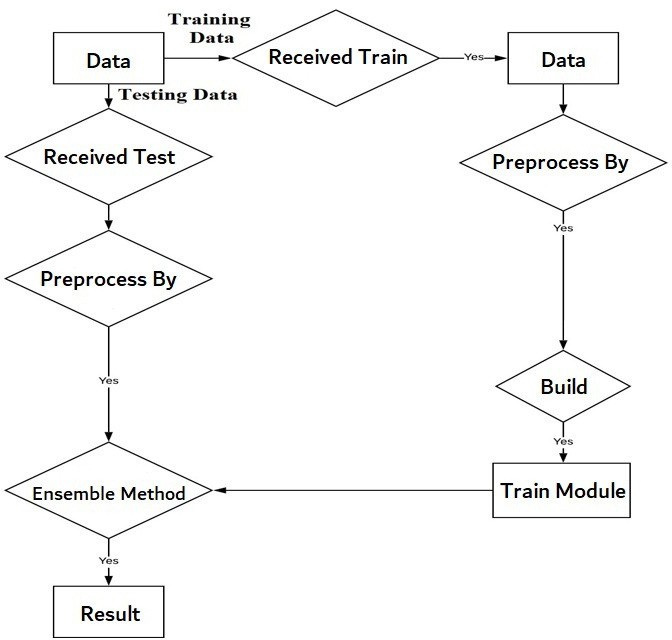
.

Figure 4.7: Class Diagram



* + 1. **ENTITY RELATIONSHIP DIAGRAMS**

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this

context is an object, a component of data. An entity set is a collection of similar entities. These entities

can have attributes that define its properties.

Figure 4.8: ER Diagram

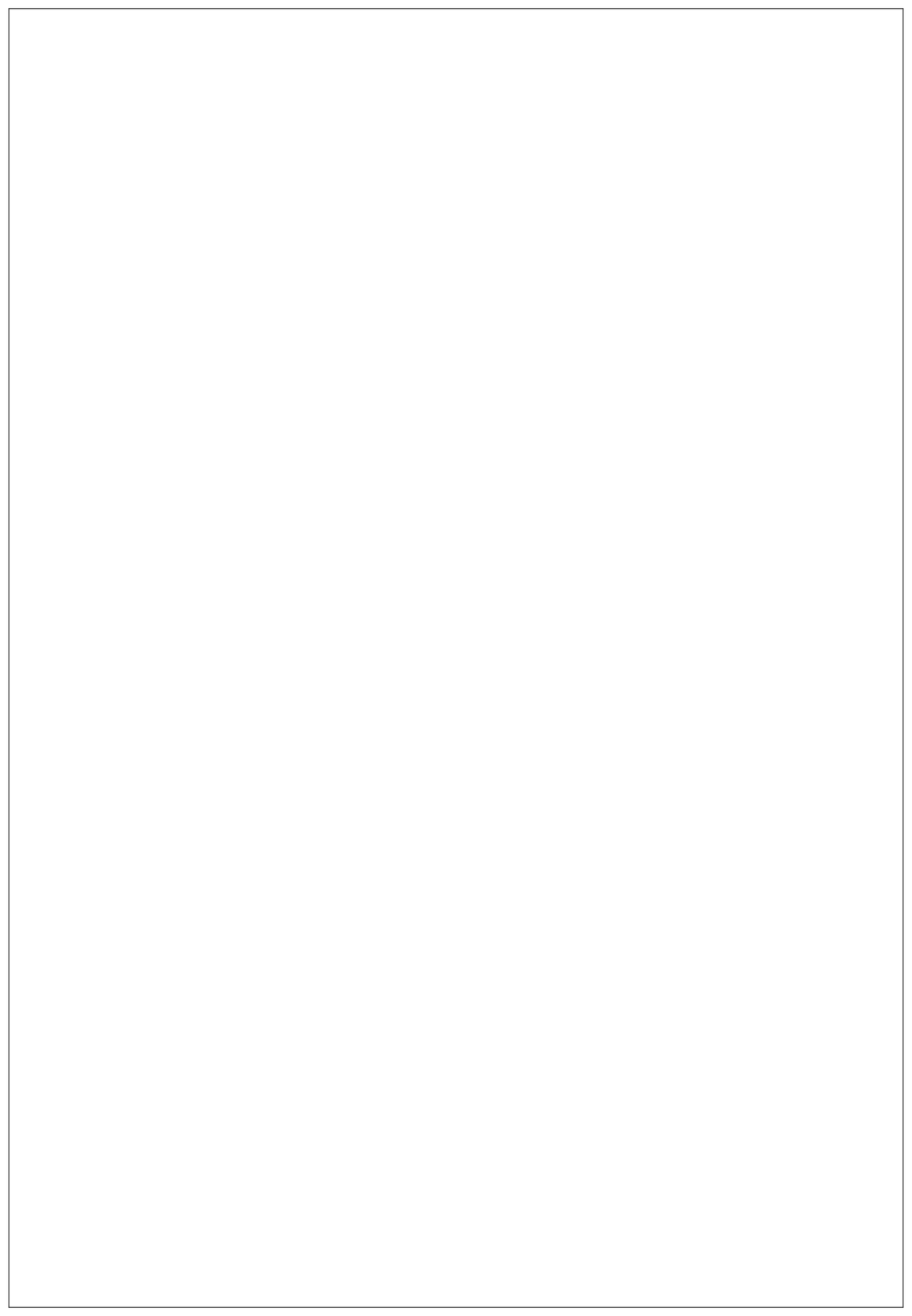
* + 1. **MATHEMATICAL MODEL**
* Inputs (I): The input to the system includes user data (such as user ID, location), charging station data station ID, availability, slot timing), booking requests, and payment details.
* Preprocessing and Normalization (P/N): This step involves validating and formatting the input data. It includes checking for valid user input, formatting timestamps, verifying charging station availability.
* Slot Booking Module (T1): This module processes slot booking requests by matching user preferences with available charging stations and time slots. It handles booking logic, updates slot status, and prevents double-booking.
* Payment and Confirmation Module (T2): This module processes secure payments and confirms the booking.
* Analysis and Reporting (A): This component analyzes booking trends, user activity, and system performance.

The mathematical representation can be summarized as:

S = {I, P/N, T1, T2, A}

Where:

* I= Input data
* P/N= Processing and Normalization of input data
* T1= Slot booking module
* T2= Payment and confirmation module
* A= Analysis and Reporting

**

# CHAPTER 5:

# IMPLEMENTATION DETAILS

* 1. **DESCRIPTION OF TOOLS**

**VS Code**

Visual Studio Code, often shortened to VS Code, is a free and open-source code editor popular among developers. It offers a user-friendly interface with features like syntax highlighting, code completion, and debugging tools, making it suitable for various programming languages. VS Code goes beyond basic editing by allowing customization through themes and extensions, letting developers tailor the experience to their specific needs. This flexibility, coupled with its lightweight design, makes VS Code a powerful and versatile tool for building modern software applications.

**Jupyter Notebook**

Jupyter Notebook is a web-based application designed for interactive data analysis. It combines code execution, rich text explanations, and visualizations into a single document. Imagine a notebook where you can write Python code (or other languages), see the results right away, and add explanations and visualizations alongside your code. This makes it ideal for data exploration, prototyping machine learning models, and creating data science reports that are easy to understand and share.

* 1. **PROGRAMMING LANGUAGE DESCRIPTION**

**Mysql:**

MySQL is an open-source relational database management system (RDBMS) that uses Structured Query Language (SQL) to manage and manipulate data. It is widely used for storing data in web applications and supports features like transactions, indexing, and data security. MySQL is known for its speed, reliability, and ease of use, making it a popular choice for both small and large-scale applications.

**Django:**

Django is a high-level, open-source web framework written in Python that enables rapid development of secure and maintainable websites. It follows the Model- View- Template (MVT) architectural pattern and includes built-in features like authentication, admin interface, and ORM for database operations.

**Python:**

Python is a versatile and beginner-friendly programming language known for its easy- to-read syntax. This makes it ideal for beginners but powerful enough for tasks like web development, data analysis, and machine learning. As an interpreted language, Python simplifies the development process by eliminating the need for compilation before running the code.

* 1. **IMPLEMENTATION DETAILS**

The implementation of the project involves the following steps: 1. Data Collection: Real estate data is collected from various sources containing information about

houses including location, square footage, number of bathrooms, bedrooms, balconies, area type, and availability.

* + 1. Data Preprocessing:

The collected data is cleaned and preprocessed to handle missing values, encode categorical variables, and scale numerical features.

* + 1. Model Training:

A machine learning model is trained using the preprocessed data. Various algorithms like Linear Regression, Random Forest, or Gradient Boosting are used.

* + 1. Web Application Development:

A web application is developed using Python for the frontend, and Django for the backend. The application allows users to input the features of a house and get an estimated price prediction.

* + 1. Deployment:

The trained model is deployed on the Django server, and the web application is deployed on a hosting service like Heroku or AWS.

* 1. **ALGORITHM DETAILS**

The Electric Vehicle Charging Slot Booking Application follows a structured algorithm to manage

user activities efficiently. First, the system handles user registration and login. During registration,

the application collects user information such as name, email, phone number, and password.

After validating the input fields to ensure correct data entry, the user information is stored securely

in the MySQL database. For login, the system requires the user’s email and password, which

are verified against stored records. If the credentials match, access is granted; otherwise, an appropriate error message is displayed.

Once logged in, the user can find available charging stations. The user selects or enters the desired

city or location, upon which the application queries the database for all active charging stations

within that area. The system then displays a list of matching stations along with relevant details

like station name, address, and number of available slots.

To book a charging slot, the user selects a station from the displayed list. The application then

shows the available time slots for that station. The user selects a preferred date and time slot.

Before confirming the booking, the system checks the database to ensure the slot is still available.

If the slot is free, the booking is processed and stored; otherwise, the user is asked to select

another available slot. After a successful booking, the database is updated to reflect that the slot is

no longer available for others.

**5.5 OUTPUT OF IMPLEMENTATION**

Our system utilizes ensemble learning models to predict house rates accurately. These models analyze various

factors such as location and size to provide reliable predictions. The interface is designed to be user-friendly,

offering a simple and clear presentation of the predicted house rates. This aids users in making informed

decisions based on data-driven insights. By leveraging ensemble learning, we ensure a robust and effective

approach to house rate prediction.

(In the form of Screenshots)

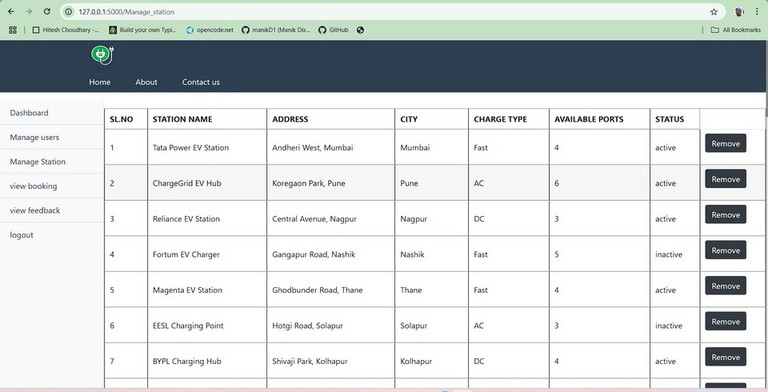
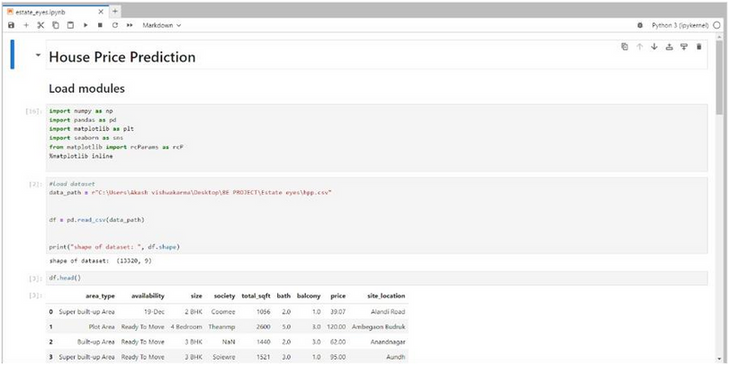


Figure 5.1: Slot allocation

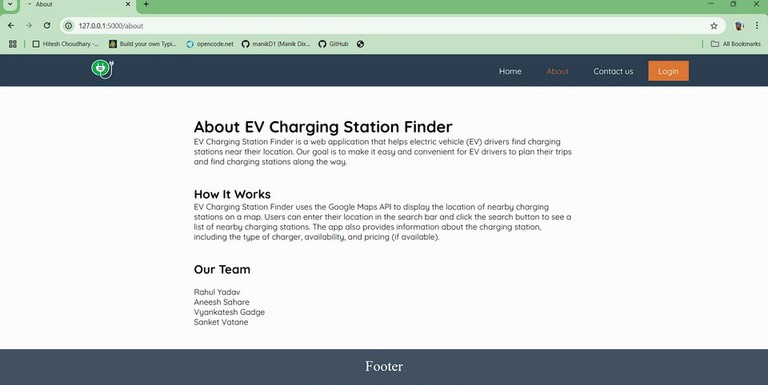


Figure 5.2: About us

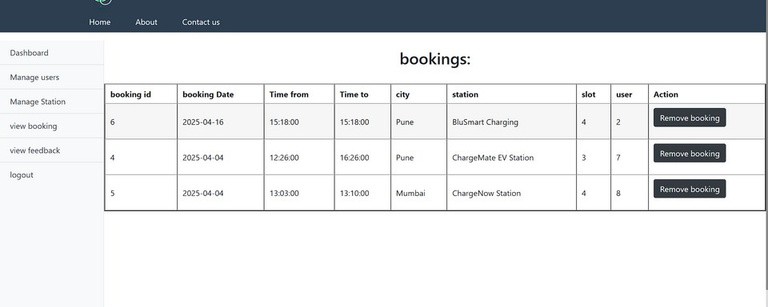


Figure 5.3: Database

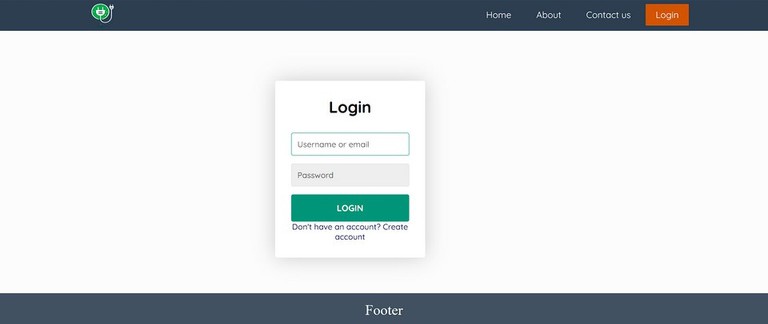


Figure 5.4: Login

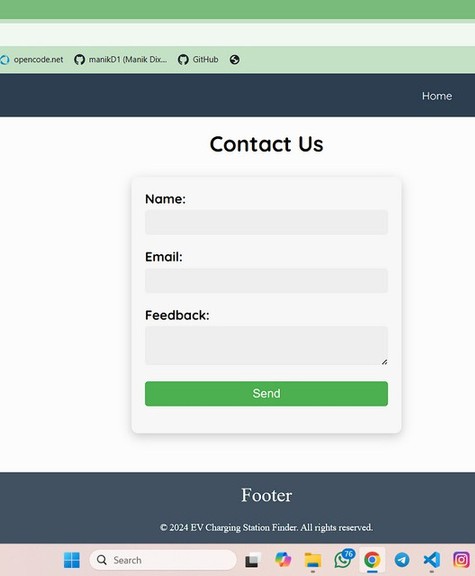


Figure 5.5: Contact page

# CHAPTER 6 :

# TESTING

* 1. **INTRODUCTION**

Testing is very important phase of the software development life cycle. The purpose of this phase is to check lifetime system. This is compulsory phase. Information given in this section gives the details for the testing activities that should be approved for the “opinion mining” of scheme. Tester has to estimate test of each component and write downs test cases according to user requirement and system structure.

**Principle of Testing:**

* + - To know the system performance.
    - To recognize the functionality of each and every individual module.
    - To verify whether system is functioning as per the user requirements.
  1. **TYPE OF TESTING**

1. **White-box testing:**

The White box testing is done by the tester who has knowledge of the programming language.

White box testing done on algorithm or source code of the project. It is the procedure of giving

the input to the project and verifying that how system process input to produce result. In white

box testing all, the interior details are required to known to tester. White box testing is also

called as transparent testing. This test needs code to check so it is essential for tester to have the

knowledge coding

Following are the techniques of White Box testing:

* + Programming style
  + Control method
  + Source language
  + Database design

This type of a test is useful to beat defects at structural level. This test goes lower the top or functional layer to expose defects. Test case designing methods:

* Statement coverage
* Decision coverage
* Condition coverage
* Multiple Condition coverage
* Path coverage

1. **Black box Testing:**

This type of testing takes place by actual validating of requirement with actual result. In the black box testing tester does not require knowing the internal logic of the project. He concern with the actual result generated by the system. Functional testing is performed in black box testing. Here, the knowledge about how the program internally executes or the programming language does not required. According test plan, Following are the functionality which cover under the black box testing.

* + Server connection
  + Data generation
  + Data extraction
  + Machine learning algorithm process
  + Prediction
  + Results

1. **Unit testing:**

Unit testing is small part of a system to check it might be as methods, functions, classes of code, interface and of system. Therefore here, tester will test every small unit of the system to investigate whether the module is suitable for the system. Software writes all units tests and carried out to verify that code complete necessities, design and perform as per user requirement. Unit testing cover a few advantages like those that error and bugs found at early stage. Because of the issues found at very early stage and determined instantly is not disturbing the other part of codes.

1. **Integration testing:** In the Integration testing, different modules has combine together and tested. To exchange information easily between distinct modules of the system, test that it performance as per the given requirement. When all testing allied work is completed, the software is deployed to the customer

**5.Validation testing:** In this testing, tester will verify the software that it covers all the requirements as per the system requirement specification. It makes sure that the software was at correct place. It also verifies whether we have built right system or not. It checks the following:

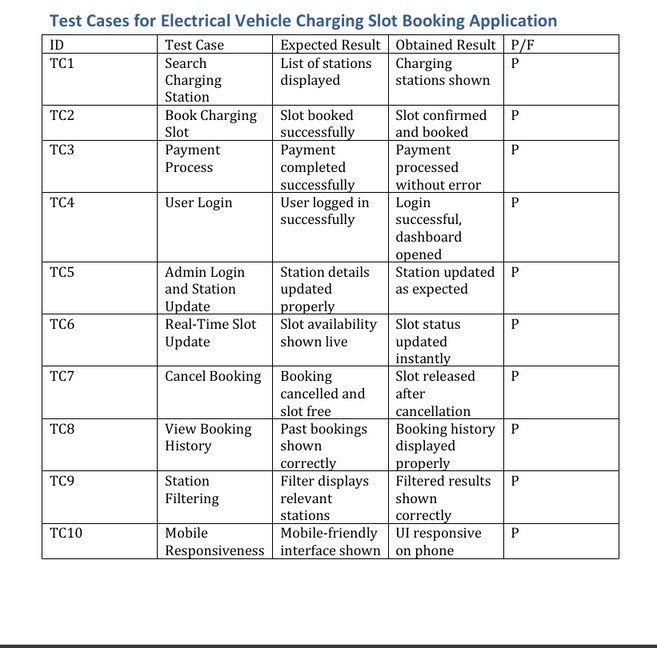
* + It justifies the execution and behavior of the system.
  + All probable input data given as input and capture projected output. Test log is used for deployment

**6.System Testing:** After performing the integration testing, the next step is output testing of the proposed system. No system could be useful if it does not produce the required output in a specified format. The outputs generated are displayed by the user. Here the output format is considered in PPT format document.

* 1. **TEST CASES AND RESULTS**

After implementation section while tester assessments code it detects the a few faults or disorder inside the code. The faults corrected through a few methods in short time. While testing the performed by means of creating the test instances. There are person test cases performed for every state of affairs, and it tested with the anticipated output by way of system or software. The following table indicates that everyone the check cases which might be vital for project.

Below Table shows the suite of test cases which are executed and passed.



# CHAPTER 7: CONCLUSION

* 1. **CONCLUSION**

The successful development of the Electric Vehicle Charging Slot Booking Application marks a significant step toward solving one of the key challenges in the electric mobility sector: inefficient and unpredictable access to charging infrastructure. By providing users with a convenient way to search, locate, and reserve charging stations based on real-time data, the application not only saves time but also enhances the reliability of EV usage for daily commuters.

From a user perspective, the platform ensures greater transparency, less waiting time, and the ability to make informed decisions regarding where and when to charge. For charging station operators, the system provides tools to streamline station traffic, manage bookings efficiently, and maintain updated status records. The backend architecture is designed for scalability and real-time synchronization, ensuring seamless interaction between users and administrators.

Overall, this system is a meaningful contribution to the growing ecosystem of electric mobility and sets a foundation for future enhancements and real-world deployment.

* 1. **FUTURE SCOPE**

As the adoption of electric vehicles accelerates, there is immense potential to expand and enhance the capabilities of our charging slot booking system. In the future, the application can be integrated with **renewable energy-powered charging stations**, such as those using solar or wind energy. This would reduce carbon dependency and further promote sustainable transportation.

Another important area of development is the **use of artificial intelligence** to predict user demand patterns. AI algorithms can be implemented to analyze user behavior, station traffic, and booking trends to suggest optimal time slots and balance peak load periods more effectively.

The system could also incorporate **dynamic pricing models**, where rates change based on demand, time of day, or energy source availability. This would not only optimize the utilization of charging stations but also encourage users to charge during off-peak hours.

# CHAPTER 8 APPENDIX A

* 1. **JOURNAL PAPER**
     1. Paper Title:

Electrical vehicle charging slot booking application

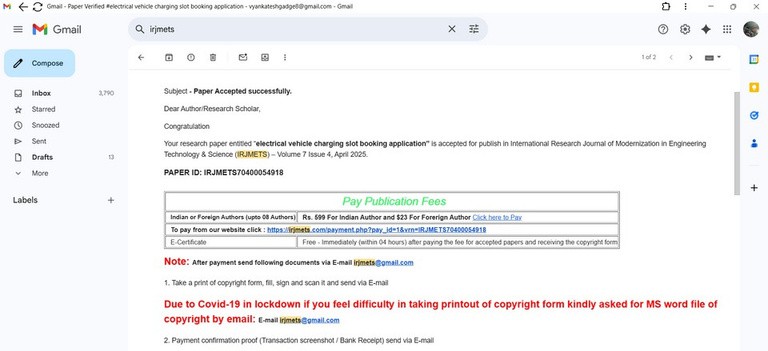
* + 1. Name of the Conference/Journal where Paper Submitted:

International Research Journal of Modernization in Engineering Technology and Science

* + 1. Paper Status:

Published

* + 1. Review comments by reviewer:



No.

* + 1. Corrective actions if any:

No.

Figure 8.1: Acceptance Letter

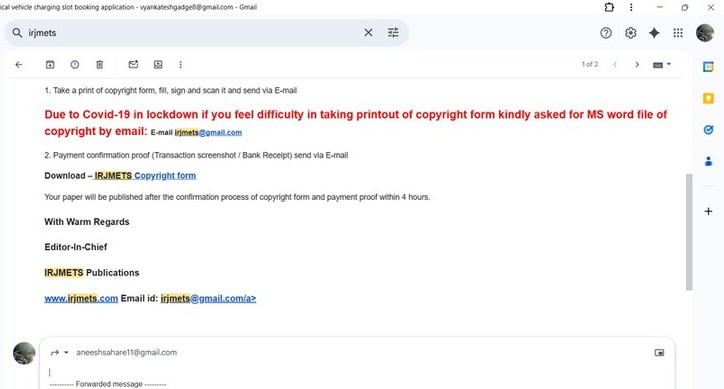
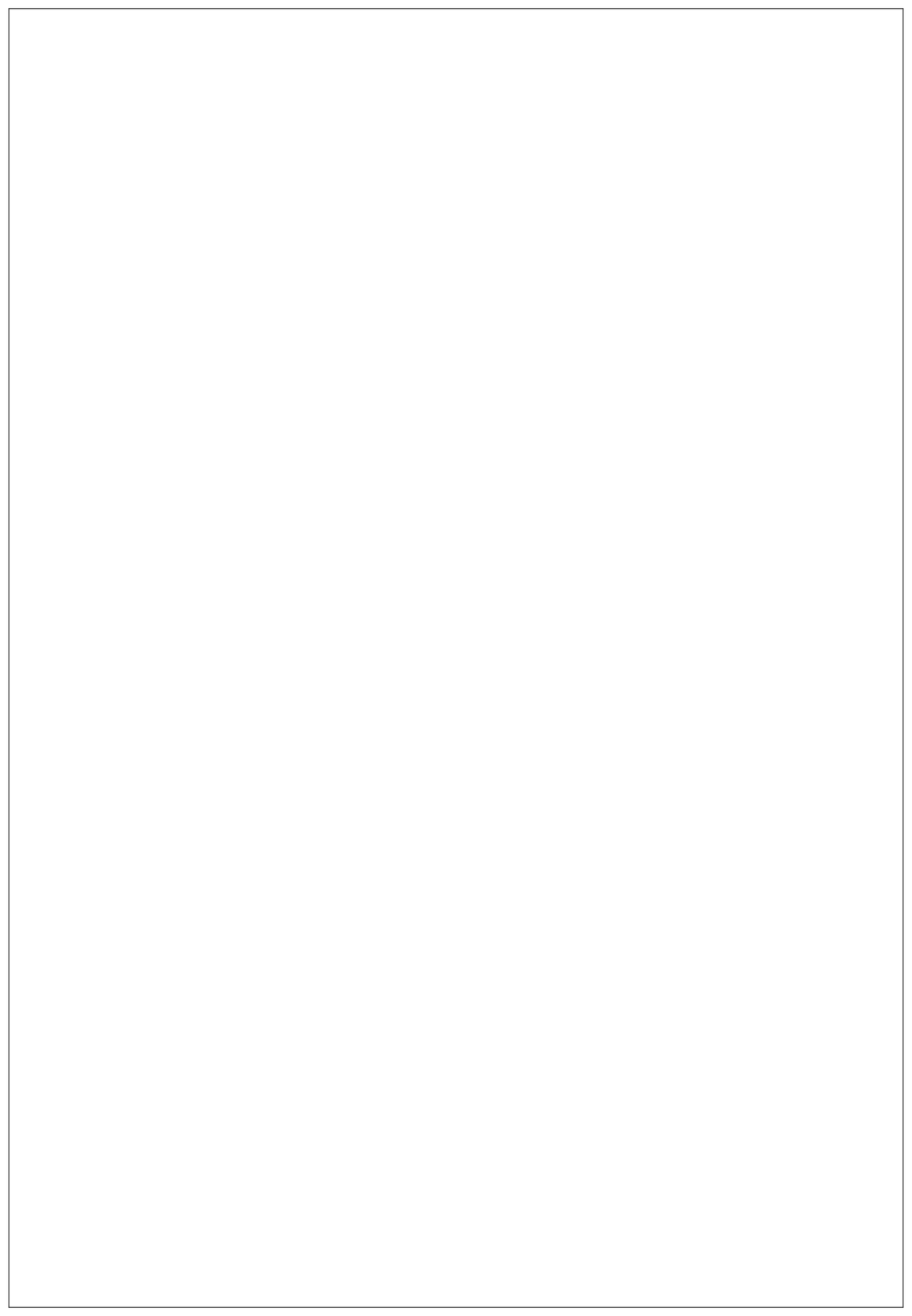


Figure 8.2: acceptance letter

# CHAPTER 9: APPENDIX B

**9.1 PLAGIARISM REPORT OF PROJECT REPORT**

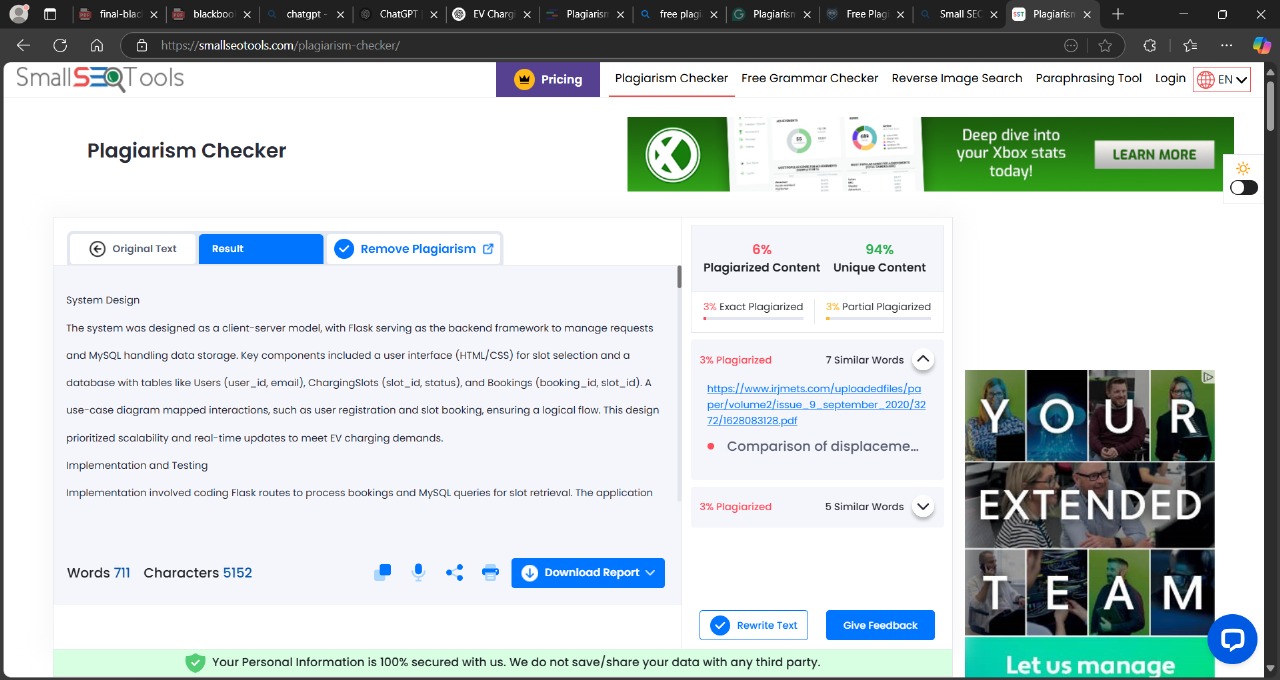


Figure 9.1: Plagiarism Report

**CHAPTER 10: APPENDIX C**

* 1. **COPYRIGHT FORM**

****

# CHAPTER 10:

# REFRENCES

1. C. Zhang, Y. Wu, Y. Li, and M. Zhang, "Smart Electric Vehicle Charging System Based on Internet of Things," IEEE Access, vol. 7, pp. 134654–134666, October 2019.
2. H. Shareef, M. S. Ahmed, A. Mohamed, and E. Al Hassan, "Review on Home Energy Management System Considering Demand Responses, Smart Technologies, and Intelligent Controllers," IEEE Access, vol. 6, pp. 24498– 24509, April 2018.

[2] H. Shareef, M. S. Ahmed, A. Mohamed, and E. Al Hassan, "Review on Home

Energy Management System Considering Demand Responses, Smart Technologies, and Intelligent Controllers," IEEE Access, vol. 6, pp. 24498–24509, April 2018.

1. A. Ghosh, S. Sen, A. Sinha, and B. Mukherjee, "IoT-Enabled Smart Charging Station for Electric Vehicle in a Smart City," IEEE Transactions on Industrial Informatics, vol. 17, no. 5, pp. 3375–3384, May 2021.
2. Y. Wang, M. Peng, Y. Lu, and Y. Liang, "Optimal Scheduling Strategy for Electric Vehicle Charging Stations Based on User Priority and Renewable Energy," IEEE Transactions on Smart Grid, vol. 11, no. 4, pp. 3306–3316, July 2020.
3. L. Chen, Y. He, and X. Huang, "An Intelligent Charging Strategy for Electric

Vehicle Stations Considering Dynamic Pricing," IEEE Transactions on Industrial Electronics, vol. 67, no. 8, pp. 6500–6509, August 2020.

1. H. Li, Y. Xu, and Z. Liu, "Optimization of Electric Vehicle Charging Stations in Smart Grid Using Machine Learning Algorithms," IEEE Transactions on Power Systems, vol. 35, no. 2, pp. 1241–1252, February 2020.