Electric Vehicles Recharge Nearest Bunk Using Mobile Application

Dissertation

Submitted in partial fulfillment of the

Requirements for the award of the Degree of

MASTER OF COMPUTER APPLICATIONS

in

COMPUTER SCIENCE & ENGINEERING

By

NEELURU BABU

(18001F0018)

Under the guidance of

Dr. K. F. Bharati **B. Tech, M. Tech., Ph. D.**

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, ANANTAPUR

COLLEGE OF ENGINEERING (Autonomous)

ANANTHAPURAMU-515002

ANDHRA PRADESH

2021

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

COLLEGE OF ENGINEERING

(Autonomous)

ANANTHAPURAMU-515002

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled, **"Electric Vehicles Recharge Nearest Bunk Using Mobile Application "**, is bonafide work of bearing **NEELURU BABU** Admn. No:18001F0018 submitted to the faculty of Computer Science & Engineering, in partial fulfillment of the requirements for the award of **MASTER OF COMPUTER APPLICATIONS** from Jawaharlal Nehru Technological University Anantapur, College of Engineering (Autonomous), Ananthapuramu.

Signature of the Supervisor

Dr. K F BHARATI , M.Tech, Ph.D.

Associate Professor

Dept. of Computer Science& Engg.

JNTUA College of Engineering

Ananthapuramu-515002.

Signature of the Head of the department

Dr. K. MADHAVI, M.Tech, Ph.D.

Associate Professor & HOD

Dept. of Computer Science & Engg

JNTUA College of Engineering

Ananthapuramu-515002.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

COLLEGE OF ENGINEERING

(Autonomous)

ANANTHAPURAMU-515002

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DECLARATION

I am **NEELURU BABU**, bearing Admn no. **18001F0018**, hereby declare that the project report entitled "**Electric Vehicle Recharge Nearest Bunk Using Mobile Application**" under the guidance of **DR. K. F. BHARATI** $_{\textbf{M.Tech , Ph,D}}$**, Associate Professor**, JNTUACEA Ananthapuramu is submitted in partial fulfillment of the requirements for the award of the degree of **MASTER OF COMPUTER APPLICATIONS** in COMPUTER SCIENCE AND ENGINEERING.

This is a record of bonafied word carried out by me and the results embodied in this project have not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

NEELURU BABU

(18001F0018)

ACKNOWLEDGMENTS

The satisfaction and exhilaration that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

The first person I would like to thank is my guide **Dr K F Bharati,**$_{\text{MTech,Ph.D}}$. Her wide knowledge and logical way of thinking have made a deep impression on me. Her understanding, encouragement and personal guidance have provided the basis for this thesis. She is a source of inspiration for innovative ideas and her support is known to all her students and colleagues.

I wish to thank **Prof. P.Sujatha,** Principal, JNTUA College of Engineering (Autonomous), Ananthapuramu, for providing excellent institution environment and all facilities in completing this project.

I wish to thank **Prof. B.Durga Prasad**, Vice Principal, JNTUA College of Engineering (Autonomous), Ananthapuramu, for providing excellent institution environment and all facilities in completing this project.

I wish to thank **Dr. K.Madhavi,** Associate Professor and Head of Computer Science and Engineering Department, JNTUA College of Engineering(Autonomous), Ananthapuramu. Her wide support, knowledge and enthusiastic encouragement have impressed me to better involvement into my project thesis and technical design also her ethical morals helped me to develop my personal and technical skills to deploy my project in success.

I wish to thank teaching staff and non-teaching staff of Computer Science and Engineering Department, JNTUA College of Engineering (Autonomous), Ananthapuramu, for their support to complete my project successfully.

NEELURU BABU

(18001F0018)

Abstract

Transportation electrification is one of the essential components in the future smart city planning and Electric Vehicles (EVs) will be integrated into the transportation system seamlessly. Charging stations are the main source of energy for EVs and their locations are critical to the accessibility of EVs in a city. They should be carefully situated so that an EV can access a charging station within its driving range and cruise around anywhere in the city upon being recharged.

In this project, we formulate the Electric Vehicle Charging Station Placement Problem, in which we minimize the total construction cost subject to the constraints for the charging station coverage and the convenience of the drivers for EV charging. In this Project examines various issues related to electric vehicle supply equipment (EVSE) or charging stations related to.

The proposed system of EV Charging mobile app to provide EV owner the convenience of locating charging stations on Google map, vacancy of charging slots, getting updates on charging, recommendations on time-of-day use. Help increase the life of batteries and ensure smooth journeys long distance.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Modern civilization relies heavily on fossil fuels to support construction, military protection, and people’s mobility. Due to the world’s shortage of fossil fuels, nations compete to secure enough reserves of natural resources for sustainability. Seeking alternative energy sources becomes crucial to a nation’s future development. One of the major sources of fossil fuel consumption is transportation. Most of our daily heavily demanded transportation, including buses and private cars, is powered by gasoline. A major consequence of burning fossil fuels is the release of tremendous amount of harmful gases, which partially constitutes the global warming effect and deteriorates people’s health. Electricity is considered as the most universal form of energy, which can be transformed from and to another form effectively. By converting the endurable renewable energy, like solar and wind, to electricity, we can manipulate energy in a much cleaner manner. Electrification of transportation like deployment of electric vehicles (EVs) can not only alleviate our demand on fossil fuels, but also foster a better environment for living. Therefore, EVs will become the major components in the future transportation system. Incorporating EVs into an existing self-contained transportation system is challenging. Solely expanding the population of EVs in a city without enough road connections and corresponding charging and parking infrastructure will suppress the practicability of EVs due to their limiting moving ranges. Conversely, constructing the facilities with low utilization will result in a waste of resources. Moreover, existing gas stations are primarily designed for gas refuelling; combining charging infrastructure with the conventional gas stations may not be appropriate as the relatively longer charging process will saturate the limited space of the gas stations. We need to carefully plan EV charging facilities to modernize our cities. To be precise, we study how EVs will be integrated into the transportation system seamlessly and this will help make our cities “smart”. To do this, we study the Electric Vehicle Charging Station Placement Problem (EVCSPP) by finding the best locations to construct charging stations in a city. Technology advances rapidly while the smart city plan should take a much longer timespan. The plan should cater for the residents. Hence, we focus on the long-term human aspects rather than the technological ones. An EV can always access a charging station within its capacity anywhere in the city.

1.1 Objectives

It consumes less amount of time when compared to manual paper work through the automated system. The system will take care of all the servicing activity in a quick manner. Data storing is easier. Various Electric industries are becoming very careful about manufacturing & distribution of their products. New technology addresses these requirements, providing the foundation to allow cooperative interaction to be developed. Thus, the unmanned Electric recharge bunk using gsm is an example of new technology which will be providing the base for security of product distribution & data keeping using database. As the project is PC controlled, the project will be connected to one of the PC ports &programming languages like android application.

1.2 System Specifications

Hardware Requirements: -

Processor	: Intel(R) 2.10GHz
Installed memory (RAM)	: 4 GB
Hard Disk	: 160 GB
Operating System	: Windows (7)

Software Requirements: -

Front End	: HTML5, CSS3, Bootstrap
Back End	: PHP, MYSQL
Control End	: Angular Java Script

Android Tools:

Android Emulator

xampp-win32-5.5.19-0-VC11

Android SDK - adt-bundle-windows-x86

IDE: Eclipse Mars

jdk-8u66-windows-i586

CHAPTER 2

LITERATURE REVIEW

Nowadays, Electric Vehicles (EVs) are considered as a solution to greenhouse gas reduction and cost reductions in transportation. In developing countries, small vehicles such as three-wheelers are a common public transportation method which is efficient and effective because of less road space occupation. Especially, small EVs have many advantages over conventional vehicles regarding efficiency and maintenance. On the other hand, small EVs are predicted to gain a huge market in the future. Therefore, if it is possible to increase the utilization of small EVs for short distance journeys, within the town areas, the goal of reducing traffic congestions and carbon emissions to the environment can be achieved. Although the small EVs are environment-friendly in operation, there are few drawbacks which have always been critical problems. One of the major constraints is the unavailability of charging facilities at regular distances across the country. Promotion of small EVs infrastructure for charging is highly required in that consideration. Most of the times EV charging is taken place at home overnight. But it is not an effective method for the small EVs since they are running in the daytime and as a result, they would be required to charge their batteries frequently during the daytime. Therefore, providing charging infrastructure at parking stations would be an effective way since then small EVs are allowed to recharge their batteries when they are idling at parking stations. Commercially available small EVs such as electrical three-wheelers have battery banks of 24 V/100 Ah, 48 V/100 Ah and 60 V/120 Ah with a range of 80 km – 150 km per charge. Therefore, the nominal energy requirement of a small EV for a 100 km run is around 4.8 kWh. Typically, small EVs are being charged at a power level of around 1 kW. Even though many papers present multi-charging facilities, they are not suitable for small EVs because of higher power ratings which are around 10 kW -50 kW. This paper presents the development of a commercial EV charging station for Small EVs. Multiple small EVs can be charged through this charging station with the capability of fast and normal charging. Also, the proposed charging station can transfer power in both directions thus G2V and V2G can be realized. Design of power electronic converters and controllers are presented with simulation results to validate the operation. Further cost analysis was carried out to investigate the payback time. Low number of vehicles operated in the Czech Republic is also corresponding to the low density of charging stations, most of which are also operated an amateur way. For these purposes seems to be very advantageous to combine the emerging system of infrastructure charging stations with renewable power sources, such as the energy produced by the sun with the possibilities of accumulation and its subsequent delivery to the uniform charging infrastructure.

2.1 Existing System:

The existing system is the manual system. All the records are maintained in the project. Need to be converted into automated system. In People should push the vehicles or get help to reach nearest electric recharge station. In the above method time and manual work is done by owners of the vehicle. For some aged people or medically ill people it will get even hard. To get electric to fill generators people need to go to an electric recharge bunk station.

- Lack of privacy
- Risk in the management of the data.
- Less Security
- Low co-ordination between
- Less User-friendly
- Accuracy not guaranteed
- Not in reach of distant users.

2.2 Proposed System:

The charging station proposed in this paper is targeting small EVs with a battery of 24 V/80 Ah and a range of 50 km which requires an energy capacity of about 2.4 kWh. Therefore, this multi-port charging station is proposed with charging ports of 1 kW for standard or regular charging and 2 kW for fast charging. The proposed charging station is capable of charging five small EVs simultaneously. The charging time of the EVs depends on the use requirement. Out of five charging ports, four ports are provided as normal chargers, which can fully charge the considered small EV within two hours. The remaining charging port is designed as the fast charger which can fully charge a small EV within an hour. In most of the South Asian cities, the average user trip length is below 15 km. Therefore, the fast charger is targeted to charge the EV up to 25% charge within 15 minutes which would allow running around 15 km. A dc off-board charging system is proposed to realize fast charging capabilities. The proposed commercial charging system for small EVs consists of a controlled rectifier with a voltage controller and five full-bridge bidirectional dc-dc converters with current controllers. As shown in Fig 1, the controlled rectifier connects the dc-link with the utility grid and five parallel connected dc-dc converters transfer the power from dc link to the small EVs connected to the system.

The Project has several features and easy to manage.

1. The System is user friendly.

2. Cost effective.

3. Back up support.

4. Secured Data.

CHAPTER 3

SYSTEM STUDY

3.1 Feasibility Study

The feasibility of the project is analysed 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.

Three key considerations involved in the feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

3.1.1 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.

3.1.2 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.

3.1.3 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.

Non-Functional Requirements

Non-functional requirements are the quality requirements that stipulate how well software does what it has to do. These are Quality attributes of any system; these can be seen at the execution of the system and they can also be the part of the system architecture.

Accuracy:

The system will be accurate and reliable based on the design architecture. If there is any problem in the accuracy then the system will provide alternative ways to solve the problem.

Usability:

The proposed system will be simple and easy to use by the users. The users will comfort in order to communicate with the system. The user will be provided with an easy interface of the system.

Accessibility:

The system will be accessible through internet and there should be no any known problem.

Performance:

The system performance will be at its best when performing the functionality of the system.

Reliability:

The proposed system will be reliable in all circumstances and if there is any problem that will be affectively handle in the design.

Security:

The proposed system will be highly secured; every user will be required registration and username/password to use the system. The system will do the proper authorization and authentication of the users based on their types and their requirements. The proposed system will be designed persistently to avoid any misuse of the application.

CHAPTER 4

THE PROPOSED SYSTEM

4.1 Module Description

The system after careful analysis has been identified to be presented with the following modules User, Fuel Station, Admin.

4.2 System Features

In this application an on-demand fuel delivery application for its client in Australia. This application is developed for Android. This on-demand fuel delivery application will serve as a platform for all the users and corporate to get fuel delivered from the truck drivers that will have trucks fully loaded with fuel.

System Modules

User modules:

- Register
- Login
- Search EV Bunk
- View Bunk details
- View slot vacancy
- My booking
- My profile

Admin modules:

- Login
- Approve/Reject recharge slots
- Create EV Bunk
- Manage Bunk details
- Manage recharge details

User

•Register – User has to register their basic details to get access with this application service.

•Login – Once they have registered they need to login to avail the service at the needy time.

•Search Ev Bunk– to search verified near electric recharge bunk station details based on user need.

•View Bunk details–the user search and view all the Bunk details and user can select the nearest bunk.

•View slot vacancy–The user can search nearest bunk and select bunk and check the availability of slots.

•My booking– User will select available slots then user will book a slot for a needed time slots.

•My profile – User once create a their profile like user name, Phone number, address, etc. they are managed by my profile section.

Admin

•Login – Enter username & password into the login page.

•Approve/Reject recharge slots–Admin only accept whether it is verified Ev bunk or not Ev bunk.

•Create EV Bunk– To create the electric recharge bunks enters all the details of a bunk category. All this information will be stored in the database.

•Manage Bunk details – Admin can manage all the bunk details like new bunk details, vacancy slots, etc.

•Manage recharge details – Once can create new bunk details then users recharge booking details also managed by admin

CHAPTER 5
DESIGN

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer's requirements into finished software or a system.

5.1 Data Flow Diagram:

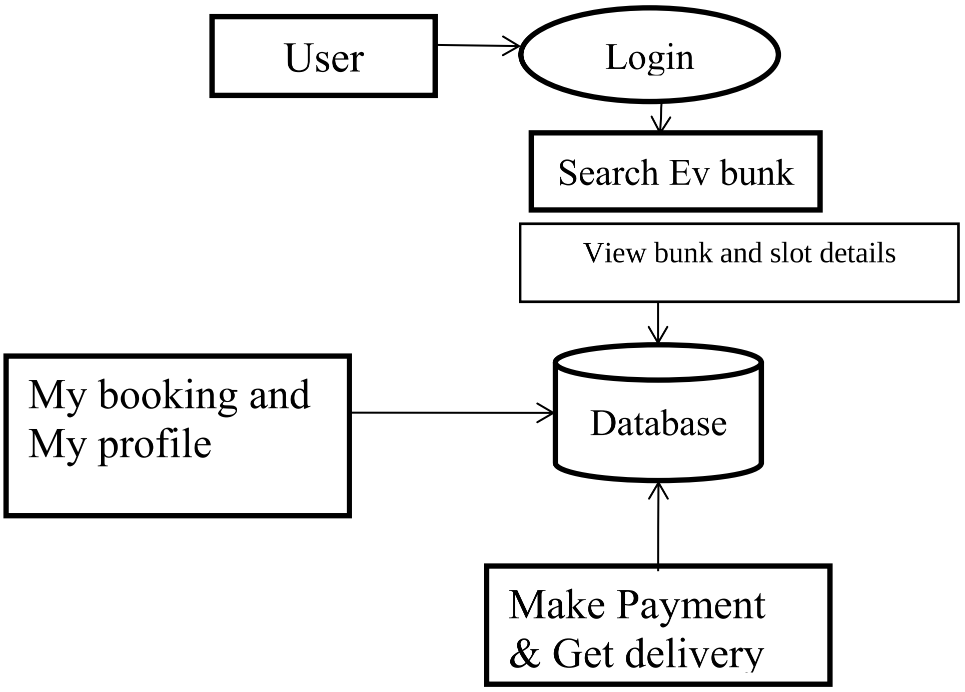

User
Login
Search Ev bunk
View bunk and slot details
My booking and My profile
Database
Make Payment & Get delivery

Fig 4.1 User Data Flow diagram

Admin

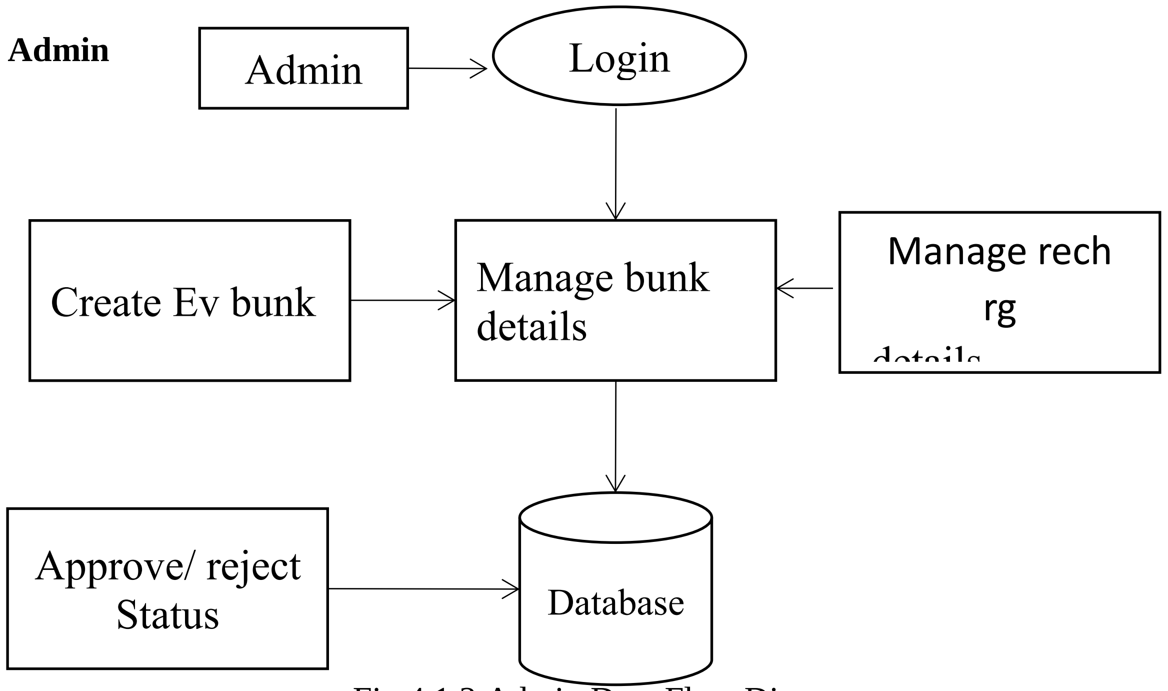


Fig 4.1.2 Admin Data Flow Diagram

5.2 UML Diagrams:

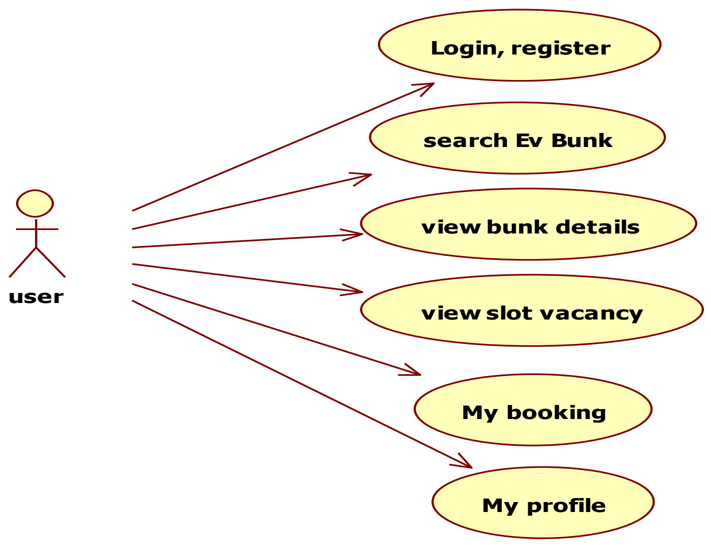
UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

There are various kinds of methods in software design:

- Use case Diagram
- Sequence Diagram
- Collaboration Diagram

5.2.1 Use case Diagrams:

Use case diagrams model behavior within a system and helps the developers understand of what the user requires. The stick man represents what's called an actor. Use case diagram can be useful for getting an overall view of the system and clarifying that can do and more importantly what they can't do. Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.



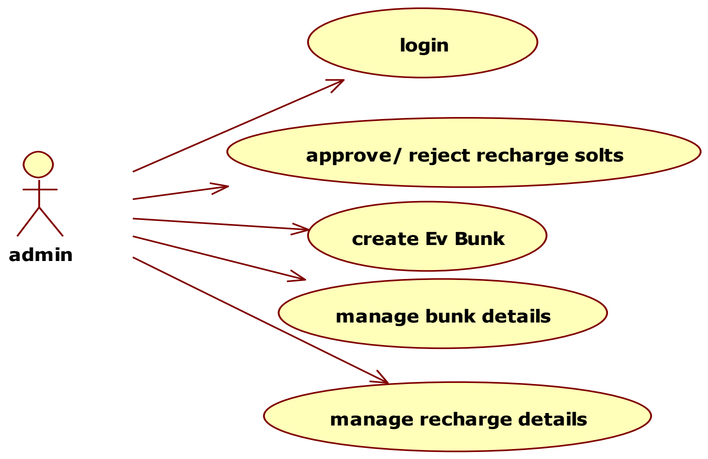


Fig 4.2 Use Case Diagram

5.2.2 Sequence Diagram:

Sequence diagram and collaboration diagram are called INTERACTION DIAGRAMS. An interaction diagram shows an interaction, consisting of set of objects and their relationship including the messages that may be dispatched among them.

A sequence diagram is an introduction that empathizes the time ordering of messages. Graphically a sequence diagram is a table that shows objects arranged along the X-axis and messages ordered in increasing time along the Y-axis.

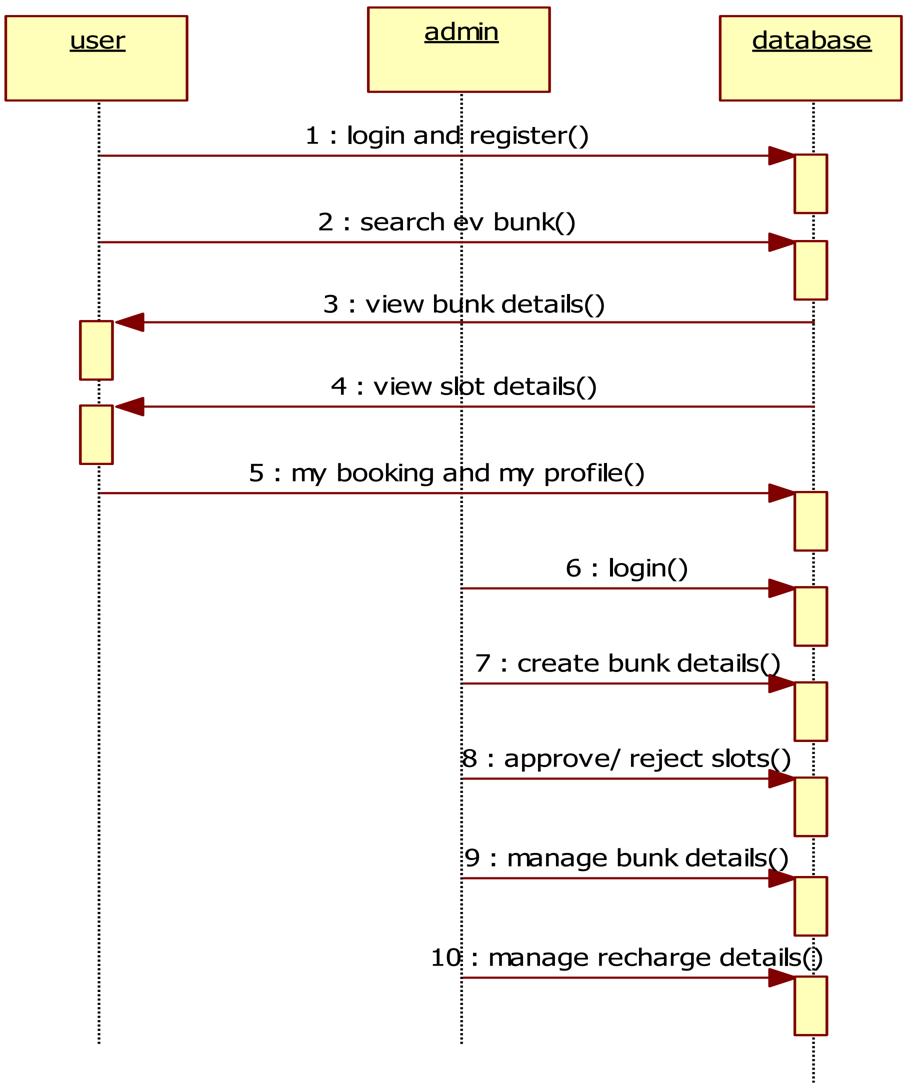

user
admin
database
1 : login and register()
2 : search ev bunk()
3 : view bunk details()
4 : view slot details()
5 : my booking and my profile()
6 : login()
7 : create bunk details()
8 : approve/ reject slots()
9 : manage bunk details()
10 : manage recharge details()

Fig 4.2.1 Sequence Diagram

5.2.3 Collaboration Diagram:

A **collaboration diagram** is a type of visual presentation that shows how various software objects interact with each other within an overall IT architecture and how users can benefit from this **collaboration**. A **collaboration diagram** often comes in the form of a visual chart that resembles a flow chart.

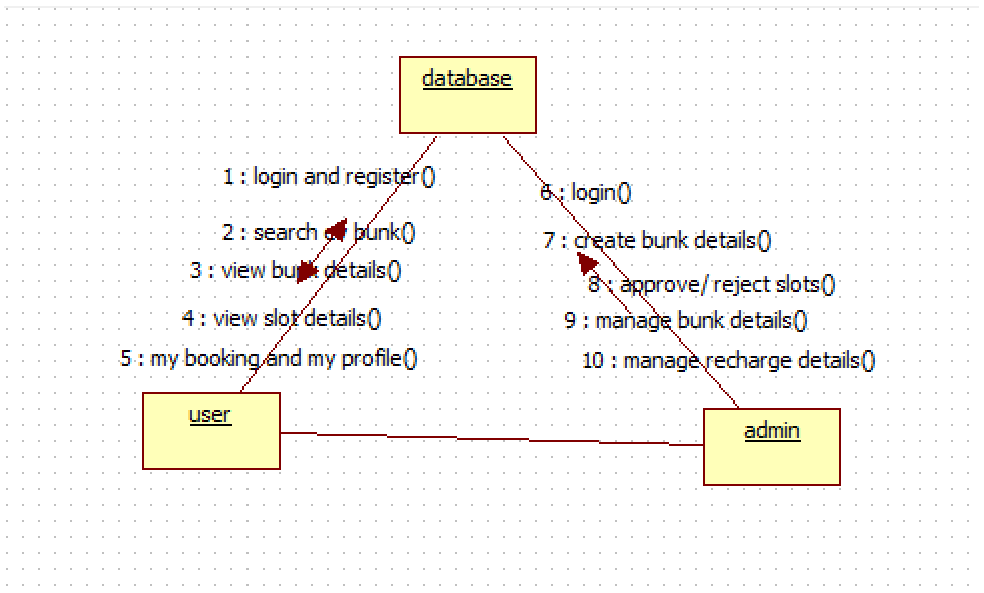

database
1 : login and register()
2 : search ev bunk()
3 : view bunk details()
4 : view slot details()
5 : my booking and my profile()
6 : login()
7 : create bunk details()
8 : approve/ reject slots()
9 : manage bunk details()
10 : manage recharge details()
user
admin

Fig 4.2.2 Collaboration diagram

5.2.4 Architecture Design

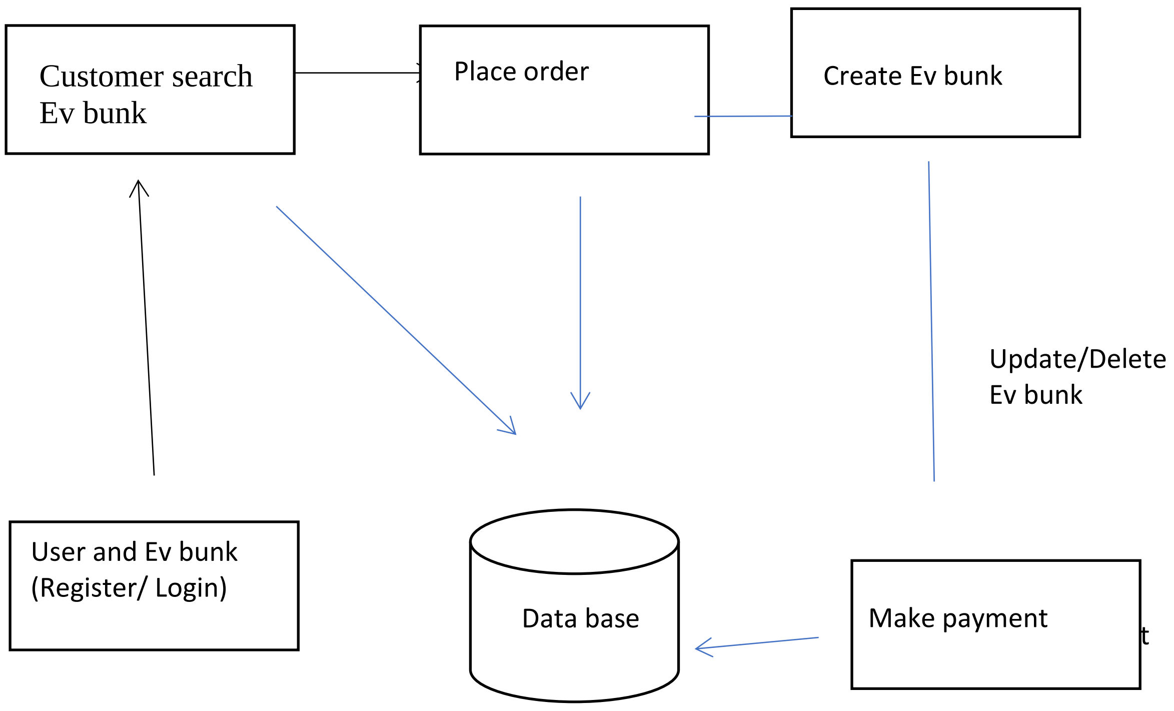
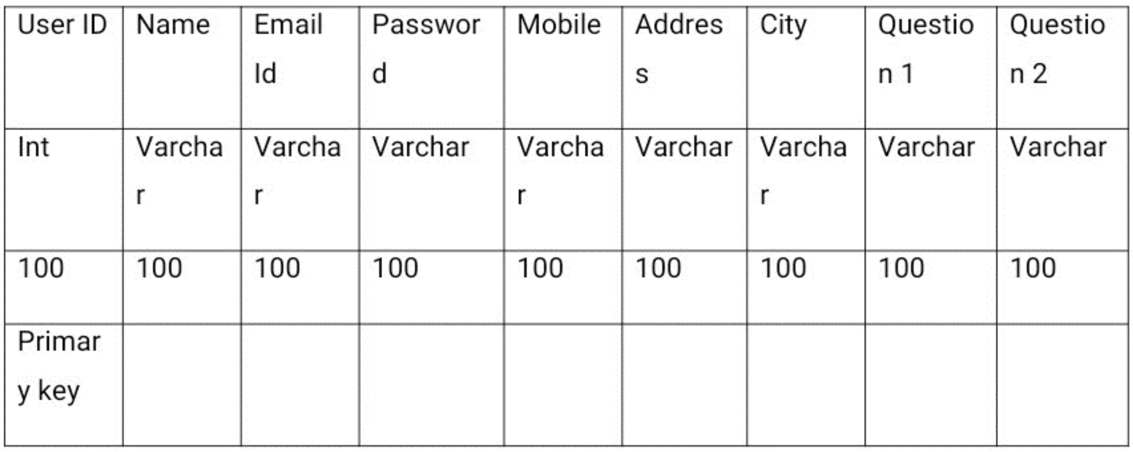


Fig 4.2.3 Architecture Design

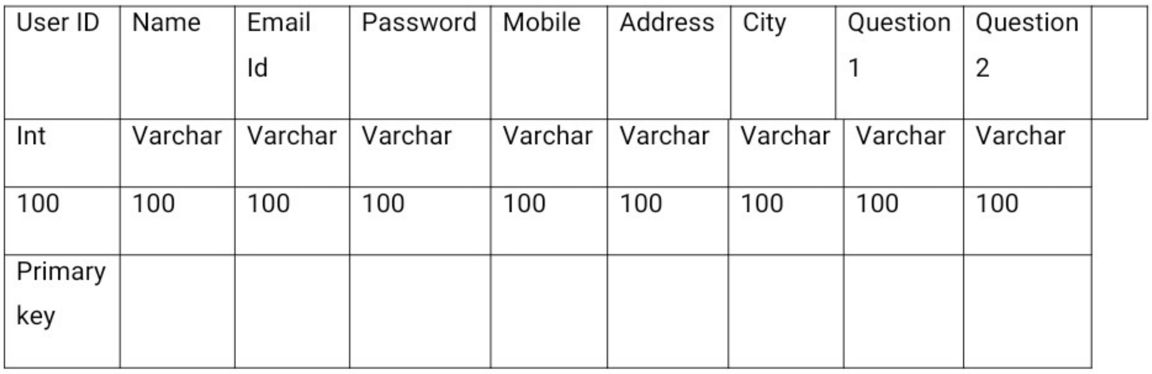
5.3 Table Design:

User Register & Login



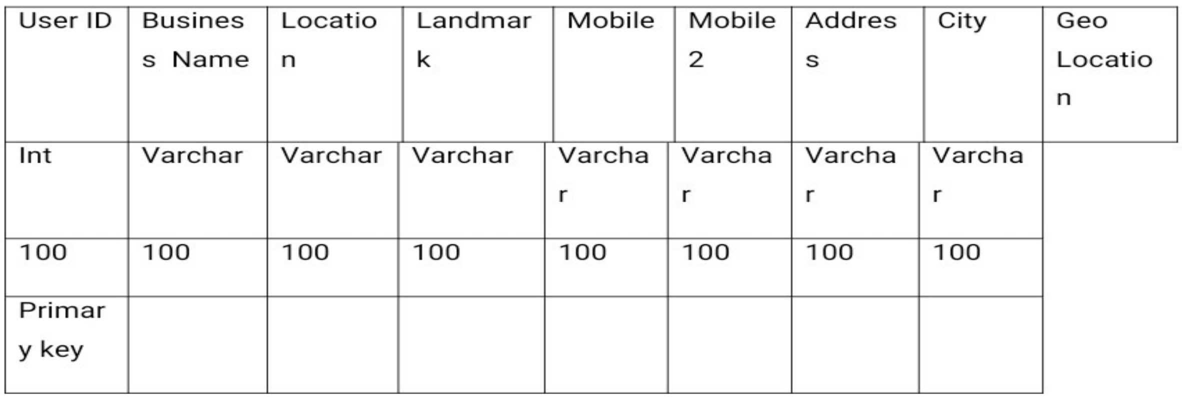
User ID	Name	Email Id	Password	Mobile	Address	City	Question 1	Question 2
Int	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar
100	100	100	100	100	100	100	100	100
Primary key								

Bunk Register & Login



User ID	Name	Email Id	Password	Mobile	Address	City	Question 1	Question 2	
Int	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar	Varchar	
100	100	100	100	100	100	100	100	100	
Primary key									

Post Bunk Details



User ID	Busines s Name	Locatio n	Landmar k	Mobile	Mobile 2	Addres s	City	Geo Locatio n
Int	Varchar	Varchar	Varchar	Varcha r	Varcha r	Varcha r	Varcha r	
100	100	100	100	100	100	100	100	
Primar y key								

Booking Status

User ID	Business Name	Card Details	Exp Date	CVV	Mobile	Qty	Pric e	Ema il
Int	Varchar	Varchar	Varcha r	Varcha r	Varchar	Varch ar	Varc har	
100	100	100	100	100	100	100	100	
Primar y key								

Admin Login

User ID	Email	Password
Int	Varchar	
100	100	
Primary key		

CHAPTER 6
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 cantered 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.

6.1 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

6.2 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.

6.3 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

CHAPTER 7

RESULTS

HOMEPAGE

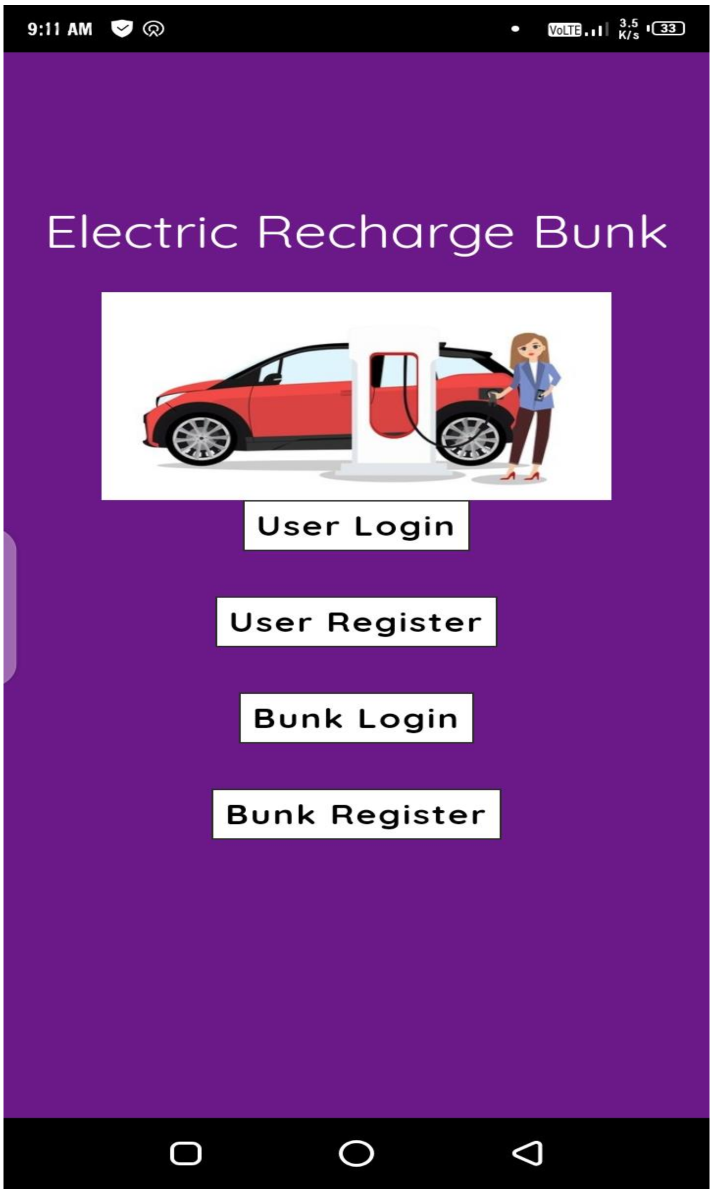


Fig 6.1 Home Page.

By open the Application we see the Homepage. It shows User Login and Register, Bunk login and register.

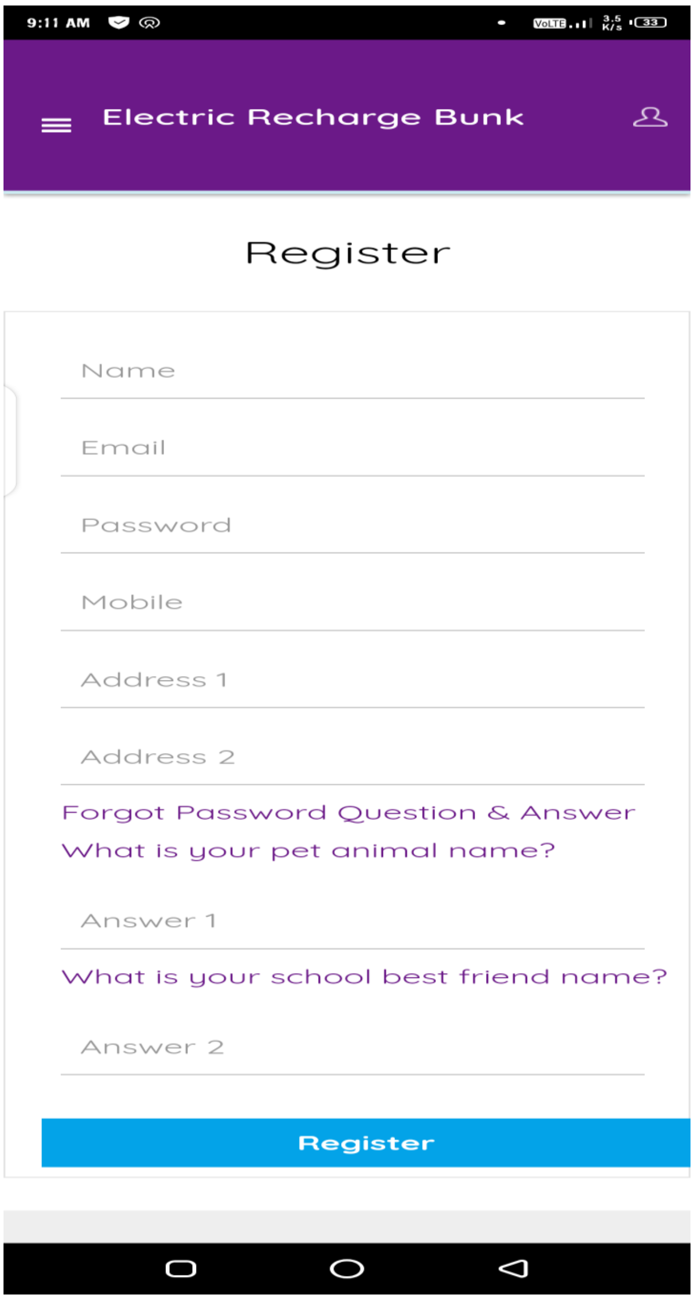


Fig 6.2 Registration Page

REGISTER: by clicking user register it shows the user details.

BUNK REGISTER: after clicking this option we need to enter our personal details to register.

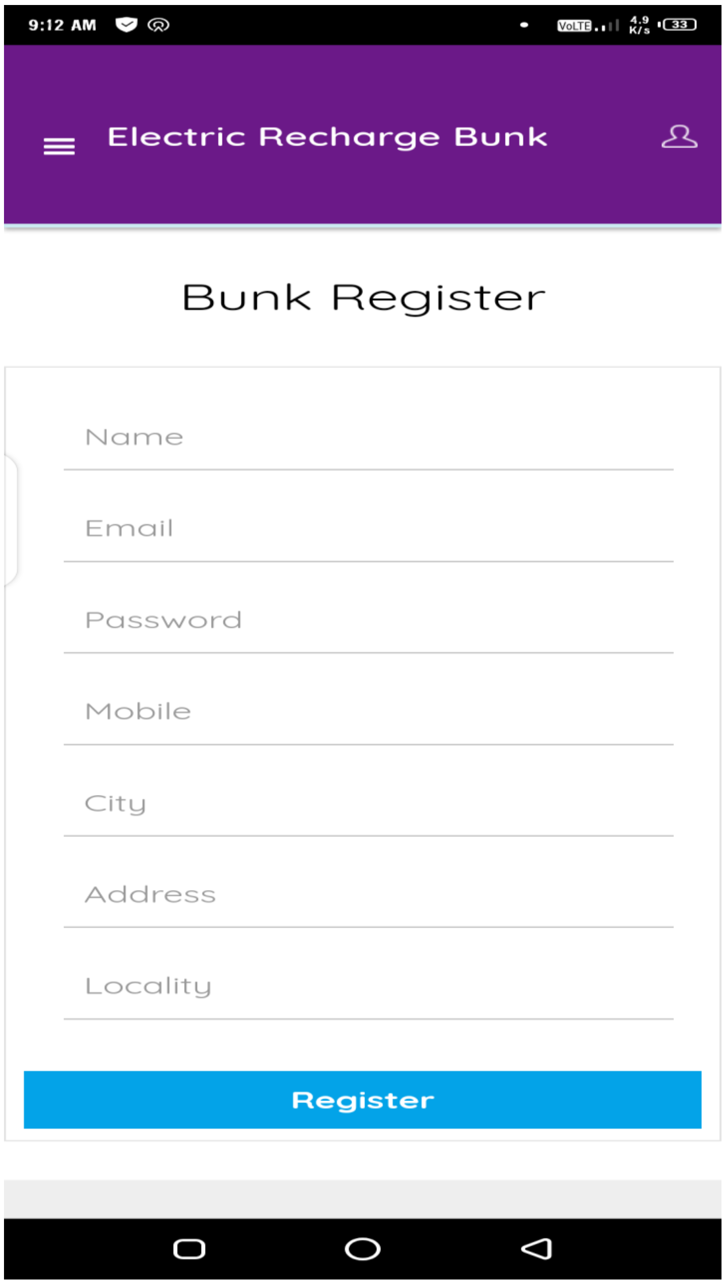
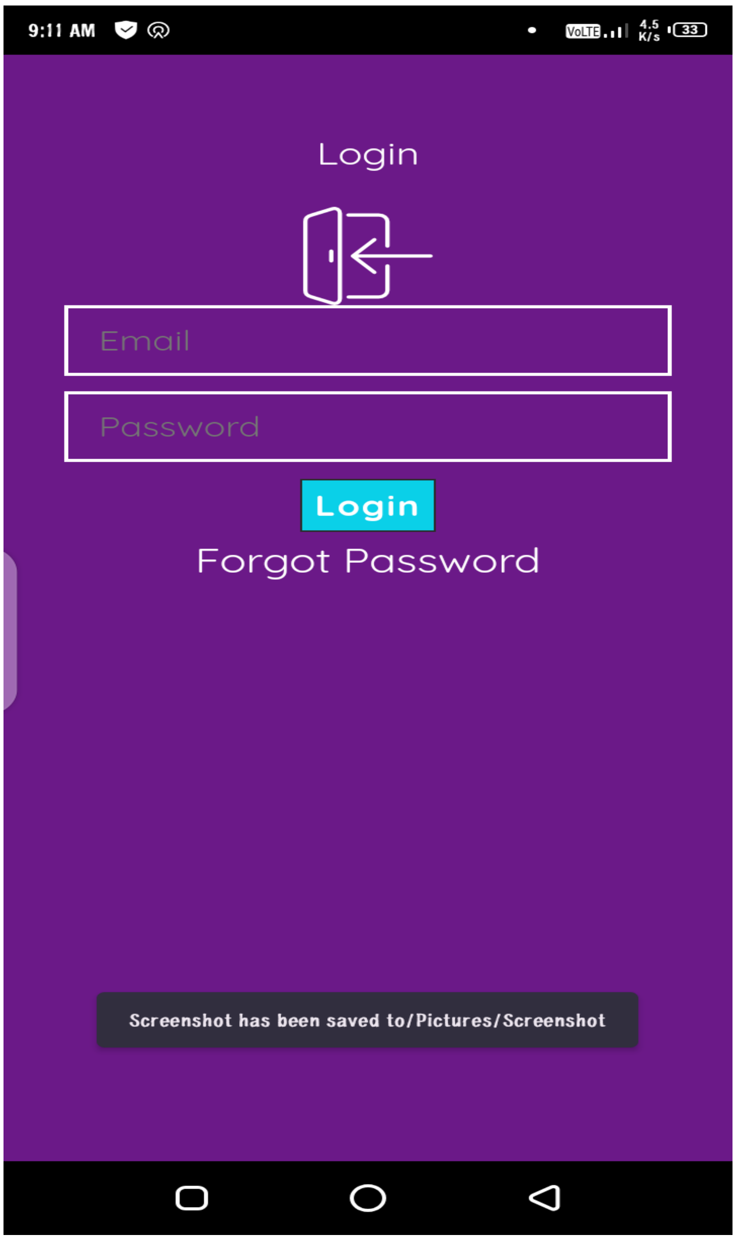


Fig 6.3 Bunk Registration Page

LOGIN: enter Username and Password to Login.



Screenshot has been saved to/Pictures/Screenshot

Fig 6.4 Login Page

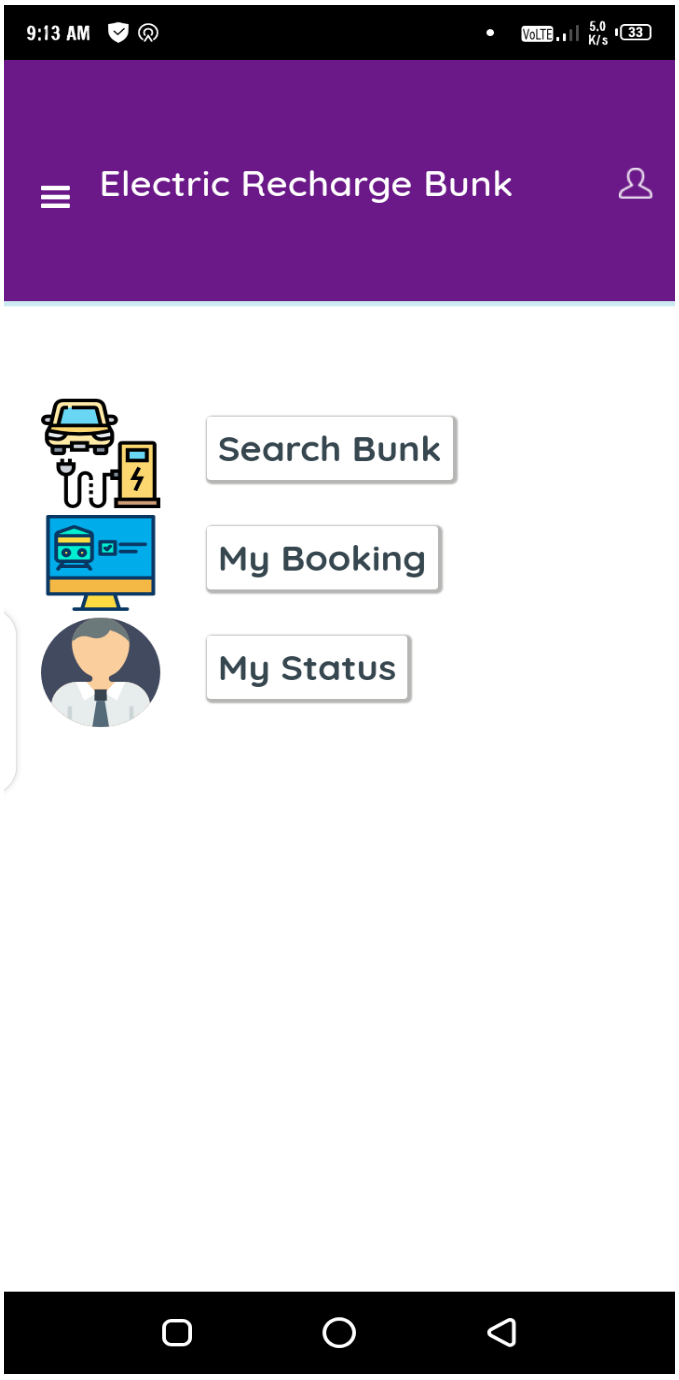
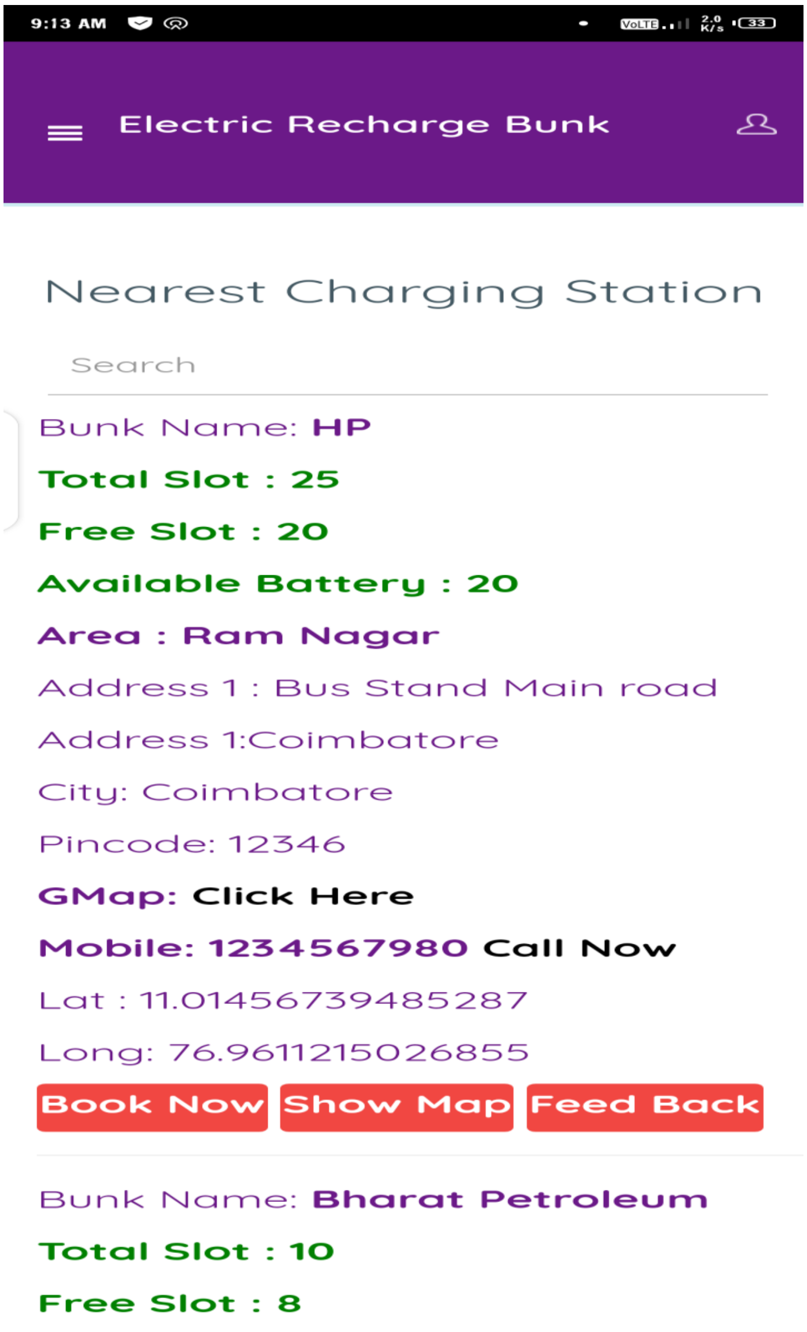


Fig 6.5 Booking Page

Recharge Bunk: by open this option it shows the details of bunks details, bookings and their status.


9:13 AM

Electric Recharge Bunk

Nearest Charging Station

Search

Bunk Name: **HP**

Total Slot : 25

Free Slot : 20

Available Battery : 20

Area : Ram Nagar

Address 1 : Bus Stand Main road

Address 1:Coimbatore

City: Coimbatore

Pincode: 12346

GMap: Click Here

Mobile: 1234567980 Call Now

Lat : 11.01456739485287

Long: 76.9611215026855

Bunk Name: **Bharat Petroleum**

Total Slot : 10

Free Slot : 8

Fig 6.6 Charging Station search Page

Charging Station: by clicking search option, it shows all nearest charging stations.

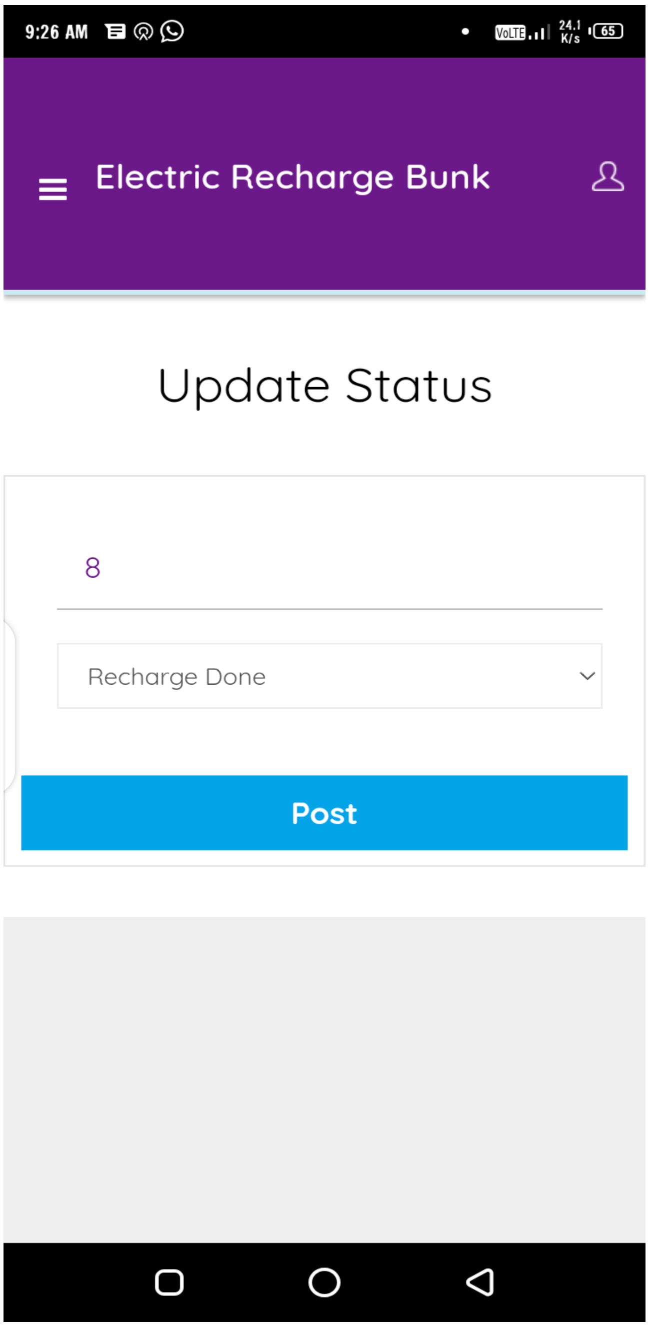
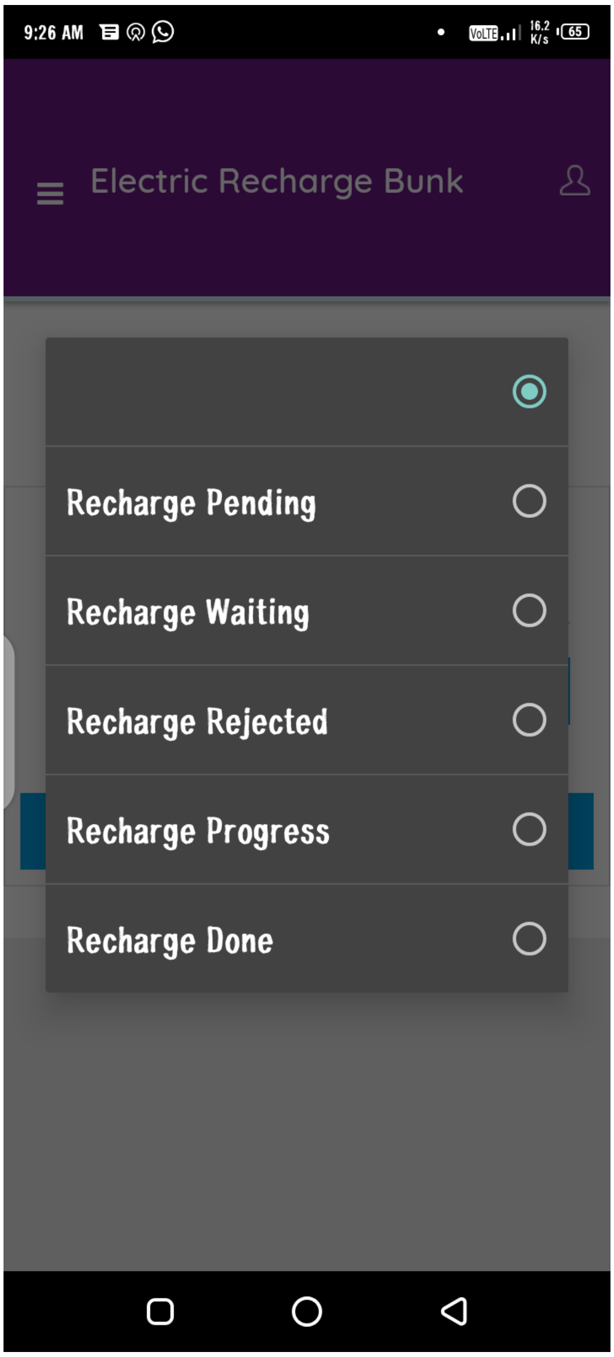


Fig 6.7 Update Status Page

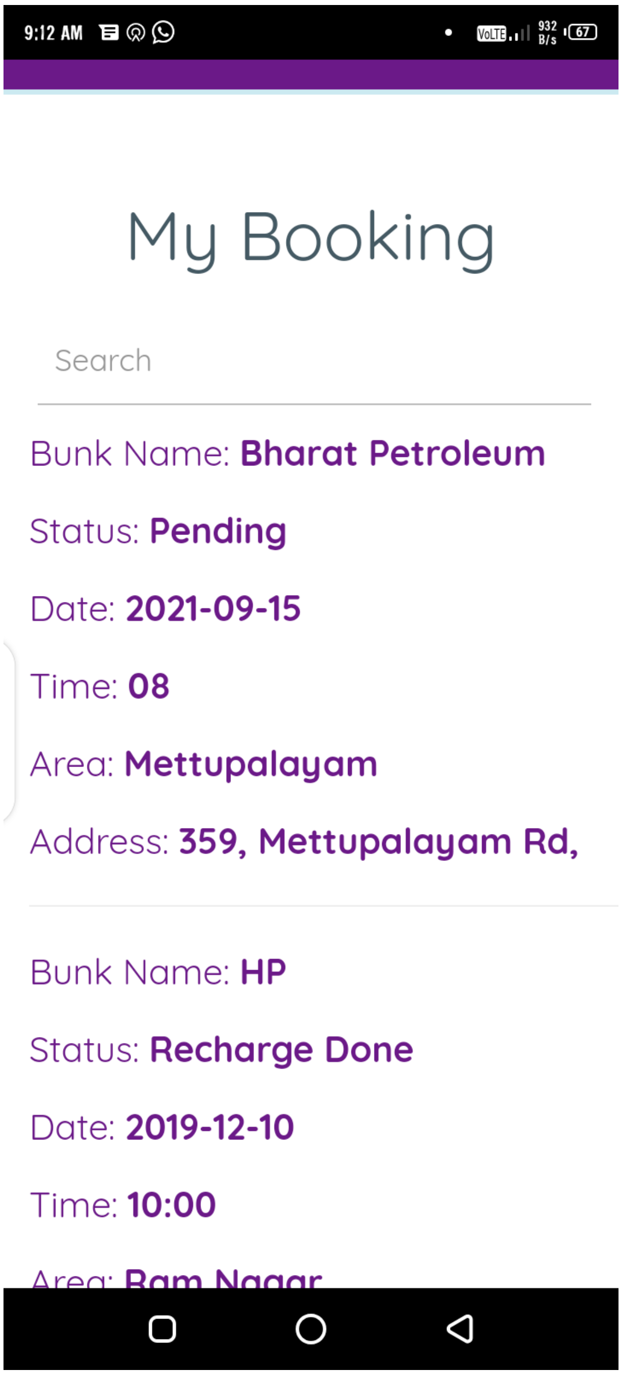
The above fig shows that how much charging done to the vehicle.



- [✓]
- Recharge Pending
- Recharge Waiting
- Recharge Rejected
- Recharge Progress
- Recharge Done

Fig 6.8 Recharge Progress Page.

The above fig shows that recharge status of the vehicle.


9:12 AM

My Booking

Search

Bunk Name: **Bharat Petroleum**

Status: **Pending**

Date: **2021-09-15**

Time: **08**

Area: **Mettupalayam**

Address: **359, Mettupalayam Rd,**

Bunk Name: **HP**

Status: **Recharge Done**

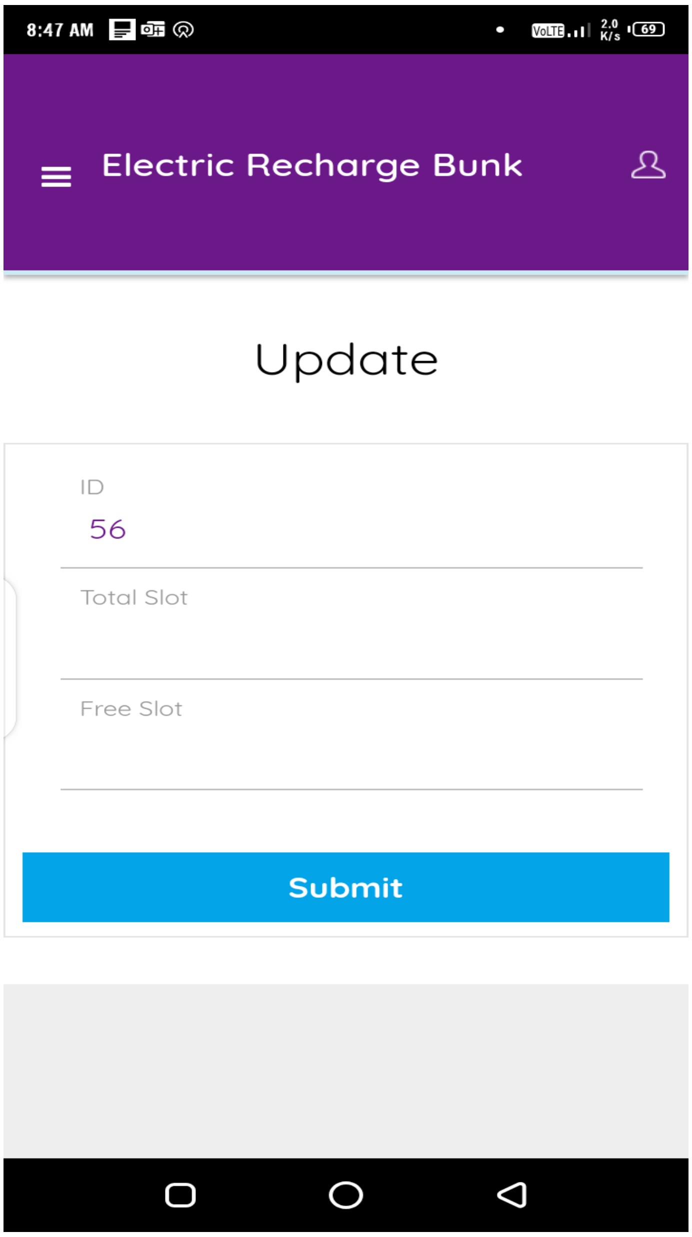
Date: **2019-12-10**

Time: **10:00**

Area: **Ram Nagar**

Fig 6.9 My booking Page

The above fig shows that the electric station booking status.



ID
56
Total Slot
Free Slot

Fig 6.10 Bunk Update Page.

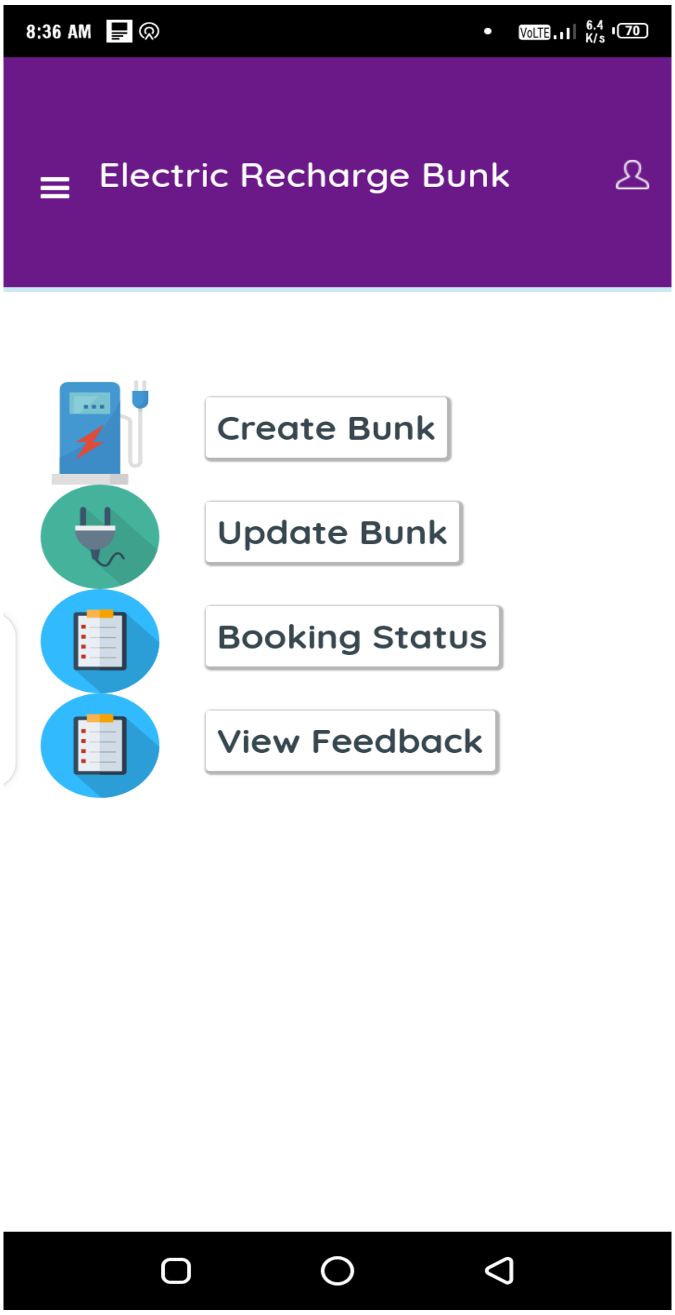


Fig 6.11 Bunk Home Page

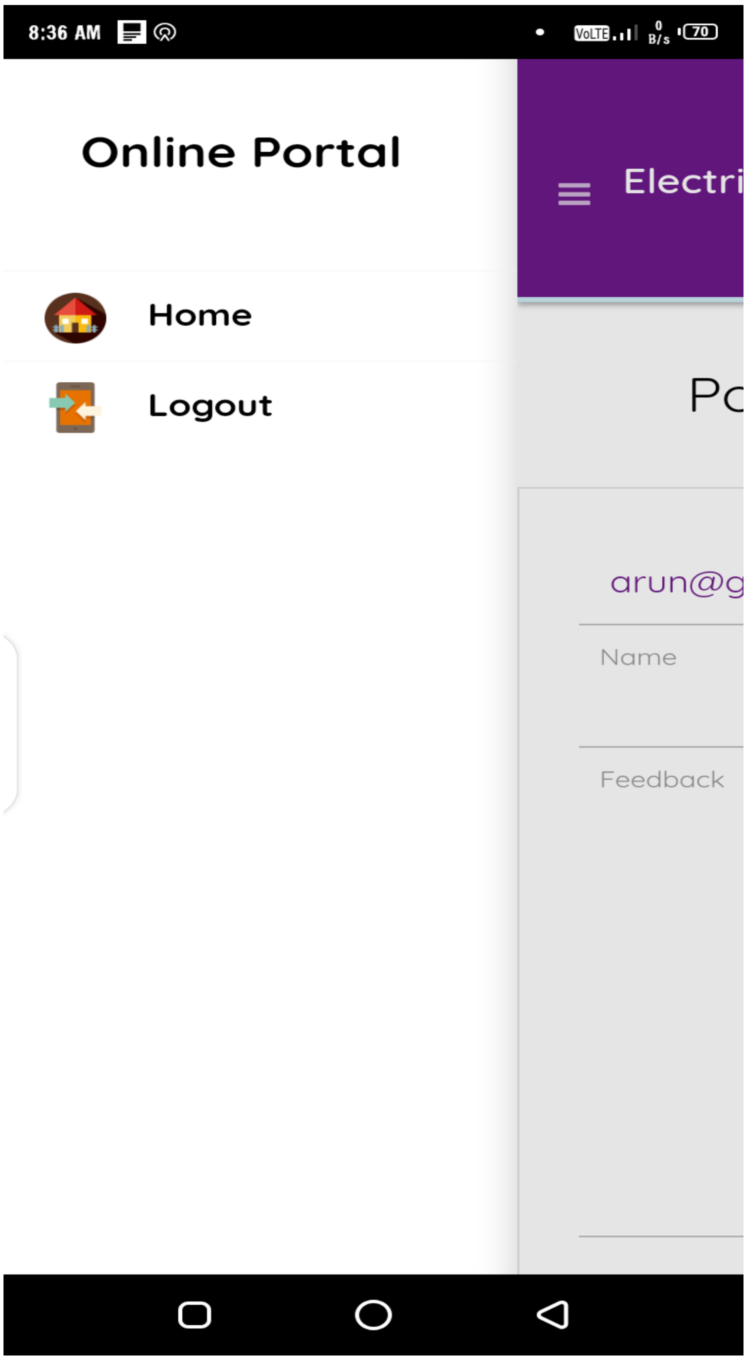
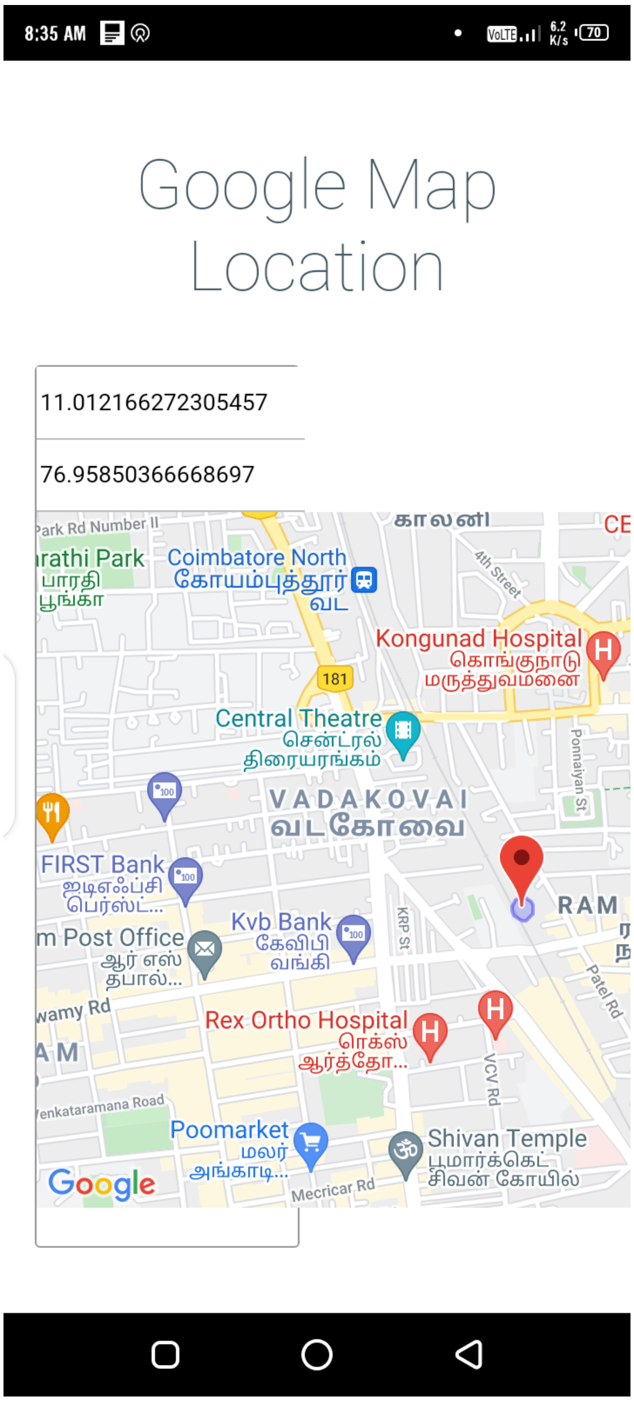


Fig 6.12 Online Portal Page


8:35 AM

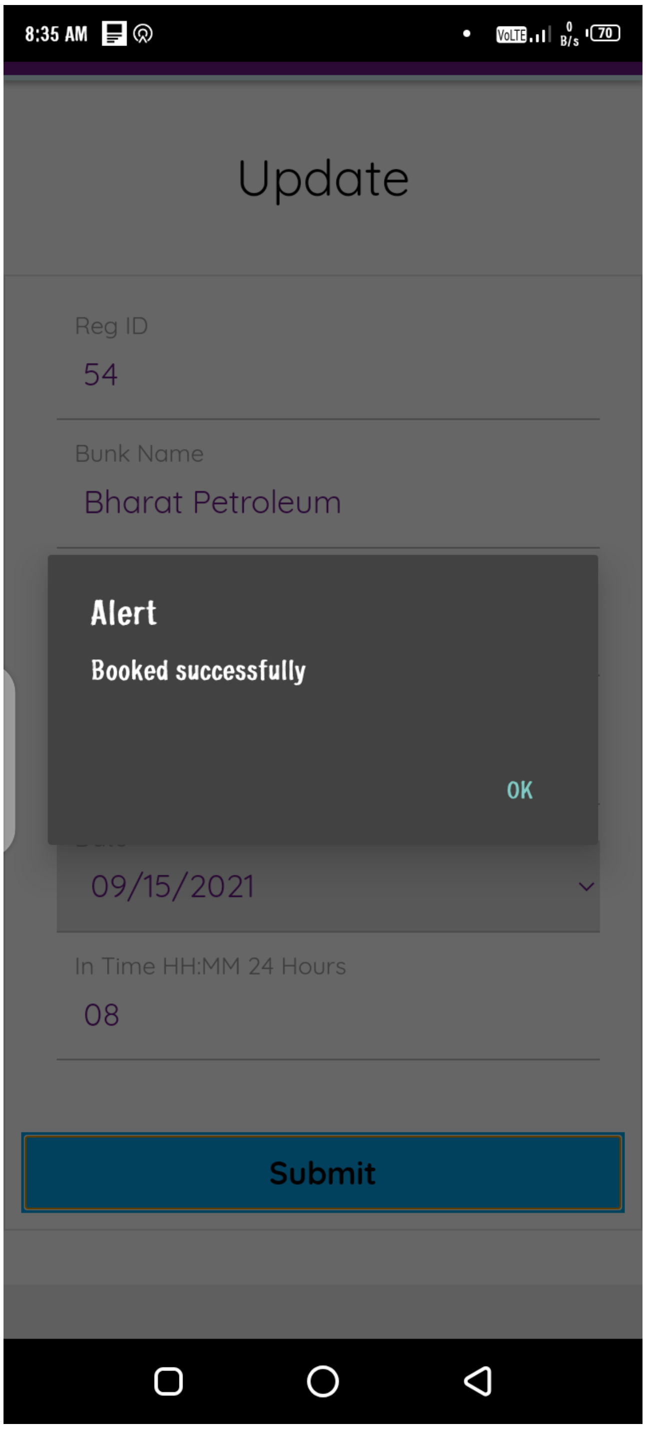
Google Map Location

11.012166272305457

76.95850366668697

Fig 6.13 Location Page

Location: by clicking show map option in nearing bunk station, it shows the Google map for nearest stations.


8:35 AM

Update

Reg ID
54

Bunk Name
Bharat Petroleum

Alert
Booked successfully

09/15/2021

In Time HH:MM 24 Hours
08

Fig 6.14 Booking Update Page

UPDATE: Once booking completed, booking alert will come.

CHAPTER 8

IMPLEMENTATION DETAILS

7.1 Introduction to Html Framework

Hyper Text Markup Language, commonly referred to as HTML, is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology used to create web pages, as well as to create user interfaces for mobile and web applications. Web browsers can read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language, rather than a programming language.

HTML elements form the building blocks of HTML pages. HTML allows images and other objects to be embedded and it can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages. HTML markup can also refer the browser to Cascading Style Sheets (CSS) to define the look and layout of text and other material.

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web.[1]Web browsers receive HTML documents from a webserver or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript which affect the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system. Berners-Lee specified HTML and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes from 1990 he listed"some of the many areas in which hypertext is used" and put an encyclopaedia first.

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Tim Berners-Lee in late 1991.[6][7] It describes 18 elements comprising the initial, relatively simple design of HTML. Except for the hyperlink tag, these were strongly influenced by SGMLguid, an in-house Standard Generalized Markup Language (SGML)-based documentation format at CERN. Eleven of these elements still exist in HTML 4.

HTML is a markup language that web browsers use to interpret and compose text, images, and other material into visual or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS. Many of the text elements are found in the 1988 ISO technical report TR 9537 Techniques for using SGML, which in turn covers the features of early text formatting languages such as that used by the RUNOFF command developed in the early 1960s for the CTSS (Compatible Time-Sharing System) operating system: these formatting commands were derived from the commands used by typesetters to manually format documents. However, the SGML concept of generalized markup is based on elements (nested annotated ranges with attributes) rather than merely print effects, with also the separation of structure and markup; HTML has been progressively moved in this direction with CSS.

Berners-Lee considered HTML to be an application of SGML. It was formally defined as such by the Internet Engineering Task Force (IETF) with the mid-1993 publication of the first proposal for an HTML specification, the "Hypertext Markup Language (HTML)" Internet Draft by Berners-Lee and Dan Connolly, which included an SGML Document Type Definition to define the grammar.[9][10] The draft expired after six months, but was notable for its acknowledgment of the NCSA Mosaic browser's custom tag for embedding in-line images, reflecting the IETF's philosophy of basing standards on successful prototypes. Similarly, Dave Raggett's competing Internet-Draft, "HTML+ (Hypertext Markup Format)", from late 1993, suggested standardizing already-implemented features like tables and fill-out forms.

After the HTML and HTML+ drafts expired in early 1994, the IETF created an HTML Working Group, which in 1995 completed "HTML 2.0", the first HTML specification intended to be treated as a standard against which future implementations should be based.

Further development under the auspices of the IETF was stalled by competing interests. Since 1996, the HTML specifications have been maintained, with input from commercial software vendors, by the World Wide Web Consortium (W3C). However, in 2000, HTML also became an international standard (ISO/IEC 15445:2000). HTML 4.01 was published in late 1999, with further errata published through 2001. In 2004, development began on HTML5 in the Web Hypertext Application Technology Working Group (WHATWG), which became a joint deliverable with the W3C in 2008, and completed and standardized on 28 October 2014.

7.2 Cascading Style Sheets (CSS)

CSS is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions. For example, a CSS rule might specify that "all heading 1 elements should be bold", leaving pure semantic HTML markup that asserts "this text is a level 1 heading" without formatting code such as a<bold> tag indicating how such text should be displayed.

This separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen) and on Braille-based, tactile devices. It can also be used to display the web page differently depending on the screen size or device on which it is being viewed. Although the author of a web page typically links to a CSS file within the markup file, readers can specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author has specified. If the author or the reader did not link the document to a style sheet, the default style of the browser will be applied. Another advantage of CSS is that aesthetic changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in one file, rather than by a laborious (and thus expensive) process of crawling over every document line by line, changing markup.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable.

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts.[3] This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

In CSS, selectors declare which part of the markup a style applies to by matching tags and attributes in the markup itself.

Selectors may apply to:

All elements of a specific type, e.g. the second-level headers h2

elements specified by attribute, in particular id: an identifier unique within the document

class: an identifier that can annotate multiple elements in a document elements depending on how they are placed relative to others in the document tree.

Classes and IDs are case-sensitive, start with letters, and can include alphanumeric characters and underscores. A class may apply to any number of instances of any elements. An ID may only be applied to a single element.

Pseudo-classes are used in CSS selectors to permit formatting based on information that is not contained in the document tree. One example of a widely used pseudo-class is: hover, which identifies content only when the user "points to" the visible element, usually by holding the mouse cursor over it. It is appended to a selector as in a:hover or #elementid:hover. A pseudo-class classifies document elements, such as :link or :visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as ::first-line or ::first-letter.

Selectors may be combined in many ways to achieve great specificity and flexibility.[6] Multiple selectors may be joined in a spaced list to specify elements by location, element type, id, class, or any combination thereof. The order of the selectors is important. For example, div .myClass {color: red;} applies to all elements of class myClass that are inside div elements, whereas .myClass div {color: red;} applies to all div elements that are in elements of class myClass.

CSS information can be provided from various sources. These sources can be the web browser, the user and the author. The information from the author can be further classified into inline, media type, importance, selector specificity, rule order, inheritance and property definition. CSS style information can be in a separate document or it can be embedded into an HTML document. Multiple style sheets can be imported. Different styles can be applied depending on the output device being used; for example, the screen version can be quite different from the printed version, so that authors can tailor the presentation appropriately for each medium.

The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. This process is called cascading.

One of the goals of CSS is to allow users greater control over presentation. Someone who finds red italic headings difficult to read may apply a different style sheet. Depending on the browser and the web site, a user may choose from various style sheets provided by the designers, or may remove all added styles and view the site using the browser's default styling, or may override just the red italic heading style without altering other attributes.

CSS was first proposed by HåkonWium Lie on October 10, 1994. At the time, Lie was working with Tim Berners-Lee at CERN. Several other style sheet languages for the web were proposed around the same time, and discussions on public mailing lists and inside World Wide Web Consortium resulted in the first W3C CSS Recommendation (CSS1) being released in 1996. In particular, Bert Bos' proposal was influential; he became co-author of CSS1 and is regarded as co-creator of CSS.

Style sheets have existed in one form or another since the beginnings of Standard Generalized Markup Language (SGML) in the 1980s, and CSS was developed to provide style sheets for the web. One requirement for a web style sheet language was for style sheets to come from different sources on the web. Therefore, existing style sheet languages like DSSSL and FOSI were not suitable. CSS, on the other hand, let a document's style be influenced by multiple style sheets by way of "cascading" styles.

As HTML grew, it came to encompass a wider variety of stylistic capabilities to meet the demands of web developers. This evolution gave the designer more control over site appearance, at the cost of more complex HTML. Variations in web browser implementations, such as ViolaWWW and WorldWideWeb, made consistent site appearance difficult, and users had less control over how web content was displayed. The browser/editor developed by Tim Berners-Lee had style sheets that were hard-coded into the program. The style sheets could therefore not be linked to documents on the web. Robert Cailliau, also of CERN, wanted to separate the structure from the presentation so that different style sheets could describe different presentation for printing, screen-based presentations, and editors.

Improving web presentation capabilities was a topic of interest to many in the web community and nine different style sheet languages were proposed on the www-style mailing list. Of these nine proposals, two were especially influential on what became CSS: Cascading HTML Style Sheets and Stream-based Style Sheet Proposal (SSP). Two browsers served as testbeds for the initial proposals; Lie worked with Yves Lafon to implement CSS in Dave Raggett'sArena browser. Bert Bos implemented his own SSP proposal in the Argo browser. Thereafter, Lie and Bos worked together to develop the CSS standard (the 'H' was removed from the name because these style sheets could also be applied to other markup languages besides HTML).

Lie's proposal was presented at the "Mosaic and the Web" conference (later called WWW2) in Chicago, Illinois in 1994, and again with Bert Bos in 1995. Around this time the W3C was already being established, and took an interest in the development of CSS. It organized a workshop toward that end chaired by Steven Pemberton. This resulted in W3C adding work on CSS to the deliverables of the HTML editorial review board (ERB). Lie and Bos were the primary technical staff on this aspect of the project, with additional members, including Thomas Reardon of Microsoft, participating as well. In August 1996 Netscape Communication Corporation presented an alternative style sheet language called JavaScript Style Sheets (JSSS). The spec was never finished and is deprecated. By the end of 1996, CSS was ready to become official, and the CSS level 1 Recommendation was published in December.

Development of HTML, CSS, and the DOM had all been taking place in one group, the HTML Editorial Review Board (ERB). Early in 1997, the ERB was split into three working groups: HTML Working group, chaired by Dan Connolly of W3C; DOM Working group, chaired by Lauren Wood of SoftQuad; and CSS Working group, chaired by Chris Lilley of W3C.

The CSS Working Group began tackling issues that had not been addressed with CSS level 1, resulting in the creation of CSS level 2 on November 4, 1997. It was published as a W3C Recommendation on May 12, 1998. CSS level 3, which was started in 1998, is still under development as of 2014.

In 2005 the CSS Working Groups decided to enforce the requirements for standards more strictly. This meant that already published standards like CSS 2.1, CSS 3 Selectors and CSS 3 Text were pulled back from Candidate Recommendation to Working Draft level.

7.3 MYSQL Server

MySQL is an open-source relational database management system (RDBMS); in July 2013, it was the world's second most widely used RDBMS, and the most widely used open-source client–server model RDBMS. It is named after co-founder Michael Widenius's daughter, My. The SQL acronym stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedishcompany MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality.

SQL Server Management Studio (SSMS) is a software application first launched with MicrosoftSQL Server 2005 that is used for configuring, managing, and administering all components within Microsoft SQL Server. The tool includes both script editors and graphical tools which work with objects and features of the server.

A central feature of SSMS is the Object Explorer, which allows the user to browse, select, and act upon any of the objects within the server. It also shipped a separate Express edition that could be freely downloaded, however recent versions of SSMS are fully capable of connecting to and manage any SQL Server Express instance. Microsoft also incorporated backwards compatibility for older versions of SQL Server thus allowing a newer version of SSMS to connect to older versions of SQL Server instances.

Starting from version 11, the application was based on the Visual Studio 2010 shell, using WPF for the user interface.

In June 2015, Microsoft announced their intention to release future versions of SSMS independently of SQL Server database engine releases.

7.4 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. Originally created by RasmusLerdorf in 1994, the PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for therecursive backronym PHP: Hypertext Pre-processor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management system and web frameworks. PHP code is usually processed by a PHPinterpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface(CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification.

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by RasmusLerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML or HTML5 code, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification.

PHP development began in 1995 when RasmusLerdorf wrote several Common Gateway Interface (CGI) programs in C, which he used to maintain his personal homepage. He extended them to work with web forms and to communicate with databases, and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI.

PHP/FI could help to build simple, dynamic web applications. To accelerate bug reporting and to improve the code, Lerdorf initially announced the release of PHP/FI as "Personal Home Page Tools (PHP Tools) version 1.0" on the Usenet discussion group comp.infosystems.www.authoring.cgi on June 8, 1995. This release already had the basic functionality that PHP has as of 2013. This included Perl-like variables, form handling, and the ability to embed HTML. The syntax resembled that of Perl but was simpler, more limited and less consistent.

Lerdorf did not intend the early PHP to become a new programming language, but it grew organically, with Lerdorf noting in retrospect: "I don't know how to stop it, there was never any intent to write a programming language […] I have absolutely no idea how to write a programming language, I just kept adding the next logical step on the way." A development team began to form and, after months of work and beta testing, officially released PHP/FI 2 in November 1997.

The fact that PHP lacked an original overall design but instead developed organically has led to inconsistent naming of functions and inconsistent ordering of their parameters. In some cases, the function names were chosen to match the lower-level libraries which PHP was "wrapping", while in some very early versions of PHP the length of the function names was used internally as a hash function, so names were chosen to improve the distribution of hash values.

7.5 ANGULAR JAVA SCRIPT

AngularJS (commonly referred to as "Angular" or "Angular.js") is an open-source web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel(MVVM) architectures, along with components commonly used in rich Internet applications.

The AngularJS framework works by first reading the HTML page, which has embedded into it additional custom tag attributes. Angular interprets those attributes as directives to bind input or output parts of the page to a model that is represented by standard JavaScript variables. The values of those JavaScript variables can be manually set within the code, or retrieved from static or dynamic JSON resources.

According to JavaScript analytics service Libscore, AngularJS is used on the websites of Wolfram Alpha, NBC,Walgreens, Intel, Sprint, ABC News, and approximately 8,400 other sites out of 1 million tested in July 2015.

AngularJS is the frontend part of the MEAN stack, consisting of MongoDB database, Express.js web application server framework, Angular.js itself, and Node.js runtime environment.

AngularJS is an open source web application framework. It was originally developed in 2009 by MiskoHevery and Adam Abrons. It is now maintained by Google. Its latest version is 1.4.3.

Definition of AngularJS as put by its official documentation is as follows −

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate much of the code you currently have to write. And it all happens within the browser, making it an ideal partner with any server technology.

Features

- AngularJS is a powerful JavaScript based development framework to create RICH Internet Application(RIA).
- AngularJS provides developers options to write client side application (using JavaScript) in a clean MVC(Model View Controller) way.

- Application written in AngularJS is cross-browser compliant. AngularJS automatically handles JavaScript code suitable for each browser.
- AngularJS is open source, completely free, and used by thousands of developers around the world. It is licensed under the Apache License version 2.0.
- Overall, AngularJS is a framework to build large scale and high-performance web application while keeping them as easy-to-maintain.

Core Features

Following are most important core features of AngularJS −

- **Data-binding** − It is the automatic synchronization of data between model and view components.
- **Scope −**These are objects that refer to the model. They act as a glue between controller and view.
- **Controller −**These are JavaScript functions that are bound to a particular scope.
- **Services −** AngularJS come with several built-in services for example $https: to make aXMLHttpRequests. These are singleton objects which are instantiated only once in app.
- **Filters** − These select a subset of items from an array and returns a new array.
- **Directives −** Directives are markers on DOM elements (such as elements, attributes, css, and more). These can be used to create custom HTML tags that serve as new, custom widgets. AngularJS has built-in directives (ngBind, ngModel...)
- **Templates** − These are the rendered view with information from the controller and model. These can be a single file (like index.html) or multiple views in one page using "partials".
- **Routing** − It is concept of switching views.
- **Model View Whatever** − MVC is a design pattern for dividing an application into different parts (called Model, View and Controller), each with distinct responsibilities. AngularJS does not implement MVC in the traditional sense, but rather something closer to MVVM (Model-View-ViewModel). The Angular JS team refers it humorously as Model View Whatever.

- **Deep Linking** − Deep linking allows you to encode the state of application in the URL so that it can be bookmarked. The application can then be restored from the URL to the same state.
- **Dependency Injection** − AngularJS has a built-in dependency injection subsystem that helps the developer by making the application easier to develop, understand, and test.

CHAPTER 9
CONCLUSIONS& FUTURE WORK

Gasoline is a heavily demanded natural resource and most is consumed on transportation. Its electrification can relieve our dependence on gasoline and tremendously reduce the amount of harmful gases released, which partially constitute global warming and worsen our health. In the 21st century, advancing EV technologies has become one of the keys to boost a nation’s economy and maintain (and improve) people’s quality of living. For long-term planning, modernizing our cities with EVs is of utmost importance. EVs will be integrated into the transportation system seamlessly and this will help make our cities “smart”. To do this, we need to determine the best locations to construct charging stations in the city. In this paper, we focus on human factors rather than technological ones. An EV should always be able to access a charging station within its driving capacity anywhere in the city. Our contributions in this paper include: 1) formulating the problem, 2) identifying the properties of the problem, and 3) developing an efficient greedy algorithm. We formulate the problem as an optimization model, based on the charging station coverage and the convenience of drivers. We study the NP-hardness of the problem and propose a greedy algorithm based on the network properties of the problem. Simulation results reveal that the greedy algorithm can result in solutions comparable to those obtained from MIP but require much less computation time.

REFERENCES

[1] S. Chen and L. Tong, “iEMS for large scale charging of electric vehicles: Architecture and optimal online scheduling,” in Proceedings of IEEE International Conference on Smart Grid Communications, Tainan City, Taiwan, Nov. 2012.

[2] N. Chen, T. Q. S. Quek, and C. W. Tan, “Optimal charging of electric vehicles in smart grid: Characterization and valley-filling algorithms,” in Proceedings of IEEE International Conference on Smart Grid Communications, Tainan City, Taiwan, Nov. 2012.

[3] A. Y. S. Lam, L. Huang, A. Silva, and W. Saad, “A multi-layer market for vechicle-to-grid energy trading in the smart grid,” in Proceedings of 1st IEEE INFOCOM Workshop on Green Networking and Smart Grids, Orlando, FL, Mar. 2012.

[4] A. Y. S. Lam, K.-C. Leung, and V. O. K. Li, “Capacity management of vehicle-to-grid system for power regulation services,” in Proceedings of IEEE International Conference on Smart Grid Communications, Tainan City, Taiwan, Nov. 2012.

[5] J. J. Q. Yu, V. O. K. Li, and A. Y. S. Lam, “Optimal V2G scheduling of electric vehicles and unit commitment using chemical reaction optimization,” in Proceedings of IEEE Congress on Evolutionary Computation, Cancun, Mexico, Jun. 2013.

[6] F. Guo, E. Inoa, W. Choi, and J. Wang, “Study on global optimization and control strategy development for a PHEV charging facility,” IEEE Trans. Veh. Technol., vol. 61, pp. 2431–2441, Jul. 2012.

[7] M. Etezadi-Amoli, K. Choma, and J. Stefani, “Rapid-charge electricvehicle stations,” IEEE Trans. Power Del., vol. 25, pp. 1883–1887, Jul. 2010.

[8] A. Masoum, S. Deilami, P. Moses, M. Masoum, and A. Abu-Siada, “Smart load management of plug-in electric vehicles in distribution and residential networks with charging stations for peak shaving and loss minimisation considering voltage regulation,” IET Generation, Transmission & Distribution, vol. 5, pp. 877–888, Aug. 2011.

[9] Z. Liu, F. Wen, and G. Ledwich, “Optimal planning of electric-vehicle charging stations in distribution systems,” IEEE Trans. Power Del., vol. 28, pp. 102–110, Jan. 2013.