IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125240

SMART MANAGEMENT OF EV CHARGING STATIONS USING AI CHATBOT AND GMAPS API

Prof.Gajanan Kumbhar, Aryan Borkar, Sunil Kamle, Zishan Inamdar, Sayyed Razzak

D. Y. Patil College of Engineering, Pune

Abstract—Car companies like TATA and TESLA have introduced new electric vehicles in the market, accompanied by the establishment of charging stations. However, the current charging time ranges from 15 minutes to half an hour, leading to potential delays when stations are fully occupied. To address these issues, our solution involves connecting all electric car charging stations into a unified system. Users can conveniently locate and select their preferred station, particularly beneficial for long-distance travel with electric vehicles, ultimately saving time. The system is user-friendly, allowing for slot reservation if available, and prompting users to input a new time if unavailable. A percentage of the payment is required online to confirm the booking. Additionally, our system provides the shortest route to the selected station and offers charging stations an interface to manage available and booked slots. Our Android-based system utilizes time-slot allocation techniques and leverages Google Maps API for direction sensing. Voice commands can be utilized to control the software through our chatbot system, while an online payment gateway facilitates swift transactions. By adopting our system, users can significantly save time, easily locate and book suitable charging stations.

Keywords: Management System, charging slot, EV Cars, Maps.

INTRODUCTION

Global warming and the depletion of fossil fuels due to excessive energy consumption have become urgent global concerns. To combat these issues, the installation of renewable energy systems, independent of fossil fuels, is crucial. In Japan, the government's implementation of Feed-in Tariffs (Fit) has led to rapid adoption of photovoltaic systems. However, the increased output from these systems has negatively impacted system frequency and voltage distribution. Consequently, the Japanese government is reevaluating the Fit system. Additionally, the cost of photovoltaic installation is decreasing annually, indicating a significant drop in future PV power prices. This study proposes the use of EV charging stations as aggregators, primarily purchasing power from PV systems in smart houses and supplying power to electric vehicles (EVs) and smart houses. These charging stations require fixed batteries for electricity trading.

In this project, we aim to provide a platform for customers to book charging slots at available charging stations according to their needs. The system offers various features, including an AI

chatbot for vocal command-based station bookings, mapping capabilities for direction sensing, digital payment options, as well as notifications, emails, and SMS alerts for each activity.
Electric vehicles can be charged using different types of charging infrastructure tailored to specific locations and requirements. This chapter emphasizes the importance of considering local planning and implementation for EV charging networks, highlighting the technical aspects and standards of EV chargers.

MOTIVATION

Our project aims to create a hybrid web-based Android application that facilitates EV car owners in booking charging station slots in advance. The primary objective is to provide a convenient system for users to reserve a slot at a charging station before arriving to charge their vehicles. To enhance user experience, we propose the integration of an AI voice chatbot that enables users to interact with the system using vocal commands. Furthermore, the system will utilize the GMAPS API to offer users the shortest route to their desired charging station.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125240

PROBLEM DEFINITION

EV charging is versatile, with various methods based on location and need. Charging infrastructure is diverse, catering to different EV types and purposes. This chapter outlines technical aspects and standards for EV chargers, emphasizing the importance of context in planning and implementing charging networks. Local considerations enable efficient and effective deployment, fostering EV adoption.

SOFTWARE REQUIREMENT

Purpose and Scope of Document

The purpose of this project is to create a web-based hybrid Android application that facilitates the smart management of EV charging stations. The main objective is to enable customers to conveniently book charging slots for their electric vehicles (EVs).

The scope of the project is to provide an efficient and user-friendly platform for customers to book charging slots at EV charging stations. The application aims to offer an easy and fast method for users to reserve their desired charging time slots, ultimately saving their time and ensuring a seamless charging experience. The focus is on enhancing customer convenience and streamlining the process of accessing and utilizing EV charging stations.

Overview of responsibilities of Developer

1.To have understanding of the problem statement.
2.To know what are the hardware and software
requirements of proposed system.
3.To have understanding of proposed system.
4.To do planning various activities with the help of planner.
5.Designing, programming, testing etc.
RAM : 2 GB (min)
Hard Disk : 1 GB
Hard Disk memory is required.
Processor : Intel i3/i5/i7
Eclipse, Android Studio IDE that Integrated Development
Environment is to be used and data loading should be fast hence
Fast Processor is required

IDE : Eclipse, Android Studio
Best Integrated Development Environment as it gives
possible suggestions at the time of typing code
snippets that makes typing feasible and fast
Front End- HTML, JDK 1.8, JSP

Application Server- Apache Tomcat 7/8/9

Scripts- JavaScript.

Server-side Script- Java Server Pages.

Database- My SQL 8.0

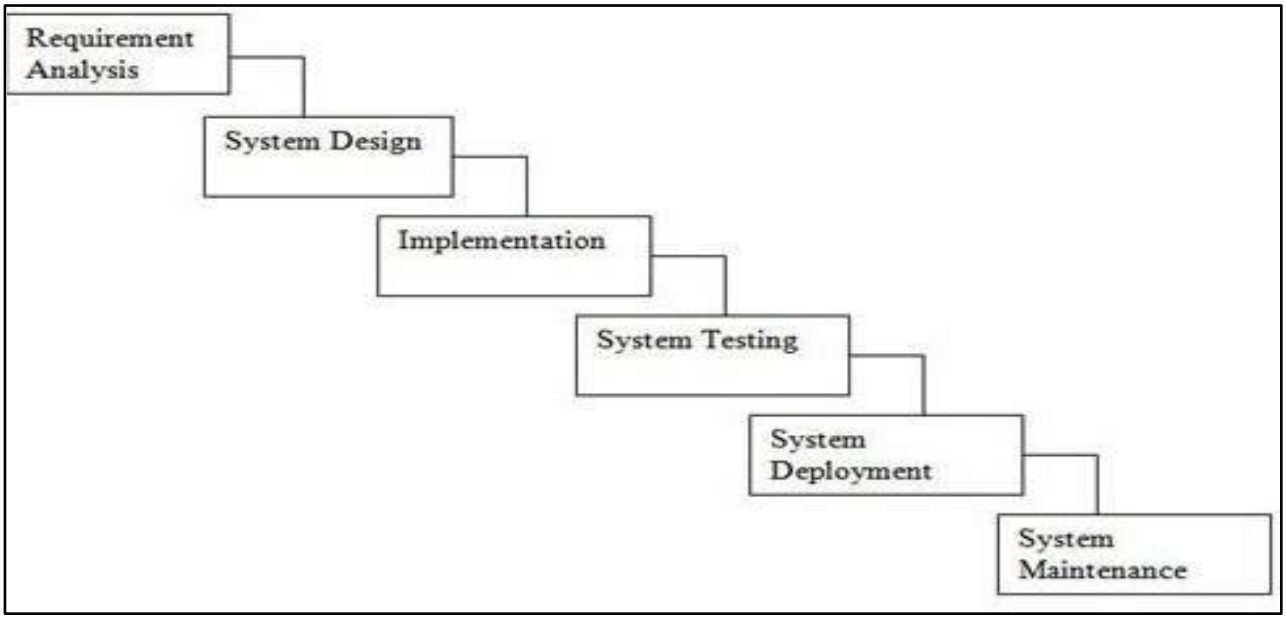
Because of availability of High-Performance Libraries
Operating System : - Windows 7/8/10/
Ubuntu18.04 LTS /20.04 LTS
Latest Operating System that supports all type of installation
and development Environment.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125240



SET-UP

To set up the environment for the proposed system, follow these important steps:

Task 1: Download and install JDK 1.8 or higher

Ensure that you have the Java Development Kit (JDK) version 1.8 or a newer version installed on your system. You can download the JDK from the official Oracle website and follow the installation instructions provided.

Task 2: Set up database connectivity using JDBC and MySQL bridge drivers

Configure the database connectivity for the system. Install the appropriate JDBC driver for MySQL, which allows the application to communicate with the MySQL database. You can download the JDBC driver from the official MySQL website. Once downloaded, follow the instructions to set up the driver and establish the database connection.

Task 3: Configure project properties in Eclipse IDE

Open the Eclipse IDE and navigate to the project properties by right-clicking on the project. Select the "Run" tab and set the main page or entry point for the application. This ensures that the correct file is executed when running the project.

By completing these tasks, you will have set up the necessary environment for the proposed system. This includes installing the JDK, establishing database connectivity using JDBC and MySQL bridge drivers, and configuring project properties in the Eclipse IDE.

❖ SDLC Model

The Waterfall model, known for its sequential and linear approach, has several applications:
1. Simplicity and Ease of Use: The Waterfall model is straightforward and easy to understand and implement. Its linear nature makes it accessible to both project managers and team members.
2. Manageability: The rigid structure of the Waterfall model makes it easier to manage projects. Each phase has well-defined deliverables and a review process, ensuring clear milestones and accountability.
3. Sequential Phases: The Waterfall model follows a sequential approach, where each phase is processed and completed one at a time. This eliminates overlap and allows for a systematic progression from requirements gathering to implementation, testing, and deployment.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125240

4. Well-Understood Requirements: The Waterfall model is suitable for projects with well-defined and stable requirements. It works effectively when the project team has a clear understanding of the desired outcome and the requirements are unlikely to change significantly during the development process.
5. Smaller Projects: The Waterfall model is often employed for smaller projects where the requirements are thoroughly understood and documented. It provides a structured approach that ensures a systematic and organized development process.

While the Waterfall model has its advantages, it may not be suitable for projects with evolving requirements or complex dependencies. Other models, such as Agile or iterative approaches, may be more appropriate in those scenarios.

Test cases:

To test the project problem statement, you can utilize various testing techniques and tools. Here are some suggestions:

1. Mathematical Models: If your project involves mathematical calculations or algorithms, you can create test cases based on known inputs and expected outputs. This can help validate the accuracy and correctness of your implementation.

2. GUI Testing: If your project includes a graphical user interface (GUI), you can perform GUI testing to ensure its functionality and usability. This can involve testing different user interactions, input validation, and responsiveness of the interface. Tools like Selenium or TestComplete can be used for GUI testing.

3. Function Testing Principles: Apply functional testing principles to test individual functions or modules of your project. This involves designing test cases that cover different scenarios and edge cases, verifying the expected outputs against the actual results. Unit testing frameworks such as JUnit (for Java) or pytest (for Python) can be helpful in automating the process.

4. Testing Tools: Depending on the nature of your project, you can select appropriate testing tools. For example, if you are developing a web application, tools like JMeter or LoadRunner can be used for performance testing. If you want to perform security testing, tools like OWASP ZAP or Nessus can be employed.

5. UML Diagram Reliability Testing: To test the reliability of UML diagrams, you can review them against the project's implementation and requirements. Ensure that the relationships, dependencies, and interactions depicted in the diagrams accurately reflect the actual behavior and structure of the system. Additionally, you can cross-reference the UML diagrams with the corresponding codebase to check for consistency and correctness.

Remember to document your test cases, execution results, and any issues encountered during testing. This will help in identifying and resolving potential problems in the project.

Test Case ID	Description	Test case I/P	Actual Result	Expected result	Test case criteria (P/F)
101	Enter the case insensitive Username click on Submit button.	Username	Error comes	Error Should come	Pass
102	Enter the case sensitive Username click on Submit button.	Username	Accept	Accept Username	Pass
201	Enter the case insensitive Password click on Submit				

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125240

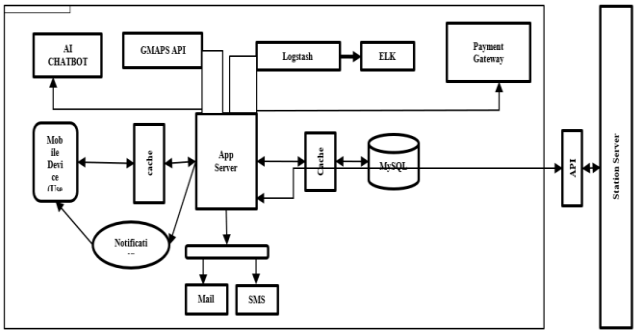
	button.	Password	Error comes	Error Should come	Pass
202	Enter the case sensitive Password click on Submit button	Password	Accept	Accept	Pass
301	Enter the case insensitive Mobile Number click on Submit button	Mobile Number	Error comes	Error Should come	Pass
302	Enter the case sensitive Mobile Number click on Submit button.	Mobile Number	Accept	Accept	Pass

Overview of Project Modules

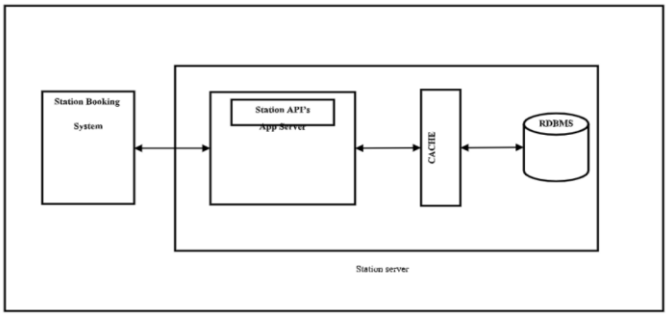
In this chapter we are going to have an overview about how much time does it took to complete each task like- Preliminary Survey Introduction and Problem Statement, Literature Survey, Project Statement, Software Requirement and Specification, System Design, Partial Report Submission, Architecture Design, Implementation, Deployment, Testing, Paper Publish, Report Submission. This chapter also gives focus on stakeholder list which gives information about project type, customer of the proposed system, user and project member who developed the system.

❖ **Design**

System Architecture



Booking System Architecture



Station System Architecture

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023
DOI: 10.17148/IJARCCE.2023.125240

In this paper, we study the function and impact of a small-sized superconducting magnetic energy storage (SMES) system in an EV charging station with a photovoltaic (PV) generation system. We also compare SMES with flywheel and capacitor energy storage systems. The SMES, PV generation system, and EV battery are connected to a common DC bus using converters. A voltage source converter (VSC) ensures grid connection and the SMES maintains the stability of the DC bus with its quick power response. To manage energy transfer among PV units, SMES, EV battery, and the power grid, an energy management strategy is designed for long-term operation. MATLAB/SIMULINK is used for modeling the EV charging station system, and simulation tests validate the function and performance of SMES.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

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

Integration testing

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

Functional test

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

Functional testing is centered on the following items:

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

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

Functions - identified functions must be exercised.

Output - identified classes of application outputs must be exercised.

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

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

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125240

System Test

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

White Box Testing

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

Black Box Testing

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

Unit Testing:

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

Test strategy and approach

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

Test objectives

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

Features to be tested

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

Integration Testing:

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

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

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

Acceptance Testing:

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

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

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

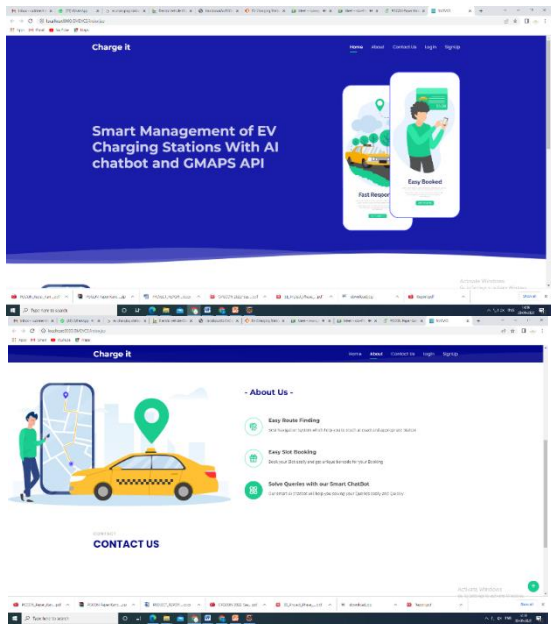
ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023

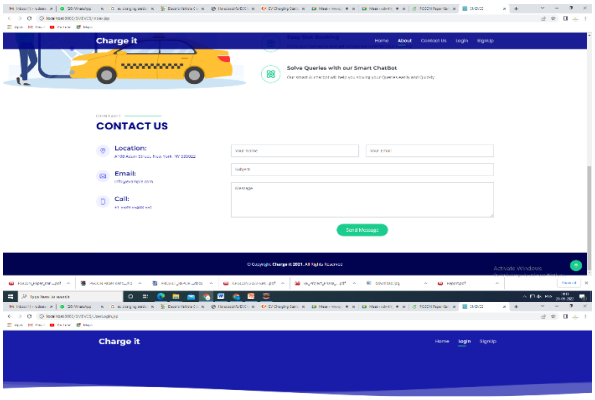
DOI: 10.17148/IJARCCE.2023.125240

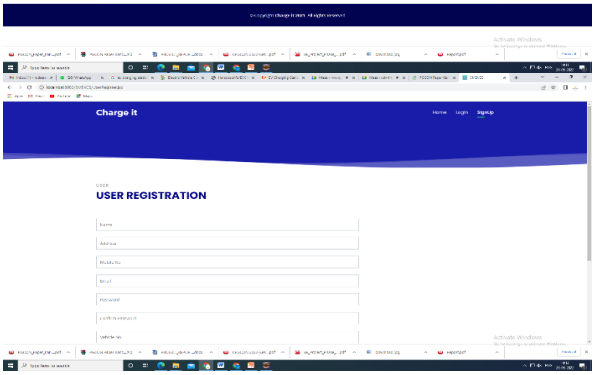
Test cases:

Testing of project problem statement using generated test data (using mathematical models, GUI, Function testing principles, if any) selection and appropriate use of testing tools, testing of UML diagram's reliability.

OUTPUT







International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023
DOI: 10.17148/IJARCCE.2023.125240

CONCLUSION

We have successfully developed the "Smart Management of EV Charging Stations" system using a hybrid approach to Android application development. One of the key functionalities of the system is the ability to book charging slots based on the specific type of charging socket required by the car.
To enhance user interaction and support, the system incorporates an AI chatbot capable of resolving queries and providing assistance. This chatbot ensures a seamless user experience and streamlines the process of accessing charging stations.
Additionally, the system utilizes the GMAPS API to provide accurate direction sensing. Users can easily navigate and locate the nearest charging station using this feature, enhancing convenience and efficiency.
In summary, the "Smart Management of EV Charging Stations" system encompasses a hybrid Android application approach. It offers charging slot booking based on charging socket types, an AI chatbot for query resolution, and utilizes the GMAPS API for efficient direction sensing. This comprehensive system aims to simplify EV charging station management and improve the overall user experience.

REFERENCES

1) Binod Vaidya1, Hussein T. Mouftah: Smart Electric Vehicle Charging Management for Smart cities: IET Research Journals, The Institution of Engineering and Technology 2015

2) Pooja Singh, Pinki Nayak, Arpita Datta, Depanshu Sani, Garima Raghav, Rahul Tejpal: Voice Control Device using Raspberry Pi: 2019 Amity International Conference on Artificial Intelligence (AICAI)

3) Subhash S, Prajwal N Srivatsa, Siddesh S: Artificial Intelligence Based Voice Assistant: 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4).

4) Heiko Knospe, Scarlet Schwiderski-Grosche: Online Payment for Access to Heterogeneous Mobile Networks: IST Programme under Contract IST-2000-25350.

5) Achmad fitro: Shortest Route at Dynamic Location with Node Combination-Dijkstra Algorithm: 978-1-5386-8402-3/18/$31.00 ©2018 IEEE

