chargeEase - EV charging station App

CSD 334 MINI PROJECT

19 MDL21CS033 Arun Shaji
27 MDL21CS055 Irene Tresa Mathews
37 MDL21CS074 Lolith Thomas

B. Tech. Computer Science & Engineering

Department of Computer Engineering
Model Engineering College, Ernakulam
Thrikkakara, Kochi 682021
Phone: +91.484.2575370
http://www.mec.ac.in
hodcs@mec.ac.in

May 2024

Model Engineering College, Ernakulam
Dept. of Computer Engineering

C E R T I F I C A T E

This is to certify that, this report titled ***chargeEase*** is a bonafide record of the work done by

19 MDL21CS033 ARUN SHAJI
27 MDL21CS055 IRENE TRESA MATHEWS
37 MDL21CS074 LOLITH THOMAS

Sixth Semester B. Tech. Computer Science & Engineering student, for the course work in **CSD334 - Mini Project** which is the Mini Project Work, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, B. Tech. Computer Science and Engineering of **APJ Abdul Kalam University** .

Guide

Name
Designation
Computer Engineering

Coordinator

Name
Designation
Computer Engineering

Head of the Department

February 29, 2024

Dr. Binu V. P
Professor
Computer Engineering

Acknowledgements

We are profoundly grateful to Asst Prof. Murali Mohanan for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. We would like to express deepest appreciation towards Dr. Mini G, Principal, Govt. Model Engineering college, Dr. Binu V. P, Head of Department of Computer Engineering. At last we must express our sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped me directly or indirectly during this course of work.

LOLITH THOMAS
ARUN SHAJI
IRENE TRESA MATHEWS

Abstract

In response to the pressing need for sustainable transportation solutions, this project proposes the development of a user-centric mobile application aimed at revolutionizing the electric vehicle (EV) charging experience. Leveraging the cross-platform development framework Flutter, the application offers seamless compatibility across iOS and Android devices, ensuring widespread accessibility for EV owners. Backed by Firebase, a sophisticated database management system, chargeEase efficiently handles user profiles, charging station details, and real-time updates. The app serves as a comprehensive solution for EV owners seeking efficient charging options. Key functionalities include user profile management, real-time availability updates, personalized charging recommendations, and seamless coordination between charging station hosts and EV owners. By prioritizing user experience, accessibility, and sustainability, chargeEase aims to accelerate the adoption of electric vehicles and contribute to the reduction of traffic congestion and carbon emissions. Through the integration of cutting-edge technology and user-centric design principles, this solution paves the way for a more environmentally conscientious mode of transportation in the modern urban landscape.

Contents

Chapter 1

Introduction

chargeEase is a groundbreaking project aimed at transforming the electric vehicle (EV) charging landscape. Designed to address the challenges faced by EV owners, chargeEase offers a user-centric mobile application that simplifies and enhances the charging experience. By providing real-time station availability, personalized charging recommendations, and integrated payment systems, chargeEase aims to make EV charging convenient, efficient, and accessible for all. With its innovative features and commitment to sustainability, chargeEase is poised to revolutionize the way we charge our EVs, contributing to a cleaner, greener future for transportation.

1.1 Proposed Project

1.1.1 Problem Statement

In light of the increasing adoption of electric vehicles (EVs) and the critical need for sustainable transportation solutions, there exists a significant challenge in efficiently locating and accessing EV charging stations. Current solutions often lack comprehensive features and user-friendly interfaces, leading to frustration and inconvenience for EV owners. Additionally, the lack of real-time availability updates and personalized recommendations exacerbates the issue, hindering the widespread adoption of EVs. Therefore, there is a pressing need for a user-centric mobile application that leverages modern technologies such as Flutter and Firebase to streamline the process of finding, reserving, and navigating to nearby charging stations. This app should prioritize user experience, accessibility, and sustainability, aiming to revolutionize the EV charging experience and accelerate the transition to

electric vehicles worldwide.

1.1.2 Proposed Solution

chargeEase presents a holistic solution to the challenges faced by electric vehicle (EV) owners in locating and accessing charging stations efficiently. Leveraging modern technologies such as Flutter and Firebase, chargeEase offers a user-centric mobile application that streamlines the EV charging experience. With a comprehensive database of charging stations, real-time availability updates, and personalized recommendations, chargeEase ensures that users can easily find and access charging spots tailored to their needs. The app's intuitive interface, seamless reservation and payment system, and community engagement features further enhance the user experience, fostering greater adoption of electric vehicles and promoting sustainability in transportation. By prioritizing user experience, accessibility, and efficiency, chargeEase aims to revolutionize EV charging and accelerate the transition to electric mobility worldwide.

Chapter 2

System Study Report

2.1 Literature Study

Chapter 3

Software Requirements Specification

3.1 Introduction

3.1.1 Purpose

ChargEase is an Android application designed to streamline the electric vehicle (EV) charging experience. It aims to simplify the process of locating, reserving, and using charging stations, thereby promoting the efficient utilization of charging infrastructure and contributing to the sustainable growth of electric mobility.

3.1.2 Document Convention

This document follows MLA (Modern Language Association) Format. The boldfaced text has been used to emphasize section and sub-section headings. Highlighting is to point out words in the glossary and italicized text is used to label and recognize diagrams. This SRS follows the universally accepted norms used for a formal document. There aren't many special conventions used, are listed as follows:

- Bold letters and words signify special importance to those words in that particular context
- Words which are in title case or uppercase in the middle on any particular sentence also

signify the emphasis on those words in the context of the topic We have also used multiple short forms in this SRS. The most important ones to be noted are mentioned as follows:

- UI : User Interface
- DBMS : Database Management System

3.1.3 Intended Audience and Reading Suggestions

This document contains software functionality, software and hardware requirements and user documentation.

- Developer: The developer who wants to read, change, modify or add new requirements into the existing program may need first to consult this document and update the requirements in an appropriate manner so as not to change the actual purpose of the system or make the system inconsistent.
- User: The user of this program reviews the specification provided in the document and checks to determine whether the software has all the suitable requirements and if the software developer has implemented all of them. The user can also consult the user guide in the event of any confusion for clarifications.
- Tester: The tester needs this document to prepare their test cases to validate that the initial requirements of this project is actually implemented in the deliverable. This document need not be read sequentially; users are encouraged to jump to any section they find relevant.

3.1.4 Project Scope

The scope of the project includes the development of an Android application that provides users with the ability to locate nearby charging stations, check real-time availability, reserve charging slots, and make payments for charging sessions. The application will also feature a user-friendly interface, secure authentication, and authorization mechanisms, and community engagement features.

3.1.5 Overview of Developer's Responsibilities

Maintain appropriate coding standards and design.

Contribute to technical design documentation.

Contribution in accordance with the software development lifecycle.

3.2 Overall Description

3.2.1 Product Perspective

ChargEase is a standalone application that operates on Android devices. It interacts with external systems, such as charging station databases and payment gateways, to provide users with real-time information and secure booking capabilities.

3.2.2 Product Functions

The primary functions of ChargEase include:Locating nearby charging stations Checking real-time availability of charging slots Reserving charging slots Making payments for charging sessions Providing a user-friendly interface Real-Time Updates and Notifications on Parking Availability Implementing secure authentication and authorization mechanisms Encouraging community engagement Secure payment gateway

3.2.3 User Classes and Characteristics

The application is designed for EV users who require access to charging stations. Users may vary in technical proficiency, but the application is intended to be user-friendly and accessible to all

3.2.4 Operating Environment

.

ChargEase operates on Android devices running Android OS version 12.0 or higher. It requires an internet connection to access real-time data and communicate with external systems.

- Database System: chargeEase utilizes a centralized database infrastructure to efficiently store and manage information related to Charging Stations, user profiles, and booking data.

- Client/Server Architecture: ParkShare operates on a resilient client/server model, enabling seamless communication between users and parking lot operators for efficient parking reservation processes.
- Operating System: The chargeEAase Android app is optimized for the Android operating system, providing a consistent and user-friendly experience across a wide range of Android devices.
- Database Management System: Supabase / Firebase offers a reliable and feature-rich database management system, ensuring efficient data storage, retrieval, and management for a responsive and seamless user experience.
- Platform: chargeEase's Flutter app leverages the capabilities of the Dart programming language and Flutter framework to deliver a modern and intuitive user interface. Technologies such as Dart, Flutter SDK, and Material Design contribute to the app's sleek and user-centric design, enhancing the overall user experience for drivers and parking lot operators alike.

3.2.5 Design & Implementation Constraints

- Regulatory Policies: NA
- Hardware Limitations: Devices must have Wi-Fi or mobile data that supports IEEE standards.
- Interfaces to other applications: NA
- Parallel operations: NA
- Audit functions: NA
- Control functions: NA
- Safety and Security Considerations:The system must prevent unauthorized users from accessing the files.
- Reliability Requirements: The total bugs in the system shall not exceed 1% of the total line number of code, except connection reliability, which is out of range.

3.2.6 User Documents

User documentation will be provided within the application, including a user guide and FAQs to assist users in navigating the application and understanding its features.

3.2.7 General Constrains

The application is subject to constraints related to Android OS compatibility, network availability, and external system dependencies.

3.2.8 Assumptions and Dependencies

The interface of the resulting system will be easy to use and accessible without a time or location constraint.

The users have access to proper internet connection.

The users have an Android phone.

It also depends on external systems, such as charging station databases and payment gateways, to provide real-time information and secure booking capabilities.

3.3 External Interface Requirement's

3.3.1 User Interface

- chargeEase's frontend User Interface is developed using Flutter, providing a seamless experience across different platforms.
- Components requiring User Interface: Landing Page, Signup Page, OTP Authentication, Searching page,Slot Booking, Data Manipulation, etc.
- The User Interface is optimized for Android devices, ensuring compatibility and responsiveness across different screen sizes and resolutions.
- Payment Integration:
  - Secure Payment Processing: Integration of a reliable and secure payment system for transactions within the application.
  - Transparent Pricing: Ensuring transparent pricing for users to compare and choose the most suitable options for their budget.

3.3.2 Software Interfaces

- **Flutter:** Flutter is a comprehensive UI toolkit developed by Google, designed to streamline the development of cross-platform mobile applications with a single codebase. It offers a rich set of features and capabilities to enhance developer productivity and improve user experience.
- **Material-UI:** Flutter's Material Design is a comprehensive UI framework that provides developers with a rich set of pre-designed widgets and tools for building visually appealing and responsive mobile applications. .
- **Supabase:** Supabase revolutionizes application development with PostgreSQL at its core, offering real-time functionality, authentication, RESTful API generation, serverless functions, and seamless third-party integration in a collaborative, open-source ecosystem.
- **Figma:** Figma redefines collaborative design with its cloud-based platform, empowering teams to create, prototype, and iterate on designs in real-time. Its intuitive interface, robust features, and seamless collaboration tools foster creativity and efficiency in the design process.
- **Visual Studio Code:** Visual Studio Code (VSCode) is a versatile and lightweight source code editor with robust features for efficient development. With support for various programming languages, extensions, and a user-friendly interface, VSCode significantly contributes to the development workflow of EnsembleHaven. erves as a powerful tool for EnsembleHaven's development team, foster
- **Android Studio:** Android Studio revolutionizes mobile app development, providing developers with a powerful integrated development environment (IDE) equipped with advanced tools and features. Its intuitive interface, comprehensive code editor, and seamless integration with the Android platform streamline the process of building high-quality Android applications.

3.3.3 Hardware Interfaces

The various Hardware Interfaces include :

- Devices enabled with good Internet Connection via Wi-Fi or Ethernet connection.
- Devices with at-least 1GHz of processing power.
- Devices of various form factors and sizes.
- Device with various input mechanisms such as touchscreens, keyboards, or voice input.

3.3.4 Communication Interfaces

The Communication interfaces include :

- **Email** : Purposed with official user registration and authentication.
- **Push Notifications** : Implementing a notification system to keep users informed about slot availability, reservation status updates, and other relevant information in real-time.
- **Real-Time Authentication Status** : chargeEase provides real-time updates and notifications to users and admins regarding the status of OTP authentication attempts, ensuring transparency and enhancing security throughout the authentication process.

3.4 Hardware and Software Requirements

3.4.1 Hardware Requirements

- Operating System: Android OS version 10.0 and above, ensuring compatibility with a wide range of Android devices.
- Ram: Minimum 1 GB RAM, providing sufficient memory to support the application's operations and ensure smooth performance.
- Processor: 1 GHz processor or better, enabling efficient processing of user interactions, data retrieval, and other app functionalities on various Android devices.
- Internet Connection: Minimum 500 Kbps bandwidth, facilitating seamless communication between the chargeEase app and its servers for real-time updates, data synchronization, and other online functionalities.

3.4.2 Software Requirements

Our software requirements are stated in 3 groups. First one is our software requirements for developing process; second one is software requirements for server side, last one is software requirements for client side : The "chargeEase" application is being built with the following technologies:

Development Requirements

- **Operating System:** Android Studio supports Windows, macOS, and Linux for development purposes.
- **Frontend Framework:** Flutter framework for building cross-platform mobile applications.
- **Frontend Libraries:** Material Design components provided by Flutter for consistent UI/UX across different devices.
- **Frontend Development Environment Tool:** Android Studio with Flutter plugin for efficient development and debugging.
- **Integrated Development Environment (IDE):** Android Studio provides a comprehensive development environment with features tailored for Android app development.

Server-Side Requirements

- **Backend Framework:** Supabase for providing backend services including authentication, database management, and serverless functions.
- **Database Management System:** Supabase utilizes PostgreSQL as its core database management system.
- **Server Deployment:** Supabase provides hosting services for deploying and managing the ParkShare backend services.
- **Caching:** Supabase offers caching mechanisms to optimize data retrieval and improve app performance.

Client-Side Requirements

Client-Side Requirements for the chargeEase Android Application entail the necessity of an Android device running Android OS version 10.0 or above, with a minimum of 1 GB RAM and a 1 GHz processor for optimal performance. The app should be compatible with various screen sizes, resolutions, and aspect ratios commonly found on Android devices, supporting touchscreen input for navigation and interaction. Adequate storage space is required for installing the chargeEase app and storing app-related data. Additionally, a stable internet connection, either through Wi-Fi or mobile data, is essential for accessing chargeEase servers, retrieving real-time data, and enabling features such as parking spot reservations and updates.

3.5 Functional Requirements

This section gives the details of the features and functions this system provides for various users and additional information on how each module works.

3.5.1 Locating Nearby Charging Stations

- Utilize GPS technology to determine the user's current location.
- Display nearby charging stations on a map interface.
- Offer options for filtering and sorting charging stations based on user preferences.

3.5.2 Checking Real-Time Availability of Charging Slots

- Integrate with charging station databases or APIs to fetch real-time data on slot availability.
- Display the availability status of each charging station or slot within the app.

3.5.3 Reserving Charging Slots

- Allow users to select a preferred charging station and reserve a slot for a specified time period.

- Provide confirmation of the reservation and display relevant details such as location, time, and cost.

3.5.4 Payment

- chargeEase will integrate a secure payment system to facilitate transactions for parking reservations. Users will be able to make payments seamlessly through the app using various payment methods.

3.5.5 Providing a User Friendly Interface

- Design a visually appealing and intuitive interface with easy navigation.
- Use clear and concise language to guide users through the app's features and functionalities.

3.5.6 Implementing Secure Authentication and Authorization Mechanisms

- Require users to sign in with credentials or use biometric authentication to access the app.
- Employ encryption protocols to protect user data and secure communication channels.

3.5.7 Encouraging Community Engagement

- Incorporate features for users to share their charging experiences, ratings, and reviews.
- Facilitate communication and interaction among users through forums, chat rooms, or social media integration.
- Offer incentives or rewards for active participation and contribution to the community.

3.6 Non - Functional Requirements

3.6.1 Performance Requirements

- **Response Time :** The application must respond to user interactions, such as locating nearby charging stations, checking real-time availability, and reserving charging slots, within a reasonable time frame (e.g., less than 2 seconds).
- **Loading Time :** The application must load and display data, such as charging station information and availability, within a reasonable time frame (e.g., less than 5 seconds).
- **Transaction Processing Time :**The application must process payment transactions for charging sessions within a reasonable time frame (e.g., less than 10 seconds).
- **Network Latency :** The application must minimize network latency when communicating with external systems, such as charging station databases and payment gateways, to ensure a smooth and responsive user experience.
- **Battery Consumption :** The application must minimize battery consumption to ensure prolonged usage without draining the device's battery excessively.
- **Memory Usage :** The application must optimize memory usage to ensure efficient performance and prevent crashes or slowdowns due to memory constraints.

3.6.2 Security Requirements

- **Authentication :** The application must authenticate users securely, using strong encryption and secure communication protocols.
- **Authorization :** The application must authorize users to access specific features and data based on their roles and permissions.
- **Data Protection :** The application must protect user data and payment information using encryption and secure storage mechanisms.
- **Secure Communication :** The application must use secure communication protocols, such as HTTPS, to protect data transmitted between the app and external systems.

- **Payment Security :** The application must use a secure payment gateway to process payments for charging sessions.
- **Compilance :** The application must comply with relevant security standards and regulations, such as PCI DSS for payment processing.

3.6.3 Software Quality Requirements :

- Reliability: The system should ensure accurate and reliable real-time updates on job openings and eligibility criteria. The platform should have minimal downtime to ensure constant availability for both PC coordinators and students.
- Usability: The user interface should be intuitive and user-friendly for both PC coordinators and students. Navigation should be straightforward, and features should be easily accessible to enhance overall user experience.
- Security: The platform should implement robust security measures to protect sensitive information such as student profiles and eligibility criteria. Authentication and authorization mechanisms should be in place to ensure data privacy and prevent unauthorized access.
- Performance: The platform should have fast response times, especially during peak usage periods, to provide a smooth experience for users. The AI tool's resume analysis should be efficient, providing quick and accurate results..
- Integration with AI Services: The AI tool's integration should be seamless, ensuring accurate and efficient analysis of student resumes. Compatibility with various AI services and frameworks should be considered for future enhancements.

These software quality attributes aim to ensure that the Easy Hire platform is not only functional but also reliable, secure, and user-friendly for both administrators and end-users.

3.6.4 Other Requirements

chargeEase is a dynamic platform that requires ongoing maintenance due to its continuous evolution. Refactoring is essential to keep the codebase efficient and adaptable to changing needs. Given the dynamic nature of

the cultural field, requirements may evolve, necessitating flexibility in the software architecture. Regular updates and responsiveness to changing user demands are crucial for the sustained success of chargeEase.

Chapter 4

Future Scope

4.0.1 Future Scope