DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SHARDA SCHOOL OF ENGINEERING & TECHNOLOGY

SHARDA UNIVERSITY, GREATER NOIDA

PEER TO PEER SELF DRIVING CAR RENTAL

A project submitted

in partial fulfilment of the requirements for the degree of

Bachelor of Technology in Computer Science and Engineering

By

Zarak Jahan (2019004410)

Manav Chauhan (2019640747)

Nazia Parween (2020002030)

Supervised by:

Dr Megha Chhabra (Associate Professor)

May, 2023

CERTIFICATE

This is to certify that the report entitled **"PEER TO PEER SELF DRIVING CAR RENTAL"** submitted by **"**ZARAK JAHAN (2019004410), MANAV CHAUHAN (2019640747) AND NAZIA PARWEEN (2020002030)**"** to Sharda University, towards the fulfilment of requirements of the degree of **"Bachelor of Technology"** is the record of bonafide final year Project work carried out by them in the **"**Department of Computer Science & Engineering, Sharda School of Engineering and Technology, Sharda University**"**.

The results/findings contained in this Project have not been submitted in part or full to any other University/Institute forward of any other Degree/Diploma.

Signature of the Guide

Name: Dr. Megha Chhabra

Designation: Associate Professor

Signature of Head of Department

Name: Prof. (Dr.) Nitin Rakesh

Place: Sharda University

Date:

Signature of External Examiner

Date:

ACKNOWLEDGEMENT

A major project is a golden opportunity for learning and self-development. We consider ourselves very lucky and honored to have so many wonderful people lead us through the completion of this project.

First and foremost we would like to thank Dr Nitin Rakesh, HOD, CSE who allowed us to undertake this project.

Our grateful thanks to Dr Megha Chhabra for her guidance in our project work. Dr Megha Chhabra, despite being extraordinarily busy with academics, took a timeout to hear, guide and keep us on the correct path. We do not know where we would have been without her help.

The CSE department monitored our progress and arranged all facilities to make life easier. We choose this moment to acknowledge their contribution gratefully.

Name and signature of Students:

Zarak jahan (2019004410)

Manav Chauhan (2019640747)

Nazia Parween (2020002030)

ABSTRACT

Peer-to-peer (P2P) self-driving vehicle sharing is one of the innovative transportation systems emerging today, however, fewer people are aware of it or its benefits. This transportation system is intended for people who stand to gain financially from using idle cars to supplement their income. In this report, we present a peer-to-peer self-driving auto rental Android app that lets users sign up as Renters or Hosts, with the latter having the ability to advertise their vehicle for hire. When creating for Android, we utilized Flutter, Android Studio, and Firebase for backend and database administration. The graphical user interface (GUI) and the operation of the software system have been discussed in addition to the system architecture and programme design. For future, field testing and development of the iOS application of the project is proposed.

Keywords – Peer-to-Peer, Self-Driving, Host, Flutter, GUI, IOS, Field Testing.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

1.1 Problem statement

A trusted and secure peer-to-peer car rental system to put the extensively available but still untapped resource of unutilized cars to use.

1.2 Project Overview

India is growing in many sectors like the car industry as well which has caused an increase in prices of vehicles as well as an increase in traffic [1], it not only affects the working man of a country by wasting his/her time in traffic jams but also the environment [2]. The government has taken various initiatives to solve this problem like encouraging people to use public transport, sharing vehicles etc. A new sector emerged to solve this problem i.e. the car rental industry. As the growing prices and population of India, it has become difficult to own a car or if you have a car its maintenance has become expensive. The younger generation of the age group 19-28 [3] in this country has preferred to use self-driving rental cars instead of owning a vehicle which is a liability, thus giving rise to companies like Zoom cars, OLA rentals [4] etc. These companies aim to solve the problem of using a car without owning it and paying for its maintenance, making them popular in no time. But they aim at one group who doesn't own a car or can't afford it, neglecting the one with unused cars. This project aims at solving this problem by the introduction of Peer To Peer self-driving car rental system which permits the vehicle owner to bring in additional cash on days when they are not utilizing their vehicles, similar to Airbnb. Therefore helping people with cars to turn their liability into an asset.

Due to the global progress of virtual and networking technologies over the past several years, peer-to-peer consumption has seen a substantial rise in popularity. One of the best examples of the P2P business model in use today is the car-sharing [5] sector. Customers can rent a car from a company or a private owner using this P2P business model in exchange for payment at the going rate. In this project, we created an Android application based on the P2P business model. The app includes features that allow users to post their automobiles for rent and get money in return.

1.3 Contribution

The contributions of this project are:

- To find and study the feasibility of the project and propose a hybrid solution i.e. having our own fleet of cars for rent in addition to P2P rentals.
- To develop an application for the proposed hybrid solution named as **SERIGO.**
- To provide additional features in the like car services.
- To publish one conference paper and one journal paper.

1.4 Expected Outcome

In consideration of the historical and rapid development of car rental companies, the way processes in the companies are taking place today is quite problematic, The current industry of self-driving car rental systems is dominated by companies like Zoom cars, OLA rentals [4] etc. which help the people who don't want to hustle with buying cars/liabilities but are not at all helpful to that part of the population who can benefit from having a car by giving it up for rent directly to the renter without any middleman, this project is planned to ease the processes of renting out your car through developing an effective and efficient self-driving car rental system, by connecting owner directly to the renter, just like other developed countries are using technologies towards facilitating their customer's processes through services like Turo Car Hire [6] etc.

1.5 Hardware and Software Specifications

Table 1.5.1 Hardware Specifications

S.NO	Hardware	Usage
1.	Processor I5/I7/I9	A lot of processing power is required while developing an android or IOS application. It is highly recommended to use the latest generation of Intel core processors.
2.	DDR4 RAM	16GB RAM is required for this project. It is highly recommended to use DDR4 RAMs due to their high cores and memory management.
3.	Storage	Solid State Drive (SSD) is required for this project as they make the operating system fast and response time low.

		In addition to that processing time is also reduced by the use of SSDs. It is highly recommended to SSDs of capacity more than 500GB. NVMe SSDs are also highly recommended for storage and processing speeds.

Table 1.5.2 Software Specifications

S.NO	Software	Usage
1.	Flutter	Flutter is a development tool made by Google to develop android, IOS applications, MacOS, Microsoft Windows and Google operating system apps and also web apps from the same code.
2.	FlutterFlow	It helps create amazing User Interface. Generates code. Provides support for APIs and database integration all through Web.
3.	Integrated Development Environment	It is highly recommended to use Visual Studio Code as the development environment VS CODE from extensive resources along with support for many languages. It also provides auto code completion which speed ups the development.
4.	Android Studios	Latest android studios is used to create a virtual environment for development of android applications. It provides support for latest android versions and screen types to test and run applications.

1.6 Non Functional Requirements

These non-functional requirements are required to understand constraints put on the system. These are estimated but not actually calculated or estimated using different measures.

Table 1.6.1 Non-Functional Requirements

S.NO	Non-Functional Requirement	Usage
1.	Capacity	The current requirement of storage of the project is estimated to be more than 1GB. It can scale up in future depending on the number of functionality added to the project.
2.	Compatibility	The minimum requirements of the application to run is estimated to be more than 4GB of RAM and 32GB of storage. This is right now only being developed for android operating system.
3.	Robustness	The project has not undergone testing phase hence failure time is yet to be calculated.
4.	Scalability	The project has not been yet tested for heavy working conditions.
5.	Usability	The user interface is being developed as simply as possible. User testing is yet to commence.

1.7 Summary

In this report, we aim to bring to knowledge of reader the current phase and advances in the development of the project. The project developed has been defined in the report followed by the overall definition of the problem along with an insight into the technicality used in the project. This report also provides information regarding the already done work in tackling the problem. The methodology is proposed in this report along with system design and analysis followed by the results obtained and conclusions.

LITERATURE SURVEY

In this chapter, we've compiled research on the creation of applications for automobile rentals and the P2P business model's integration into the industry.

Ajinkya et al. [7] created a ridesharing application for this study with the ideas of speed, affordability, and safety in mind. Their intended audience was college students coming from remote areas with limited or no access to public transportation who may make money by offering their car for ridesharing while commuting to school. The software allows users to register themselves as either drivers or passengers, allowing them to coordinate transportation for one another within a certain geographic area and time window. The applications include a number of distinctive features, including price suggestions based on mileage, time, and distance as well as route suggestions based on the type of ride and the rider's past experiences.

In this study, Celesty et al. [8] analysed a range of literature articles on carpooling in order to find out about intelligent carpooling solutions that consumers may use. It was noted that most of the frameworks that were offered employed the Dijkstra and Matching algorithms to find the least cost course of action. They also provided readers with information on the benefits of carpooling.

A user may use an online bus-hailing service, enter their journey details, and then wait to be picked up once the bus had enough passengers, according to the ride-sharing model proposed by LIU et al. in their study [9]. The supplier distributes drivers to consumers after integrating the compatible ride requests. To improve the ride-matching service and raise the success rate of ridesharing, they developed both precise and approximative algorithms. Using a real-world dataset, it was discovered that the suggested framework may provide enhanced cost performance and on-demand bus services for each trip request. This methodology reduced traffic by 92% and 96% and petrol usage by 87% and 92% when compared to vehicle ridesharing and no ridesharing.

In this work, Pooyan and Jia [10] proposed a ridesharing model that dealt with the problem of online matching on general graphs in a way that discourages users from using their own automobiles instead of shared taxis. They created an algorithm that calculated the best time for waiting for arriving passengers, resulting in matching with the lowest overall overheads while increasing the number of partnerships. To test the created system, they used the NYC real-time dataset, which significantly reduced total overheads.

A useful smartphone app for carpooling with unique features like traffic anomaly detection and user location tracking was proposed by Binu and Viswaraj [11] in this work. As a security feature, location tracking was included to let users share their whereabouts with their relatives. The purpose of the anomaly detection feature was to save unnecessary travel time by analyzing previously recorded traffic anomalies, such as processions, accidents, road construction, etc., reported by others and rerouting the vehicle through another optimal route. In order to make anomaly identification foolproof, they provided a truth estimation strategy with a recursive EM algorithm. Users of the Android-based application were able to organize travels, track their positions, report abnormalities, and more thanks to the app's user-friendly design. Each client's GPS data was gathered and transmitted in real time to the database using the Google Map API to offer position tracking.

The focus of this study, conducted by Xiaoyi et al. [12], was personalized taxi-sharing, which took into consideration each user's chosen payment method, travel length, and waiting time. They ascertained the degree of satisfaction of each participant in the scheduling plan, and based on that information, they created two goals, MaxMin and MaxSum, to assess the effectiveness of the overall plan. They then developed a two-stage process to address this issue.

In this work, Raza et al. [13] suggested a novel way for college personnel and students in Middle Eastern countries to use the Smart Peer Vehicle Pooling System. In order to help developers build applications using the specified architecture, this article offers a business model. A linear programming technique was used to address the highlighted concerns, which not only helps to reduce traffic congestion and pollution problems but also helps to achieve sustainable mobility by decreasing individual expenses, fuel consumption, usability, and reliability. The Dijkstra Shortest Path Algorithm was shown to generate the greatest results, according to the study.

In this paper, Bilong et al. [14] offered a few real-time ridesharing methodologies and described several ways that might be used at various phases of the Filter and Refine framework in dynamic ridesharing.

Bin et al. [15] proposed SHAREK, a ride-sharing service that made an effort to solve the drawbacks of the previous approaches, in this study. The maximum price users of this application are prepared to pay as well as the maximum amount of time users are willing to wait before being picked up are both selectable options. Moreover, SHAREK provides price estimates that account for the distance of the trip as well as the detour the driver

must take to reach the user. Moreover, SHAREK employs a number of trimming techniques to provide customers with efficient and scalable drivers.

To lessen the population's overall travel distance, Armant and Brown [16] presented a mixed integer programming paradigm in their work. The third model outperformed the other two by orders of magnitude and could serve 50% more consumers in the same amount of computation time when compared to the other two mixed integer programming models that they provided. The third model reduced the search space by taking use of symmetry breakdown on the shifter.

The design principles, distribution, and cloud computing methodologies that Dejan et al. [17] believe any future global carpool and ride-sharing solution may employ in order to be highly scalable and widely adopted in order to successfully reach and service a global user base are described in this paper. They made an effort to satisfy the necessity for developing real-time carpool and ride-sharing solutions by using cutting-edge web development approaches. They believed that using an open-source JQuery, Apache Cordova [18] mobile development framework, together with other UI techniques like HTML5, would be the best course of action moving forward for developing mobile apps across multiple OS systems.

Shuo et al [19] proposal of a large-scale taxi ridesharing service was made in response to the growing demand for ridesharing. This software efficiently responds to requests made by taxi drivers in real-time while producing ridesharing itineraries that significantly reduce the overall distance travelled. They first suggest a taxi-searching algorithm that uses a spatiotemporal index to swiftly find possible taxis that are likely to match a client request. A scheduling algorithm is then recommended. The requested journey was added to the schedule of the taxi that could accommodate the request with the least amount of additional travel time after each possible taxi was checked.

X. Xing et al. [20] introduced a ride-sharing concept for short-distance travel inside metropolitan areas in this study that aims to handle unplanned ride-sharing requests from possible passengers with transport alternatives available on quick calls. The ride-sharing contracts are negotiated by the driver agent and the passenger agent in this multiagent system that was created. The results of experiments employing the multiagent-based simulation environment PLaSMA in Bremen, Germany, were clear.

2.1. Gap Analysis

We have summarized the research in Table 2.1.1 and used a gap analysis to identify any gaps in the study in this section.

Table 2.1.1 Gap Analysis

S. No.	Ref No.	Year	Gap Analysis	Features	Methodology
1.	[7]	2021	There is a shortage of information about efficiency in diverse geographic locations.	• Java for the development of Android. • Algorithms for suggesting the shortest routes. • Rider-driver interaction. • Machine learning techniques for recommending prices.	• Built a ride-sharing application. • Allowed users to register as either drivers or passengers. • Find each other rides within a certain location and time limit.
2.	[8]	2020	No evidence is offered to back up the claim.	• Euclidean distance was established by Dijkstra is mostly utilized for routing techniques.	• Evaluated a variety of studies in the carpooling. • Identified practical carpooling strategies that consumers may employ.
3.	[9]	2019	The success rate is predicated on conjecture.	• They solved the problems of low capacity and high cost. • Using a real-world dataset of 65,065 journeys, they assessed their proposed methodology.	• Devised a ridesharing programme. • Allowed anyone to use the bus service online. • They developed both rough and precise ride-sharing optimization methods.
4.	[10]	2017	Cooperative agents are used to solve the problem.	• Algorithm for optimization. • Used a dataset of actual NYC taxis.	• Developed an optimization method to cope with the problem of online matching.

5.	[11]	2016	The accuracy of the recursive approach is not supported by any data.	• Monitoring of locations using the Google API. • Identification of traffic anomalies. • A suggested EM recursive algorithm truth estimation technique was made to reduce inaccurate anomaly detection.	• Proposed a practical smartphone application for carpooling. • Features developed like traffic anomaly detection and user location monitoring.
6.	[12]	2016	Every preference is unique and based on fulfilment.	• Determined an estimate of each user's degree of satisfaction with the schedule.	• Centered on tailored taxi sharing, accounting for each user's preferred method of payment, the distance travelled, and the duration of waiting
7.	[13]	2016	Research is conducted in a particular area and is based on the number of employees and students utilizing parking spots.	• Dijkstra Shortest Path Algorithm was used. • Included features like the ability to reserve parking spaces.	• Proposed a special and clever carpooling system so that Middle East College personnel and students may share a trip to school.
8.	[14]	2016	The consequences of traffic or any other murky related concerns have not been	• To cut the cost of going through all the trip request data, the filter was used, and the framework was refined.	• Outlined several real-time ridesharing options. • Demonstrated a number of potential applications for the Filter and Refine framework in dynamic ridesharing.

			considered in the study.		
9.	[15]	2015	The assertion regarding the rider request response time is unsupported by any evidence.	• The maximum price range and waiting period that users are willing to put up with may both be specified by users. • Calculates charges based on the distance travelled and any diversions taken. • SHAREK employs pruning techniques to minimize the requirement for any short-path calculations.	• SHAREK is a suggested ride-sharing service that enables users to request rides and is scalable and efficient.
10.	[16]	2014	The consequences of obstructions have not been considered.	• Three mixed integers were established, with the third performing better than the other two.	• Presented a mixed integer programming approach to reduce the overall journey distance for the population.
11.	[17]	2013	Data from actual testing are provided. It uses web sockets, which android browsers do not support.	• HTM5 and CSS3 were used for the user interface. • Used NoSQL for the database and Web sockets for communication.	• The author identified the design principles, distribution plans, and cloud computing tactics. • Believed carpool and ride-sharing solution may use to be highly scalable and extensively.

12.	[19]	2013	The study requires sophisticated travel time estimation approaches to improve trip time prediction.	• Used a spatial-temporal based search algorithm to look for taxis that can meet passenger needs. • Used a GPS trajectory dataset that more than 33,000 taxis created over the course of three months.	• It was possible for customers to schedule ridesharing and make real-time requests to taxi drivers. • It minimized overall journey distances through the development of a comprehensive taxi ridesharing service.
13.	[20]	2009	There is currently no testing data on the experiment's real-world relevance; it is based on simple maths.	• Employed multiagent system.	• Presented a ride-sharing service for urban regions where users could directly negotiate with drivers.

2.2. Feasibility study

To determine the limitations of this business model, we looked at several parts of it. These restrictions were divided into three categories by us.

2.2.1 Competence of the people

Peer-to-peer carsharing was noted to be unfamiliar to the majority of the public, which makes them reluctant to take part in this innovative mode of transportation. In addition to this, many were reluctant to rent out their personal automobiles due to concerns about security, trust, and maintenance costs. Also, people have a tendency to view their automobiles as exclusive personal property. Because they don't want to bother with

purchasing and maintaining a personal automobile, the younger age group found the idea to be intriguing.

2.2.2 Self-drive automobile rental sector

Companies that hire automobiles, such as Uber and Zoomcars, have dominated the industry in recent years. By lowering the number of personal automobiles on the road, these businesses have significantly contributed to the solution of the growing traffic issue. According to surveys, persons between the ages of 25 and 35 are more likely to use these applications. While they have been successful in addressing the traffic issue, they have not been as successful in meeting the employment or affordability demands of people with low family incomes.

2.2.3 Economy

According to studies, the automobile rental sector has grown to be a multimillion dollar asset. Uber and Ola control the majority of the market in India. They have also generated a wide range of jobs. Yet, the cost of these leases is rising daily, making it difficult for the average person to keep up. People with modest yearly incomes have been seen to be interested in the peer-to-peer carsharing business model from an economic standpoint in order to supplement their income.

2.3. Summary

The P2P carsharing business model has been around for a while, but there have always been problems with it, some of which are listed in the feasibility study. In this part, we summarized the related work done in this and provided some gap analysis of the work done.

SYSTEM DESIGN& ANALYSIS

3.1 Methodology

We discussed some of the market's existing challenges, ideas, and solutions for ride-sharing and carpooling in the section above. As a result of those solutions and ideas, we are developing an Android application based on a peer-to-peer collaborative consumption model using Flutter, Android Studio, and Firebase.

3.1.1 Flutter

In May 2017 [21], Google created and released Flutter, a free and open-source mobile UI framework. With only one codebase, you can create a native mobile application. This suggests that you may create two different apps using the same programming language and codebase (for IOS and Android).

Flutter is composed of two crucial parts:

- Your apps will be created with the help of an SDK (Software Development Kit), a collection of tools. Also provided are tools for translating your code to native machine code (code for IOS and Android).
- A group of reusable user interface (UI) elements, such as buttons, text input fields, sliders, and other objects, that you may change to suit your needs is known as a Framework (UI Library based on widgets).

3.1.2 Dart

Apps may be made using both Flutter and Dart. Google developed this object-oriented language. As a JavaScript replacement, Dart was initially developed [22]. Dart, a powerful programming language, is customized to solve this issue in memory standard with the aid of "Generational Waste Accumulation [23]".

3.1.3 Android Studio

Android Studio is the name of the official Google Integrated Development Environment (IDE) [24] for developing Android apps. We can produce Android apps more fast thanks to Android Studio's additional capabilities.

Features of the Android Studio:

- Its build system, which is based on Gradle, is flexible.

- It comes with a speedy and feature-rich emulator for testing apps.
- Android Studio, which provides a uniform environment, may be used to build applications for all Android devices.
- Both NDK and C++ are compatible.

3.1.4 Firebase

With the Google-sponsored Firebase application development platform, developers produce IOS, Android, and Web apps. Firebase offers tools for tracking metrics, documenting and fixing app issues, as well as creating marketing and product trials. Data across all clients is synchronized in real-time through the Firebase Database, a cloud-based NoSQL database [25]. The Real-time Database instance is shared by all clients of cross-platform apps made with the JavaScript SDK for Android, iOS, and android [26] and instantly updates with the most current data changes. Firebase remains operational even when it is not connected to the servers because it retains user data when it is unplugged and updates it when the connection is made [27 - 28]. Firebase may be accessible via browsers and mobile devices as an application server is not necessary [29].

Firebase offers a variety of services, including the following:

- Data evaluation.
- System of secure authentication.
- Using cross-platform communications to access the cloud.
- Cloud-based, real-time NoSQL database.
- Crash reporting in real time.
- Service for monitoring Firebase Production.
- Lab for Firebase Testing.

3.2 Platform Architecture

The host module, where users may advertise their own vehicles for hire, and the basic renter module, where users can rent automobiles, form the foundation of the system design. As seen in Figure 1, the procedure for logging in, posting a vehicle, and completing a booking is outlined in this section. Table 2 also includes a mathematical justification for the functional interdependence of various objects. The system's structural arrangement has also been described using the algorithm for the host and renter.

3.2.1 Functional Dependency Graph

- Af = Log in to the system
- Bf = Including personal data
- Cf = List of Vehicles
- $C'f$ = Register as a Host
- $C''f$ = Type in the necessary vehicle details
- Df = Selecting a vehicle from the market
- Ef = Filling out the credentials
- Ff = Complete Payment

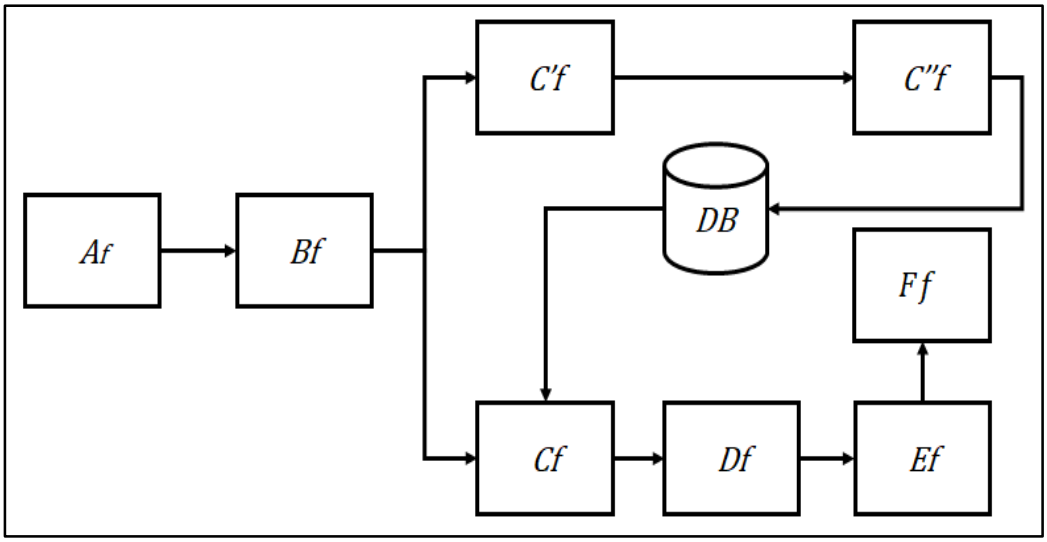


Figure 3.2.1.1. Functional Dependency Graph

3.2.2 Project functioning Pseudocode

The project is developed in two main modules: Renter and Host.

Table 3.2.2.1 Pseudocode for Renter and Host Functioning

$\partial = FilterVehicle(Cf)$ $C''f = Specs(\partial)$ $\Delta_s = Start\_date(\partial)$ $\Delta_e = End\_date(\partial)$ $\partial' = Df(C''f, \Delta_s, \Delta_e)$ $E'f = Ef(\partial')$ $G' = Ff(E'f, G'')$
$C'f = Host(Af)$ $F'f = Ef(C'f)$ $G'' = Verfiy\_Host(F'f)$

Table 3.2.2.1 describes the Pseudocode of the functioning of the application where $\boldsymbol{\partial}$ function filters the vehicle data by taking $\boldsymbol{Cf}$ as perimeter. $\boldsymbol{C''f}$ Specifies the vehicle data by taking $\boldsymbol{\partial}$ as the perimeter. The renter specifies start Δ_s and end Δ_e dates for the trip. $\boldsymbol{\partial'}$ Function maps all the entered perimeters into selecting the vehicle. $\boldsymbol{E'f}$ Function prompts user to enter the credentials by taking $\boldsymbol{\partial'}$ as the perimeter. Finally $\boldsymbol{G'}$ completes the booking through payment.

Similarly, $\boldsymbol{C'f}$ function registers the host by taking $\boldsymbol{Af}$ as the perimeter. The host need to enter the credentials $\boldsymbol{F'f}$ and finally the credentials are verified $''$.

3.2.3 Mathematical functioning of the project

The mathematical basis of the application is described in Table 3.2.3.1. Here, **"f(g)"** stands for the function of **g**, and we employed a number of functions with a wide range of inputs to obtain the desired results.

Table 3.2.3.1 Mathematical Mapping of Project Functioning

Mapping of App functioning f(g) -> h	Input (g)	Output (h)
$G1(d) \rightarrow G$ d – login data from the user G – homepage of the application	d	G
$G2\ (d, d1) \rightarrow G'$ G' - vehicle booked $d1$ – user details vehicle booking	$d, d1$	G'
$G3(G') \rightarrow DB$ DB – Vehicle Database update	G'	DB
$G4(d, d2) \rightarrow G''$ G'' - Host $d2$ – Host registration details	$d, d2$	G''
$G5(G'') \rightarrow G^x$ G^x – Vehicle advertisement post	G''	G^x
$G6(G^x) \rightarrow DB$	G^x	DB

The login functionality of the programme is represented by the **G1** function, which receives **d** as input, where **d** is the user's login details. The output **G** serves as a representation of the application's home page.

The inputs **d** and **d1**, which provide the user data required to reserve a car, are accepted by the **G2** function, where the user selects the vehicle for booking. **G'** stands for the reserved vehicle.

Using **G'** as its input and **DB** as its output, the **G3** function updates the vehicle database.

Users of the **G4** functionality can create hosting accounts. It will take the input functions **d** and **d2**, where **d2** represents the host registration data and **G"** represents the registered host account.

The **G5** function allows users to publish automobiles by taking **G"** as an input and producing **Gx** as an output, where **Gx** stands for the posted advertisement.

G6 refreshes the database by taking **Gx** as input.

3.2.4 Systematic Structural Design

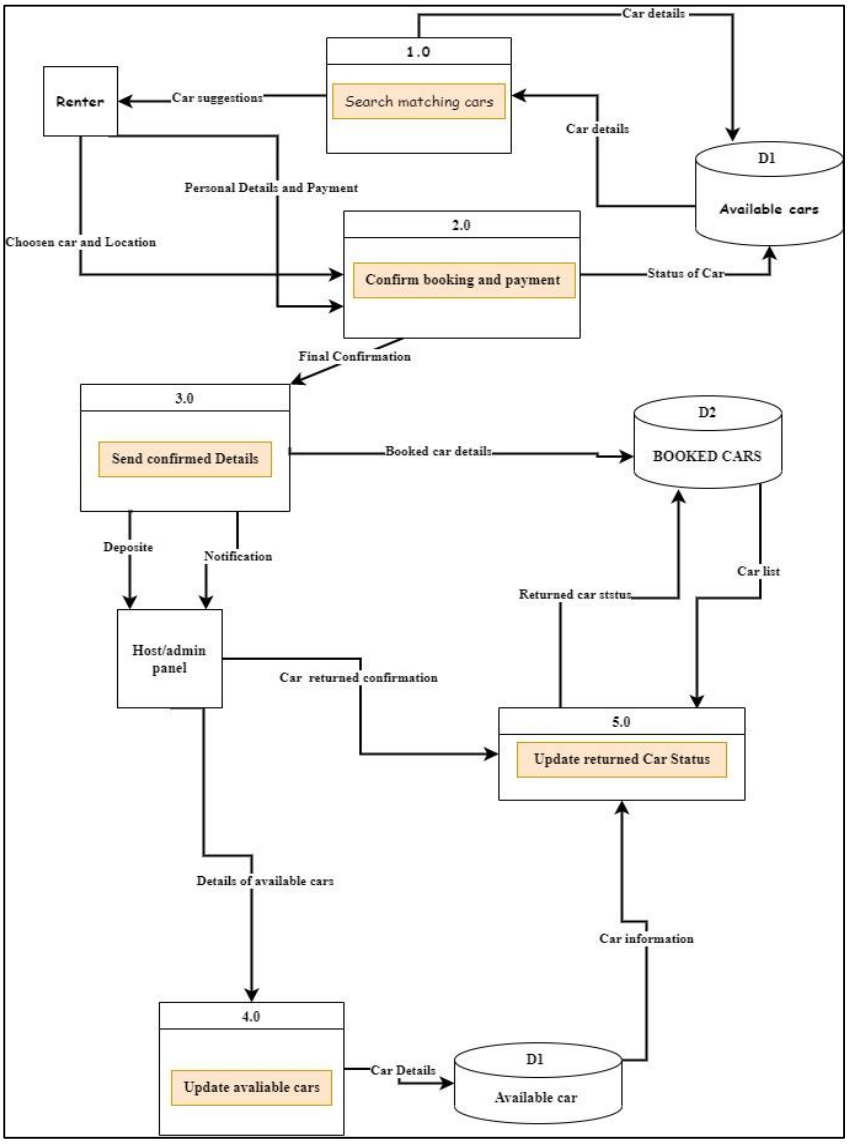


Figure 3.2.4.1 System Workflow Design

Figure 3.2.4.1 outlines the structural elements of the planned work in the architectural diagram in order to stimulate conversations about improvements and give a framework to

guide software development. In this Figure, it can be seen that Renter receives vehicle suggestion from 1.0, once confirmed the user can select the vehicle and location and submit personal details payment 2.0. All this is updated in database D1. On final confirmation, the information is send to host. On return of the vehicle the database D1 is updated.

3.2.4.1 Use Case Diagram

In the Unified Modeling Language, a use case diagram is a behaviour diagram [30-31]. It is also used to understand the working of the software system [32]. Figure 3.2.4.1.1 depicts the system's architecture as well as the relationship between the host and renter. The user base is extended into two categories in this system: user host and renter user after successful login. With the addition of listing their own car for hire, the host can examine and rent automobiles similar to the user renter. From car availability through booking confirmation, the admin oversees every procedure. In this Figure, the use case diagram of the project have been explained, in which it is seen that there are 3 types of users: Admin, Host and Renter. The solid arrows represent the function each type of user can access like select vehicle, available vehicle, book vehicle, book request, confirmation, post vehicle. The outwards extending dotted lines (<<include>>) represent the function that follows the main function and the inwards extending dotted lines (<<extend>>) represent the function that made up the main function.

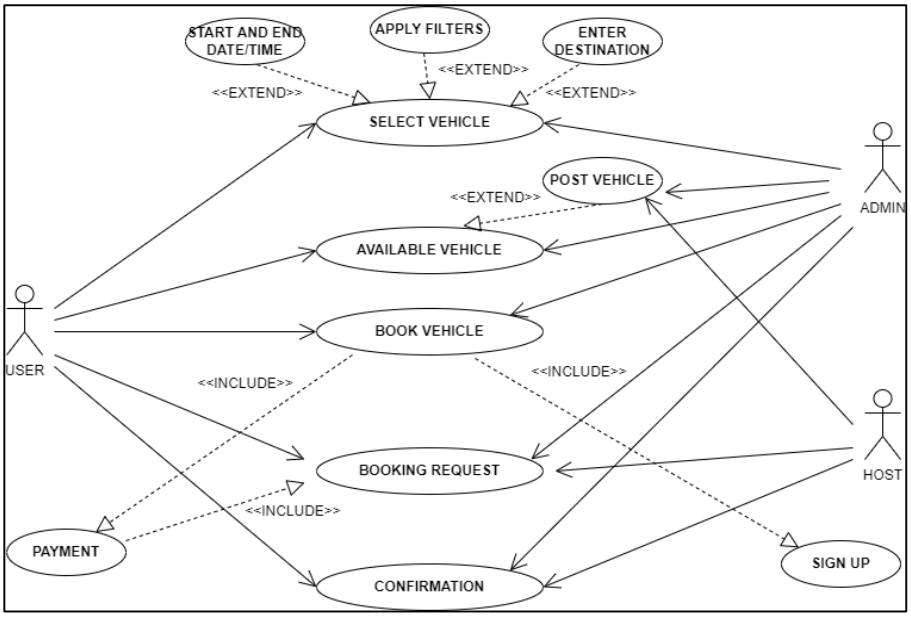


Figure 3.2.4.1.1 Use Case Diagram

3.2.4.2 Class Diagram

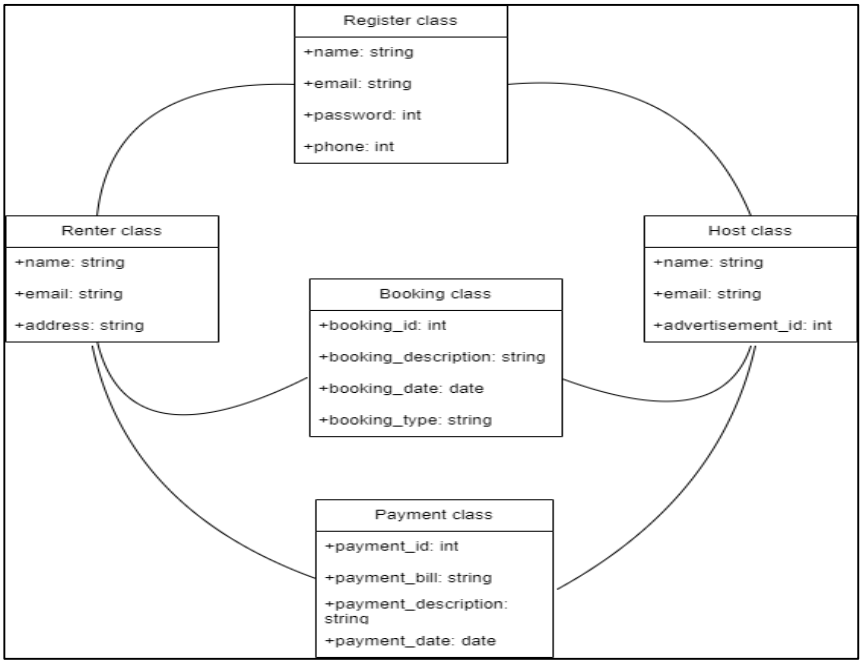


Figure 3.2.4.2.1 Class Diagram

Class diagrams are the most widely used UML diagrams [33]. The system's classes are described by the class diagram, as depicted in Figure 3.2.4.2.1. Classes in an object-oriented programme include relationships with other classes as well as attributes (member variables), actions (member functions), and other classes. In this Figure, it illustrates the different classes of the project like Register Class, Renter Class, Host Class, Booking Class, and Payment Class. Each class consists of members which have variables assigned to it like member variables: name, email, address, booking_description, booking_type, payment_bill, payment_description - can only be a string, similarly, member variables: password, phone, booking_id, payment_id, advertisement_id – can only be an integer, and member variables: booking_date, payment_date – can only be a date.

3.2.4.3 Flow Chart

Unlike texts, a flowchart is a graph that comprises multiple types of structures, such as loop, selection, and others [34]. In the study of system requirements, preliminary design, and thorough design for all UML, flowcharts are crucial [35]. From registration until logout, the system's flow is depicted graphically in Figure 3.2.4.3.1. The flow chart describes the extension of the user into Host and renter and their functionality. The host can post advertisements for their vehicle and can modify them. The renter can book the vehicle and make payments.

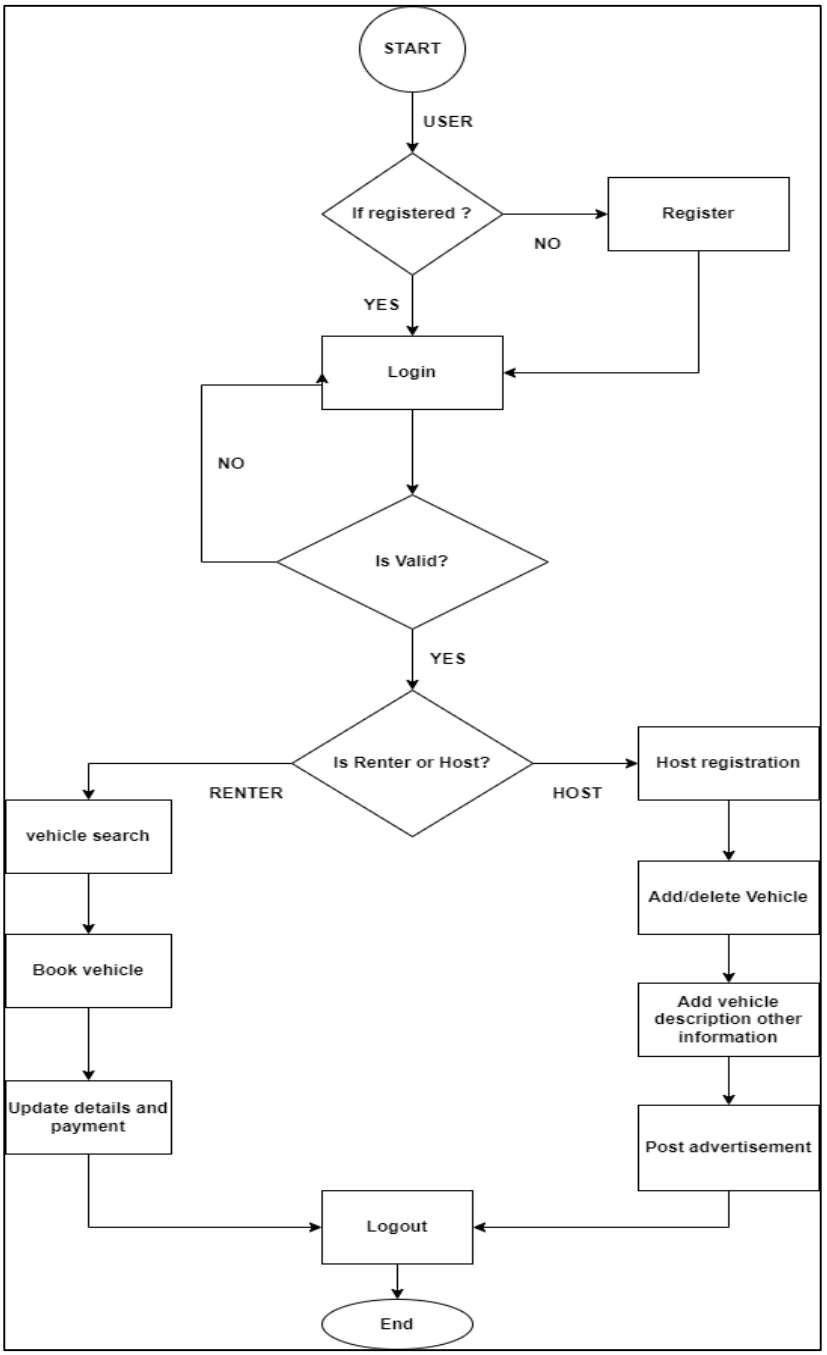


Figure 3.2.4.3.1 Flow Chart

In this Figure, the flow of login and booking is illustrated. The user needs to register before he/she can log in. The logged-in details are validated if authentic the logs in otherwise they are asked to log in again. On successfully logging in the user can either be a renter or a host. If a renter, he/she can search for a vehicle, book vehicles, and update details and payments. If the host, he/she needs to first register for the host, then add or delete the vehicle, add the vehicle description and other pieces of information and post the advertisement of the vehicle. Then log-out.

3.2.5 Proposed Algorithm

The application is divided into two primary sections. The application's main user component allows users to log in and hire the car of their choosing by filtering and demonstrating vehicle parameters, as well as the start and finish dates. By asking the user to provide extra information and upload documents in addition to their login information, supplementary user identification is carried out **E'f**. According to Algorithm 1, the money is sent to the Host **G'** to reserve the car.

Algorithm1: User login into the app and rent a vehicle

Successful Login Af

$\partial = FilterVehicle(Cf)$ % Apply filter for the vehicle required %

$C''f = Specs(\partial)$ % Provide vehicle specifications %

$\Delta_s = Start\_date(\partial)$

$\Delta_e = End\_date(\partial)$ % Add Start date and end date %

$\partial' = Df(C''f, \Delta_s, \Delta_e)$ % Select the vehicle required %

$E'f = Ef(\partial')$ % Provide the details and documentation%

$G' = Ff(E'f, G'')$ % Make Payment to verified host G'' and vehicle is booked%

Algorithm2: Host Registration

Successful Login Af

$C'f = Host(Af)$ % Apply Host Login %

$F'f = Ef(C'f)$ % provide details & Documents %

$G'' = Verfiy\_Host(F'f)$ % Host H verified and account opened %

Inside the programme, users can select the application's second component (Host). The primary user can register to become a Host **C'f** following a successful login, at which point they are requested to provide vehicle information and submit papers **F'f**. The vehicle is then published in accordance with Algorithm 2 once the host account has been authenticated **G'**.

3.3 Summary

We have explained the techniques utilized to structure the project in this section of the paper. It illustrates the project's working architecture and other elements utilizing use case diagrams, class diagram and flowchart.

RESULTS AND OUTPUTS

4.1 Proposed Project Output

In this chapter, we have illustrated and described the functional techniques of the project like Firebase, API keys used to integrate the project with Google maps, the application screen and the test cases designed to check the errors and faults in the project.

4.1.1 Implement Specification

In this section, the firebase implementation and usability is discussed.

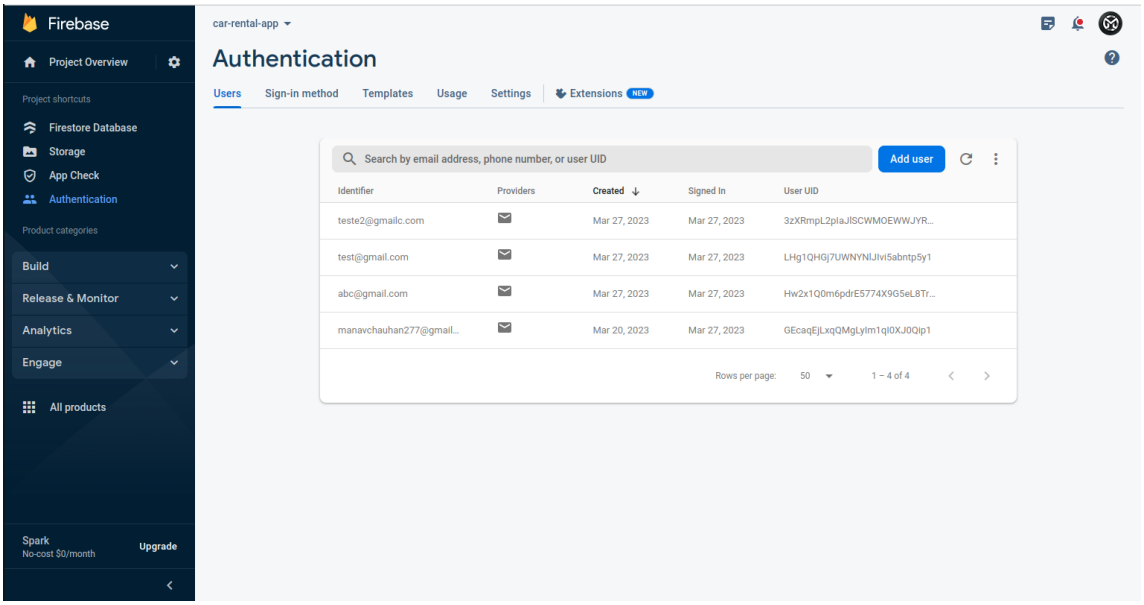


Figure 4.1.1.1 Authentication Panel of Firebase

Figure 4.1.1.1 illustrates the authentication panel of the application in the firebase. This authentication module keeps track of the registered user and provides admin the flexibility to manage, delete, update and authenticate user data. The authentication panel consists of email ids of the registered user, date of creation of account and signed in date and user UIDs.

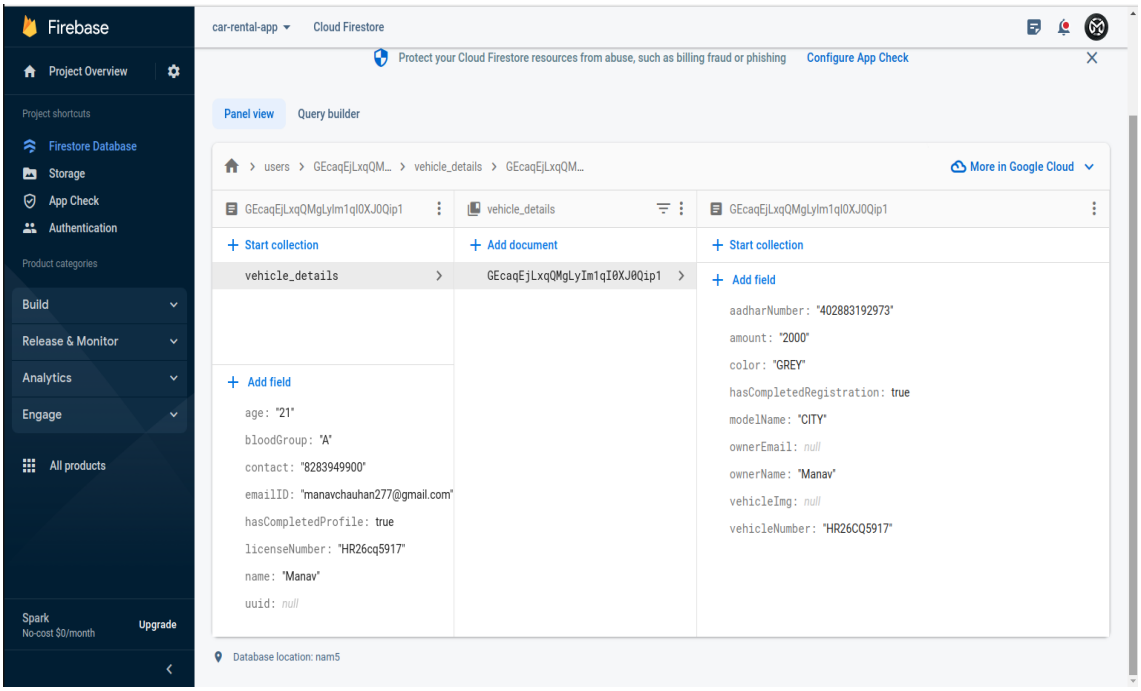


Figure 4.1.1.2 Vehicle Data Panel of Firebase

Figure 4.1.1.2 illustrates the vehicle data in the firebase module. Here all the vehicle data is managed by the admin. The vehicle data consists of aadhar number, amount, color, has Completed Registration, model Name, owner email, owner name, vehicle image, and vehicle number.

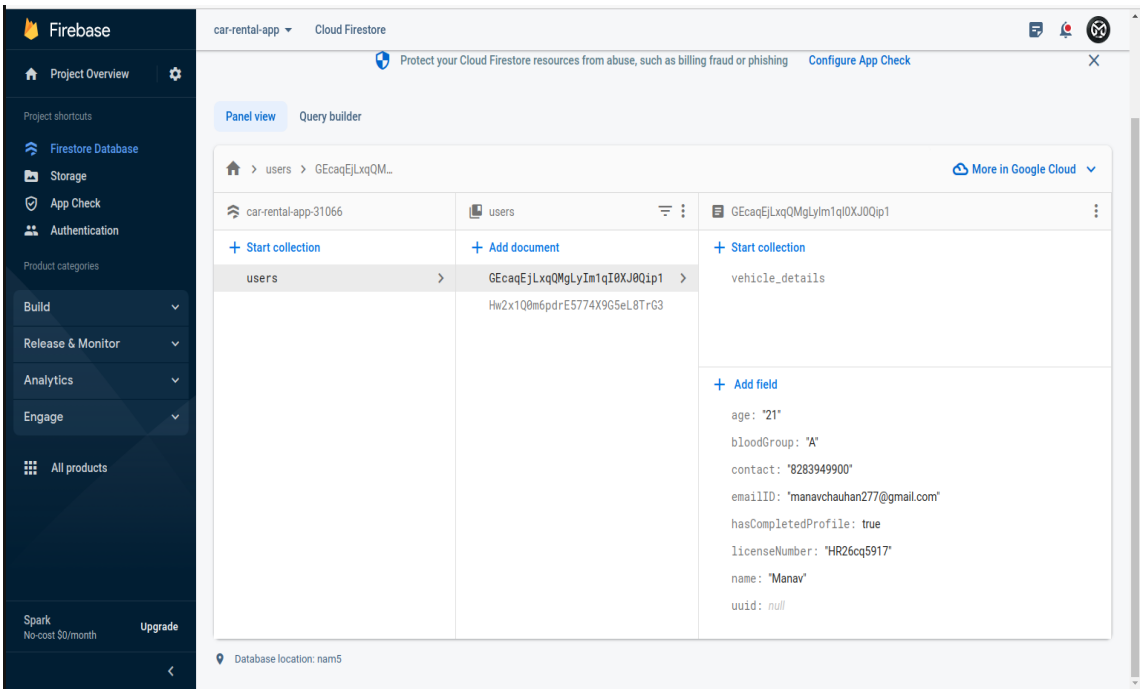


Figure 4.1.1.3 User Data Panel of Firebase

Figure 4.1.1.3 illustrates the user data module in the firebase. It manages the number of users in the application. It consists of data like age, blood group, contact, email ID, and has completed Profile, license number and name.

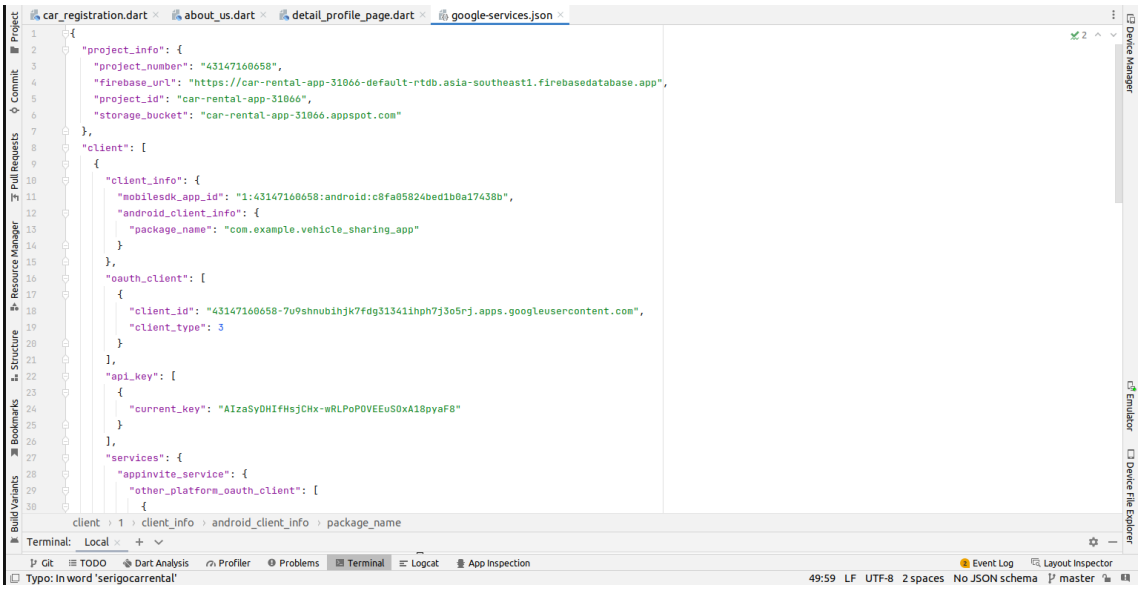


Figure 4.1.1.4 API keys used to connect to Google Services

Figure 4.1.1.4 illustrates the API keys used in the project to connect to Google services like Google maps.

4.1.2 Flow of Implementation

Figure 4.1.2.1 – 4.1.2.3 illustrates the flow of each screen to its corresponding screens. It shows the dependency of each module upon other and also help in debugging to find if there is any unassigned function.

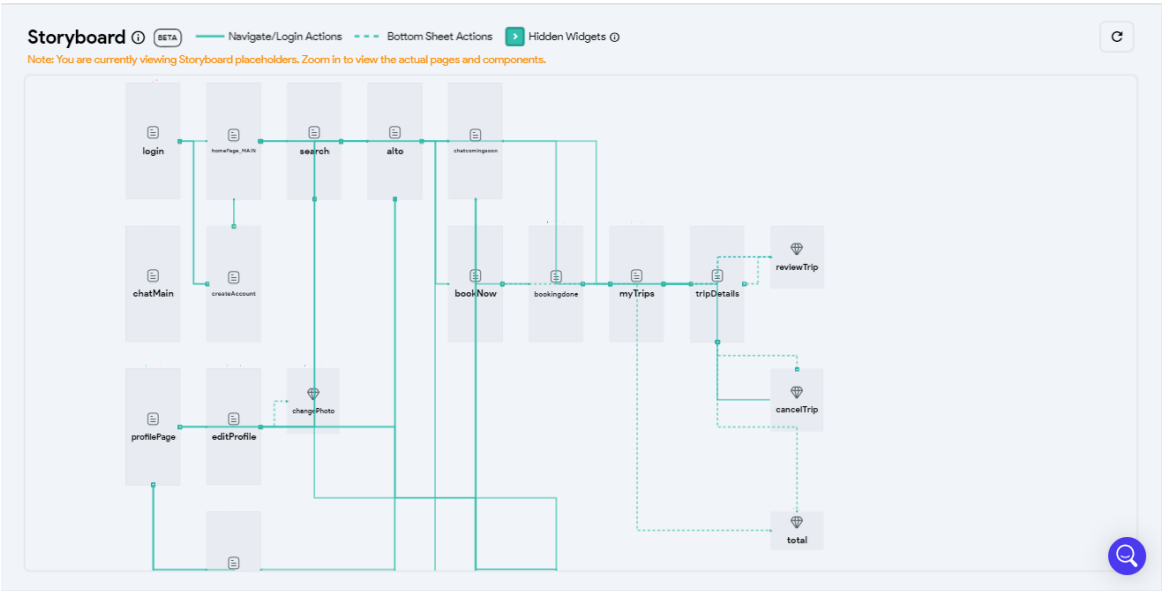


Figure 4.1.2.1 Flow of Application Screens part 1

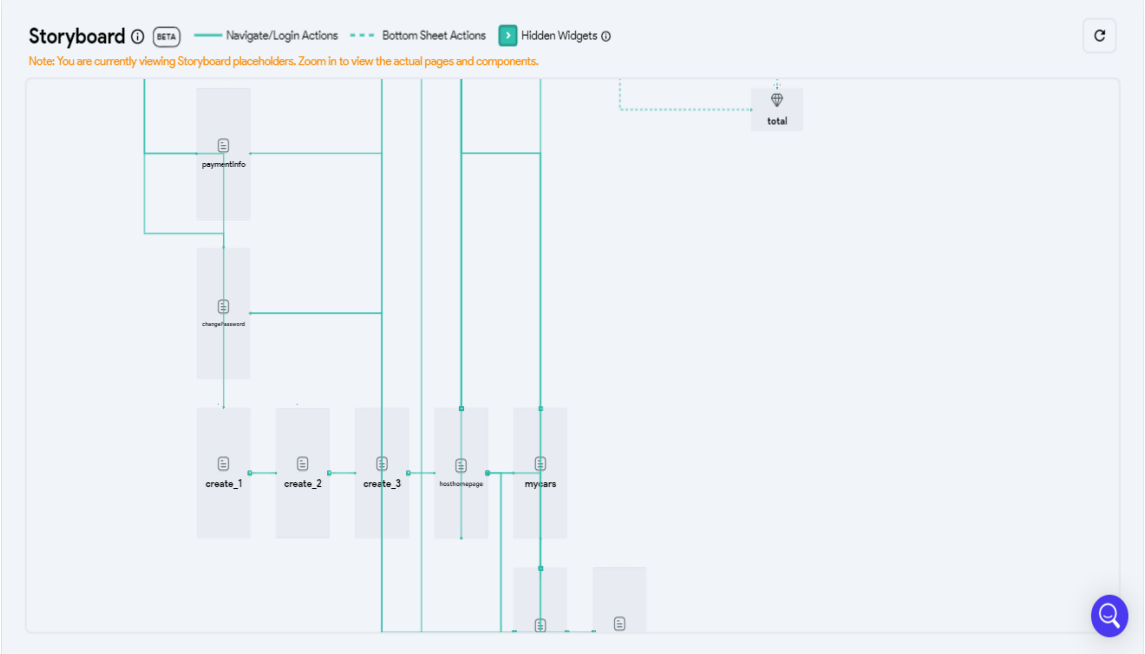


Figure 4.1.2.2 Flow of Application Screens part 2

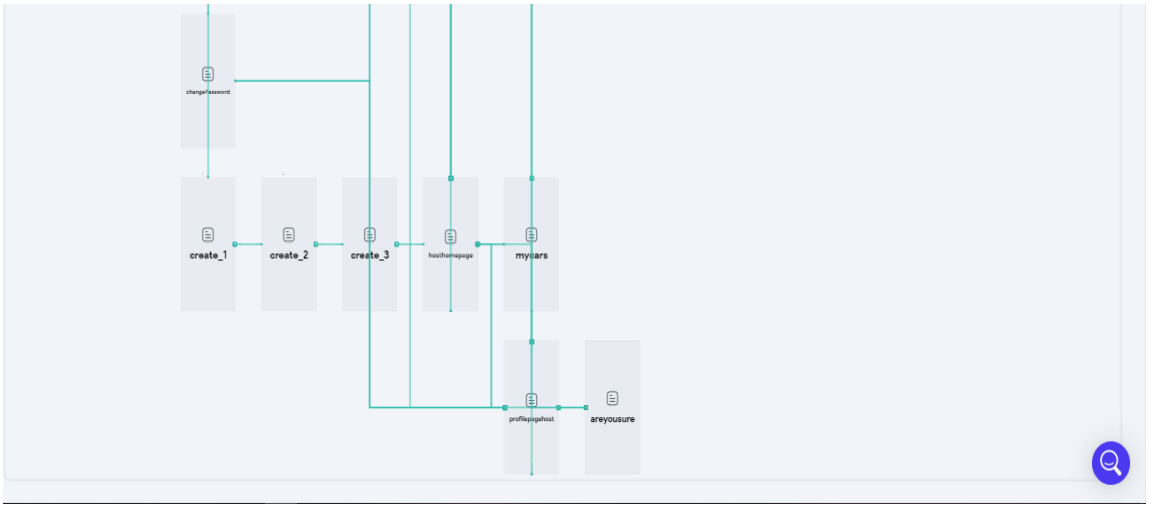


Figure 4.1.2.3 Flow of Application Screens part 3

4.1.3 Critical System Modules

The system modules illustrates the screens of the application and the code used to run them. In this section, we describe the user interface of the developed modules and their functioning w.r.t to the application.

Figure 4.1.3.1 Login Screen of the Application

Figure 4.1.3.1 illustrates the first screen of the application showing the login screen where the user who already has an account can log in. It also shows the code employed in the development of the screen.

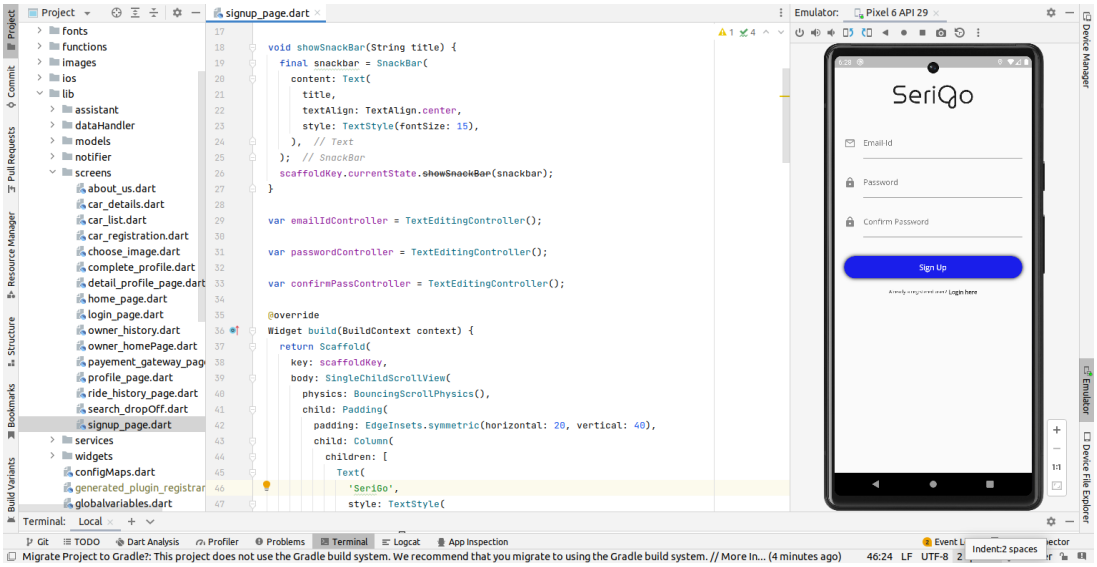


Figure 4.1.3.2 Create Account Screen of the Application

Figure 4.1.3.2 illustrates the create account screen of the application which is used by the user to register an account in the application. It also shows the code employed in the development of the screen.

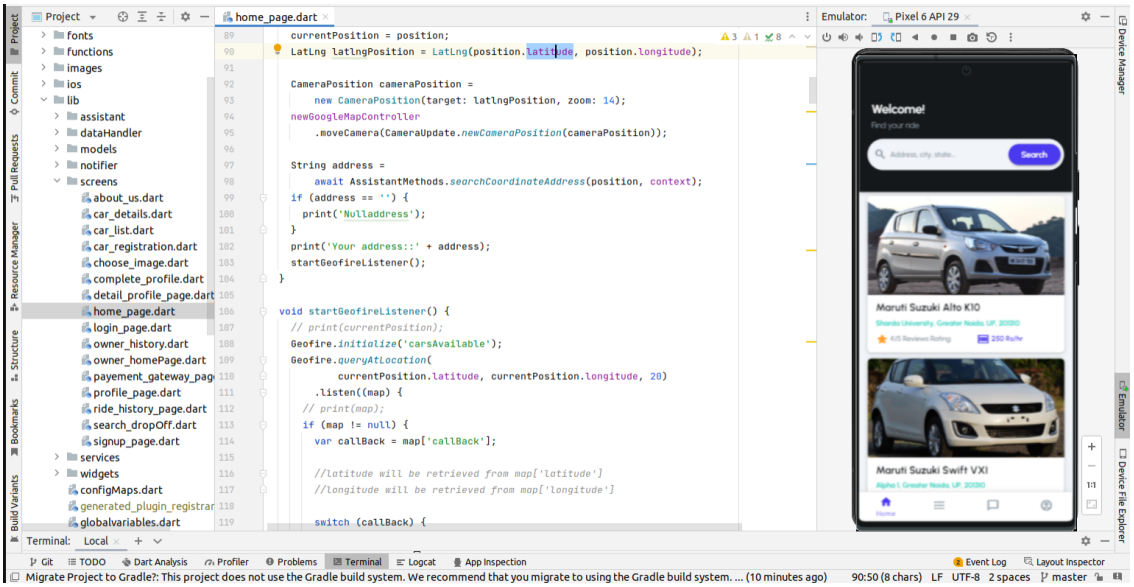


Figure 4.1.3.3 Home Screen of the Application

Figure 4.1.3.3 illustrates the home screen of the application. In this user interface, there are vehicles listed for rent. Along with that, it consists of search bar where the user can search of desired vehicle. The vehicle cards display some of the details of the vehicle.

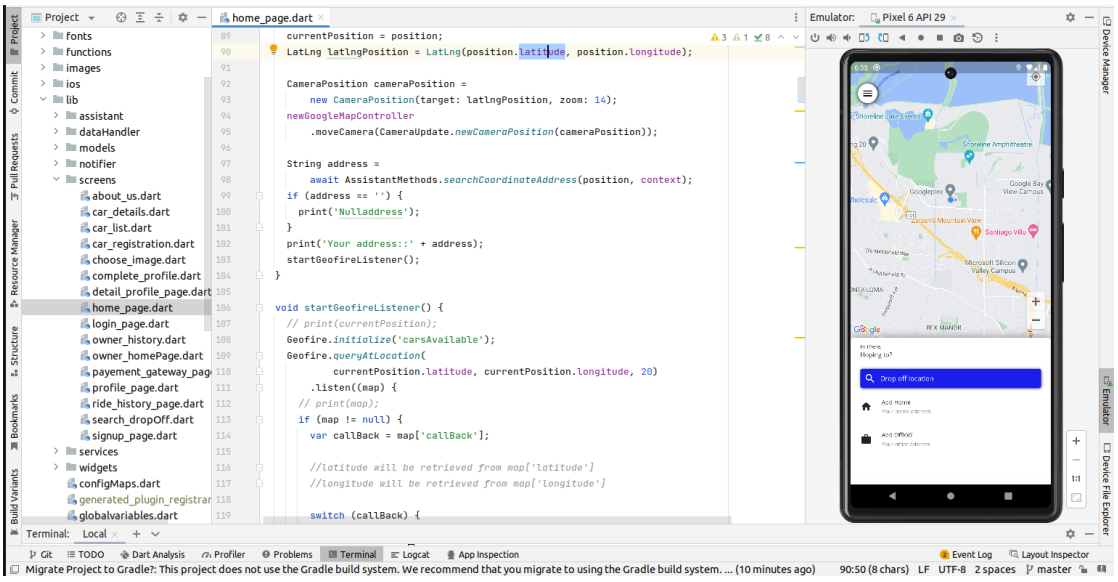


Figure 4.1.3.4 Google Maps Screen of the Application

Figure 4.1.3.4 Illustrates the Google map service used in the application which is used to find the nearby vehicles for rent purpose. In addition it can be used to reach the location of the vehicle.

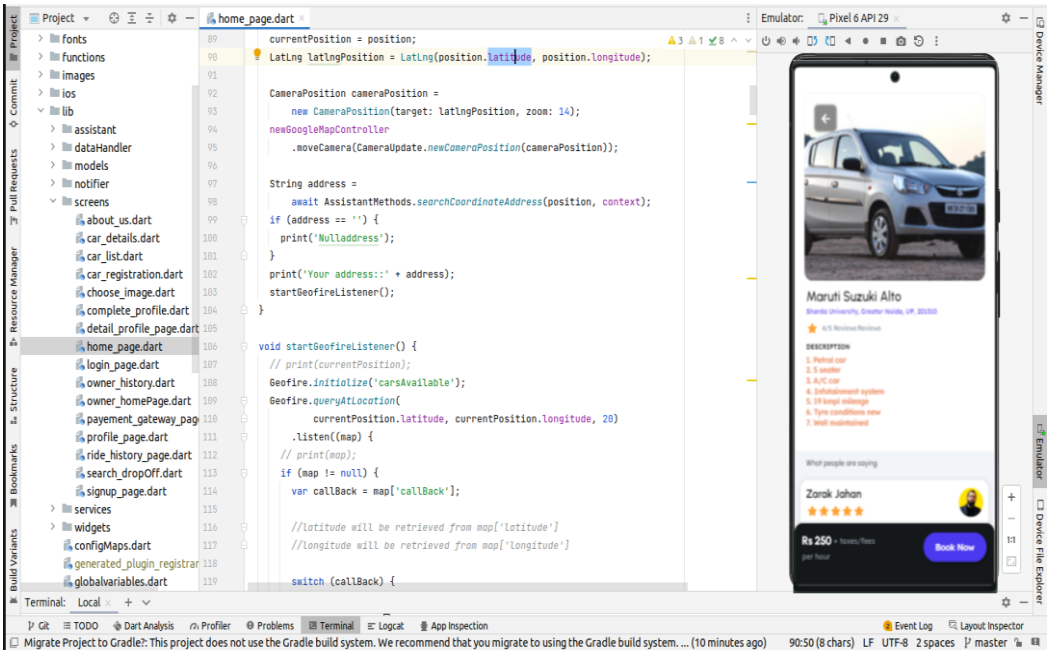


Figure 4.1.3.5 Car Details Expanded Screen of the Application

On clicking the vehicle card as shown in Figure 4.1.3.3 the user is greeted with the details of the vehicle as shown in Figure 4.1.3.5. It illustrates all the details of the vehicle along with the ratings of users and a tab to book the vehicle, which also shows the prices.

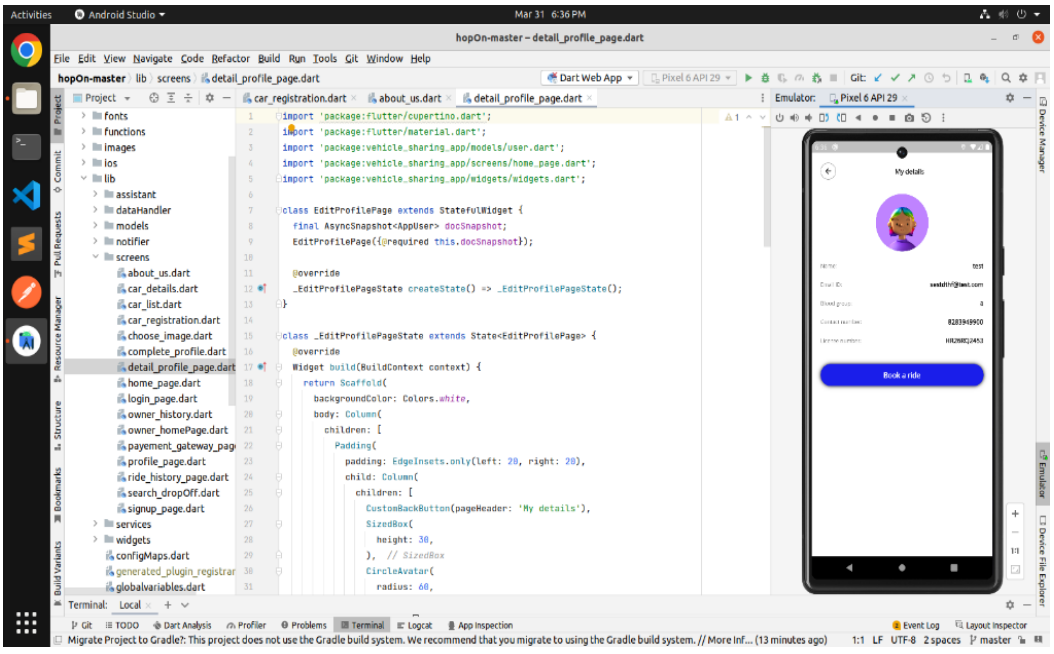


Figure 4.1.3.6 Renter Details Screen of the Application

Figure 4.1.3.6 illustrates the renter details which he/she submits while completing the profile. It consists of information like name, email, and license number.

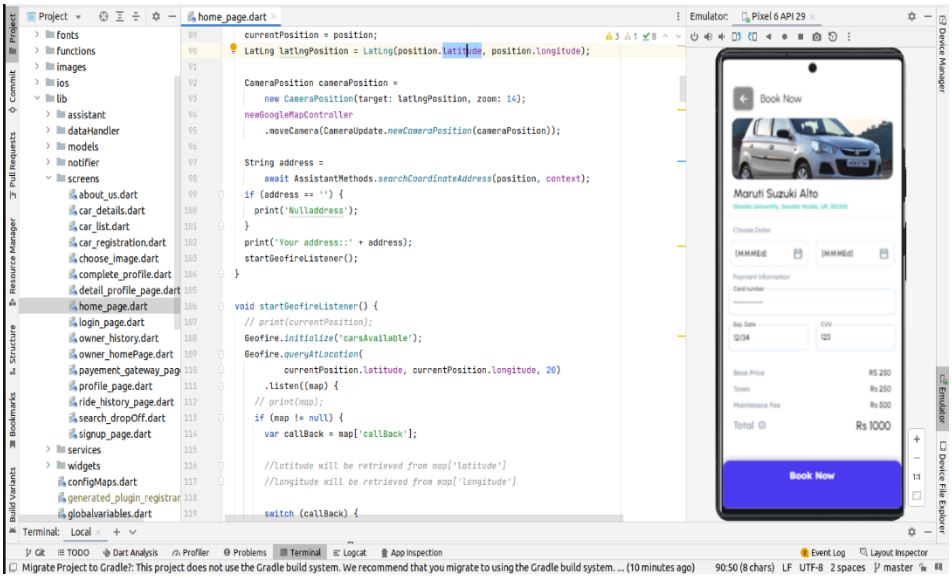


Figure 4.1.3.7 Booking Details Screen of the Application

On clicking book a ride the user is greeted with the screen as shown in Figure 4.1.3.7 In this screen, the used to add details like booking dates, and credit card details and is shown the base price of the vehicle (illustrative not accurate). On adding details and clicking on book now the user prompted to wait as shown in Figure 4.1.3.8.

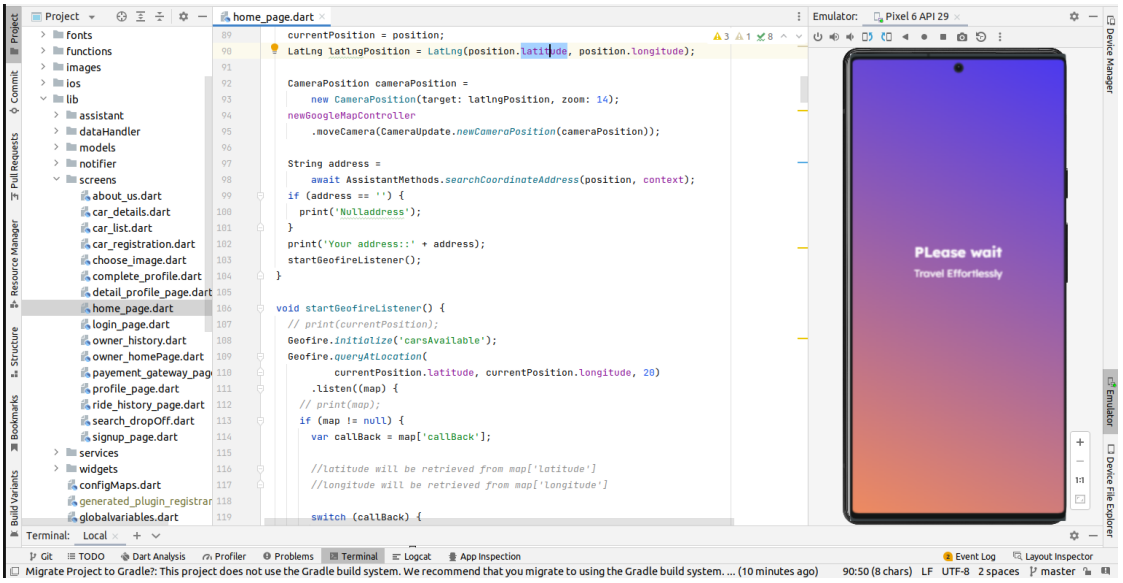


Figure 4.1.3.8 Waiting Prompt Screen of the Application

After 100 ms the vehicle is booked and the user is navigated to the following screen as shown in Figure 4.1.3.9. It illustrates the booked vehicles of the user.

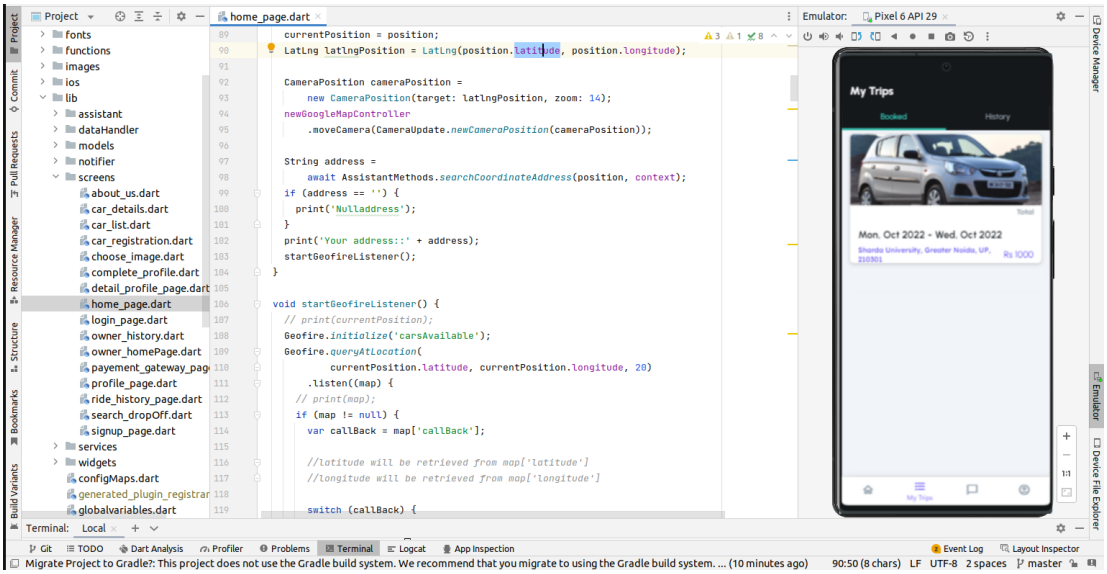


Figure 4.1.3.9 Booked Vehicles Screen of the Application

This screen also shows the option for the history of booking as shown in Figure 4.1.3.10. On this screen, the user can see the completed trips. The vehicle cards show the details of booking dates, the amount for the trip and an option to rate the trip.

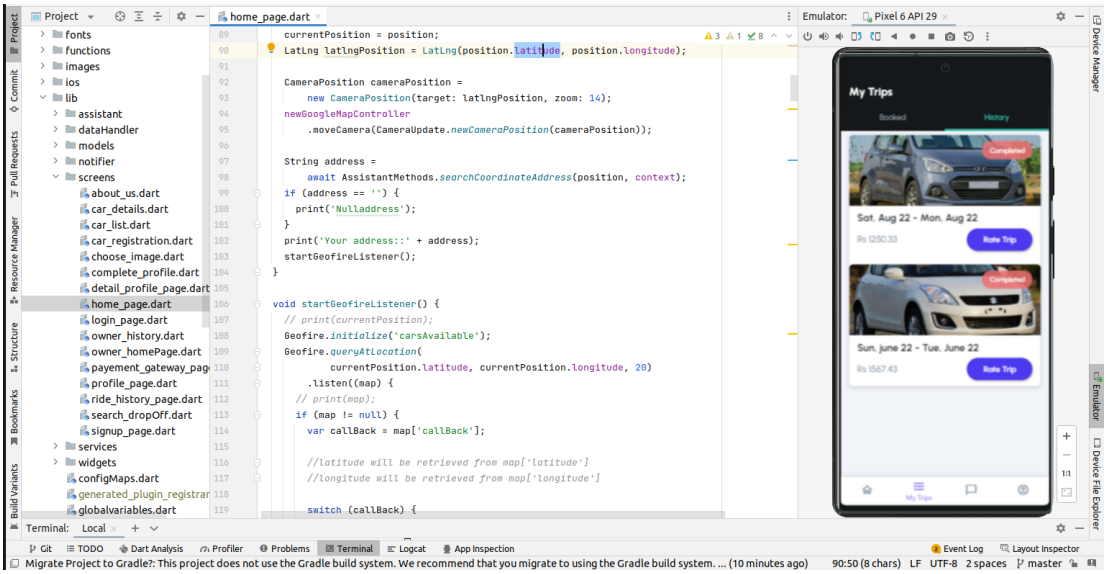


Figure 4.1.3.10 History of Bookings Screen of the Application

On clicking on the vehicle card the user can expand the details of the trip as shown in Figure 4.1.3.11. On this screen, the user can see all the details of the trip like the date of booking, destination booking, price breakdown (illustrative not accurate), an option to rate the trip and a bottom bar to chat with the host.

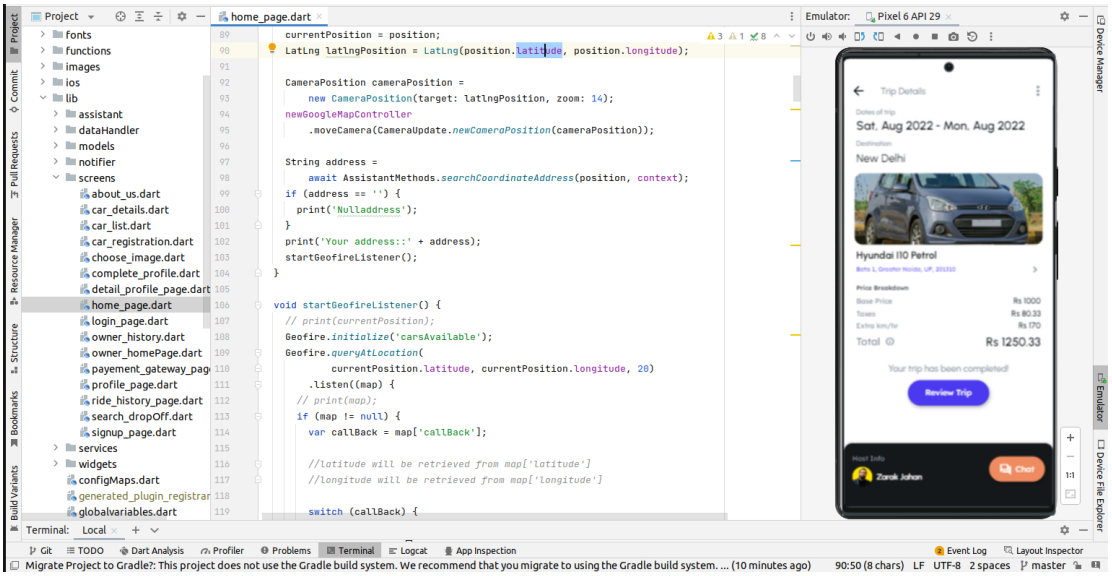


Figure 4.1.3.11 Expanded History of Bookings Screen of the Application

The chat option can be accessed on the navigation bar as shown in Figure 4.1.3.12. On this screen, the user can manage the chats.

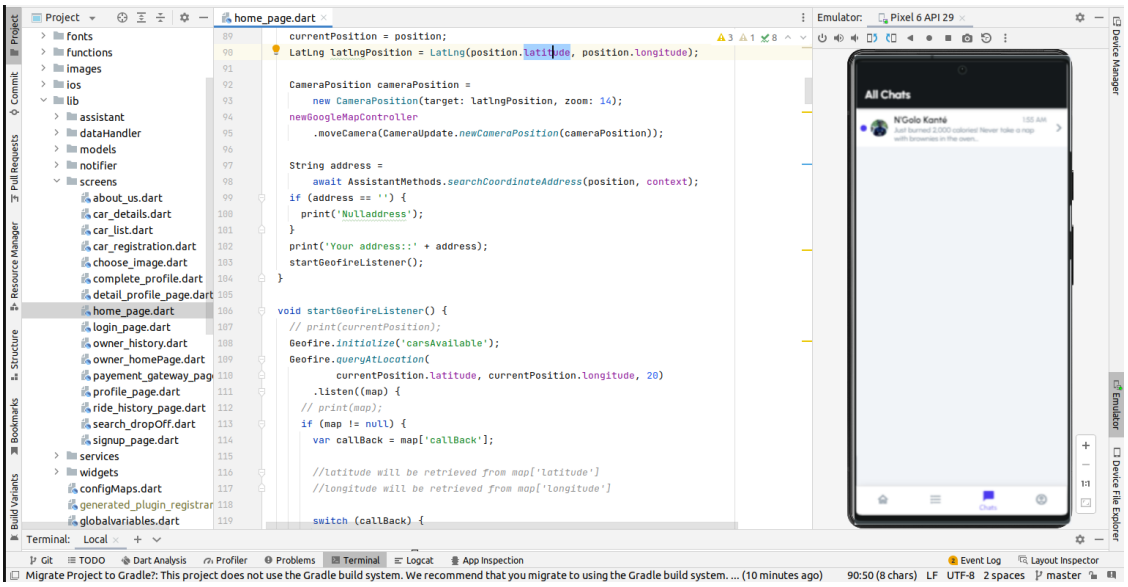


Figure 4.1.3.12 Chat Screen of the Application

Figure 4.1.3.13 illustrates the profile section of the user where he/she can edit profile, logout and most importantly the user can opt to become a host by clicking on the “Become a Host” button.

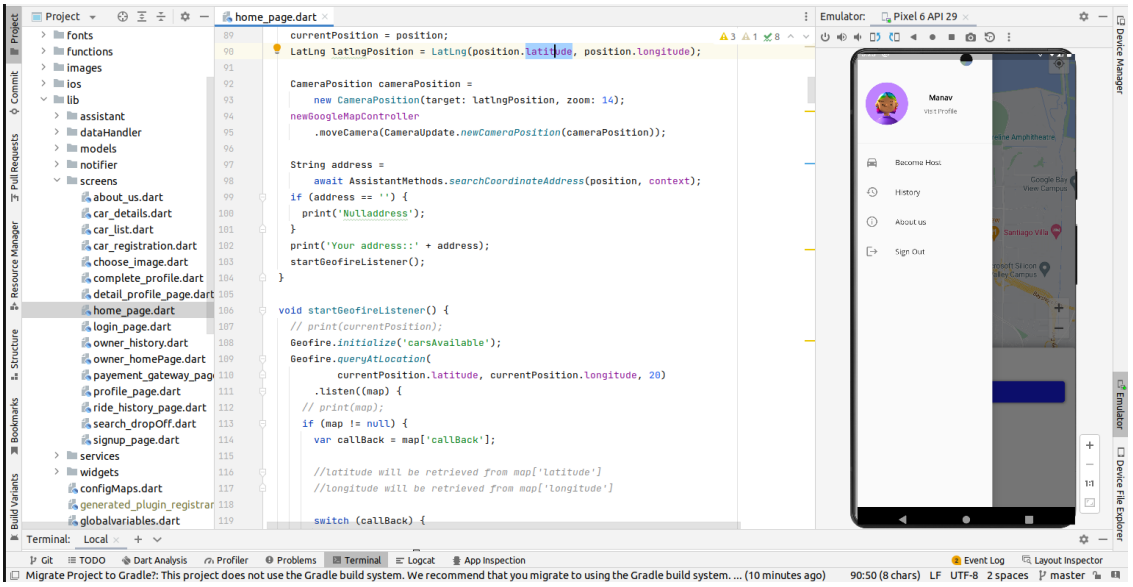


Figure 4.1.3.13 Side Bar Screen of the Application

The user is greeted with a screen where he/she needs to upload the details of their vehicle to publish their vehicle as shown in Figure 4.1.3.14.

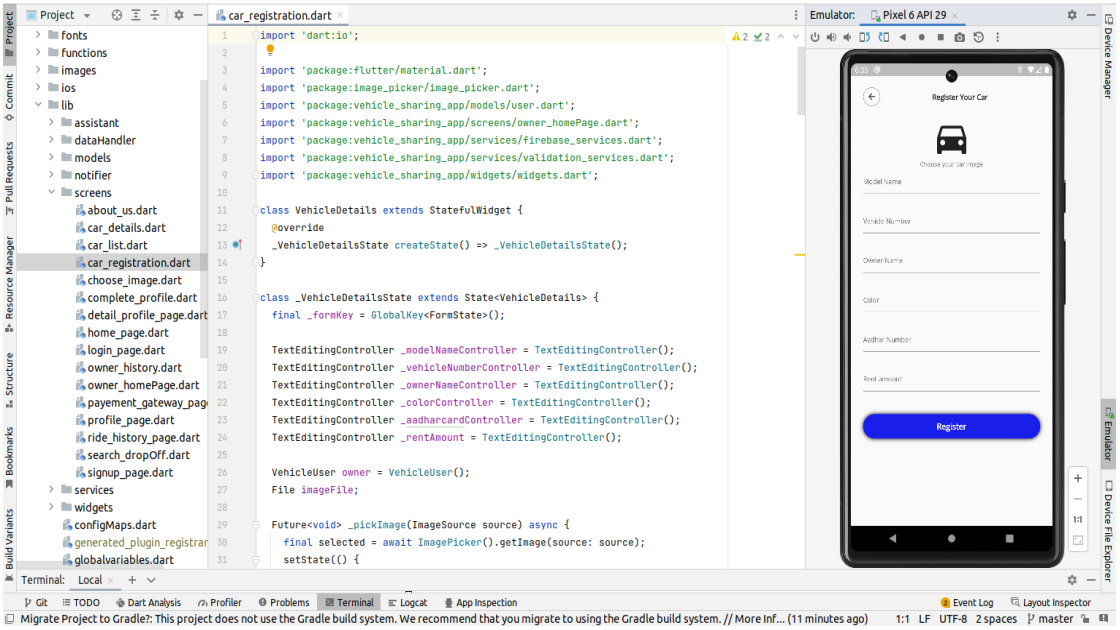


Figure 4.1.3.14 Vehicle Registration Details Screen of the Application

4.2 Software testing process

Software testing [36] is the process of examining a software programme object in order to determine any discrepancies between the input and the anticipated result as well as to evaluate the item's quality. An approach that must be applied at some point throughout

the development cycle is software testing. In other terms, software testing is a method for validation and verification.

Verification is the process used to make sure the product complies with the specifications laid forth at the start of the development phase. To put it another way, we want to make sure the product does what we want it to.

Validation is the procedure used to make sure the product meets the relevant requirements once the development phase is complete. To make certain that the product is made in a way that is consistent with what the consumer wants.

The two fundamental types of software testing are black-box testing and white-box testing.

Black box testing ignores the system's internal environment and only focuses on the output generated. Structural testing, commonly referred to as white box testing, focuses on how the system is set up inside and is always utilized for validation.

The few test cases that were run to look for faults and problems are shown in Tables 4.2.1-4.2.4.

Table 4.2.1. Test case for Executing the code

Test Case Number	**01**
Overview of the Test Case	Open VS Code and run the main code file.
Operation	Compiling and debugging the code. Run the programme.
Projected outcome	The programme ought to launch effectively.
Final result	The programme launches successfully.
Status	Test Case Pass

Table 4.2.2. Test Case for Registration of the User

Test Case Number	**02**
Overview of the Test Case	The registration form should load and accept entries.

Operation	Put in your email and password
Projected outcome	Login should be possible for the user.
Final result	The user logs in successfully.
Status	Test Case Pass

Table 4.2.3. Test Case for Booking a Vehicle

Test Case Number	**03**
Overview of the Test Case	Vehicle reservation.
Operation	Choose the preferred vehicle. Booking.
Projected outcome	The user ought to be able to reserve the vehicle.
Final result	The booking module gave the desired response.
Status	Test Case Pass.

Table 4.2.4. Test Case for Posting a vehicle

Test Case Number	**04**
Overview of the Test Case	Posting of vehicles.
Operation	Submit the necessary information and files.
Projected outcome	It should be possible for the Host to post their car.
Final result	On the homepage, the car is marked as available for hire.
Status	Test Case Pass.

4.3 Summary

In this chapter, we describe the modules of the application that are developed to reach the objective. The screenshot of the application is shown and description of each screen is given to explain the working and features of the screen. This part also includes test cases designed to look for errors and the algorithms used in the programme.

CONCLUSION

5.1 Conclusion

This application offers a platform for a cutting-edge transportation system that enables users to both hire cars and list their own cars for rent in order to generate additional cash from idle automobiles. In order to identify the market gap and open the door for the development of this project, the related work done in this area is described in this paper. Together with a description of the project's structural analysis, the technology used there is also discussed. The objectives mentioned in section 1.3 of Chapter 1 are accomplished as follows:

- We successfully published a conference paper “Peer-to-Peer Self-Driving Car Rental: A Case Study on the Development and Limitations of a Novel Transport System” and a journal paper “SERIGO: Development and Implementation of a Peer-to-Peer Self-Driving Car Rental App Using Flutter Framework”.
- The conference paper accomplishes the objective of doing the feasibility study and proposing a hybrid solution for the P2P rental.
- The journal paper accomplishes the objective of developing the flutter based application for the proposed hybrid solution.

5.2 Future Scope

Our plans and initiatives for the future include using and testing it in real-world settings. In addition, we want to create an iOS version of the programme. There will also be consideration on security and privacy problems. In this study, we tried to underline the significance of developing a dynamic and creative car-sharing platform. We have described the use cases, functionality, and system architecture of our peer-to-peer self-driving car rental application. This programme targets those who own cars but don't use them very often by allowing users to post their vehicles for rent on the website to make a little more money. Similar to this, we planned to include a number of features, such as a real-time tagging system to determine whether the car leaves the approved area and a capability to schedule future vehicle maintenance with any authorized service provider.

5.3 Limitations

The limitations faced in the development of the project are as followed:

- Insufficient feasibility study.
- Low-powered equipments were available.
- Unable to incorporate few APIs as they were paid.
- Couldn’t incorporate some features due to lack of advance programming skills.
- Testing was done manually.

REFERENCES

[1]. B. G. Rajagopal, “Intelligent traffic analysis system for Indian road conditions,” Int. J. Inf. Technol., vol. 14, no. 4, pp. 1733–1745, 2022.

[2]. Alharbi, G. Halikias, M. Yamin, and A. A. Abi Sen, “Web-based framework for smart parking system,” Int. J. Inf. Technol., vol. 13, no. 4, pp. 1495–1502, 2021.

[3]. M. C. Suryadev Singh Rathore, Ed., Analysis of Self Drive Rental Cars Industry in India, vol. 8, no. 4. International Journal of Management & Business Studies, 2018.

[4]. L.-H. Wu, “Understanding collaborative consumption business model: Case of car sharing systems,” DEStech trans. mater. sci. eng., no. mmme, 2017.

[5]. P. J. L. St, Ed., “Life in a Sharing Economy: *What AirBnB, Turo,” and Other Accommodation-Sharing Services Mean for Cities*, vol. 350, 2020.

[6]. E. Ferguson, “The rise and fall of the American carpool: 19701990,’’ Transportation,” vol. 24, pp. 349–376, 1997.

[7]. Ghorpade, A. Joshi, S. Mali, P. Agrawal, and H. Agrawal, “Pool: A peer-to-peer ride sharing app,” International Journal of Engineering Applied Sciences and Technology, vol. 5, no. 12, 2021

[8]. Gedam, M. Sahare, R. Sachdeo, and N. Kulkarni, “Smart transportation based car pooling system,” E3S Web Conf., vol. 170, p. 03004, 2020.

[9]. K. Liu, J. Zhang, and Q. Yang, “Bus Pooling: A Large-Scale Bus Ridesharing Service,” IEEE Access, vol. 7, pp. 74248–74262, 2019.

[10]. P. Ehsani and J. Y. Yu, “The merits of sharing a ride,” in 2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2017.

[11]. P. K. Binu and V. S. Viswaraj, “Android-based application for efficient carpooling with user tracking facility,” in 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2016.

[12]. X. Duan, C. Jin, X. Wang, A. Zhou, and K. Yue, “Real-Time Personalized Taxi-Sharing,” in Database Systems for Advanced Applications, Cham: Springer International Publishing, 2016, pp. 451–465.

[13]. R. Hasan, A. H. Bhatti, M. S. Hayat, H. M. Gebreyohannes, S. I. Ali, and A. J. Syed, “Smart peer carpooling system,” in 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), 2016.

[14]. B. Shen, Y. Huang, and Y. Zhao, “Dynamic ridesharing,” SIGSPATIAL Spec., vol. 7, no. 3, pp. 3–10, 2016.

[15]. B. Cao, L. Alarabi, M. F. Mokbel, and A. Basalamah, “SHAREK: A scalable dynamic ride-sharing system,” in 2015 16th IEEE International Conference on Mobile Data Management, 2015.

[16]. V. Armant and K. N. Brown, “Minimizing the driving distance in ride sharing systems,” in 2014 IEEE 26th International Conference on Tools with Artificial Intelligence, 2014.

[17]. D. Dimitrijevic, V. Dimitrieski, and N. Nedic, Real-time carpooling and ride-sharing: Position paper on design concepts, distribution and cloud computing strategies Computer Science. 2013.

[18]. “Apache Cordova,” Apache.org. [Online]. Available: https://cordova.apache.org. [Accessed: 27-Dec-2022].

[19]. S. Ma, Y. Zheng, and O. Wolfson, “T-share: A large-scale dynamic taxi ridesharing service,” in 2013 IEEE 29th International Conference on Data Engineering (ICDE), 2013.

[20]. X. Xing, T. Warden, T. Nicolai, and O. Herzog, “SMIZE: A spontaneous ride-sharing system for individual urban transit,” in Multiagent System Technologies, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 165–176.

[21]. Tashildar, N. Shah, R. Gala, T. Giri, and P. Chavhan, “Application development using flutter,” International Research Journal of Modernization in Engineering Technology and Science, vol. 2, no. 8, pp. 1262–1266, 2020.

[22]. G. Idan Arb and K. Al-Majdi, “A Freights Status management system based on Dart and Flutter programming language,” J. Phys. Conf. Ser., vol. 1530, no. 1, p. 012020, 2020.

[23]. “Flutter architectural overview,” Flutter.dev. [Online]. Available: https://flutter.dev/docs/resources/technical-overview. [Accessed: 07-Feb-2023].

[24]. R. K. Megalingam, S. Vishnu, S. Sekhar, V. Sasikumar, S. Sreekumar, and T. R. Nair, “Design and implementation of an android application for smart shopping,” in 2019 International Conference on Communication and Signal Processing (ICCSP), 2019.

[25]. L. Moroney, “An Introduction to Firebase,” in The Definitive Guide to Firebase, Berkeley, CA: Apress, 2017, pp. 1–24.

[26]. S. Khedkar and S. Thube, “Real Time Databases for Applications,” Real Time Databases for Applications” International Research Journal of Engineering and Technology (IRJET), 2017.

[27]. N. Smyth, Firebase Essentials - Android Edition. North Charleston, SC: Createspace Independent Publishing Platform, 2017.

[28]. Kumar, Mastering Firebase for Android Development. Packt Publishing Ltd, 2018.

[29]. K. G. Sudiartha, I. N. E. Indrayana, I. W. Suasnawa, S. A. Asri, and P. W. Sunu, “Data structure comparison between MySql relational database and firebase database NoSql on mobile based tourist tracking application,” J. Phys. Conf. Ser., vol. 1569, p. 032092, 2020.

[30]. R. S. Pressman, Software Engineering A Practitioner’s Approach 7th Ed. Palgrave macmillan, 2009.

[31]. M. J. Chonoles, OCUP 2 Certification Guide: Preparing for the OMG Certified UML 2.5 Professional 2 Foundation Exam. Morgan Kaufmann, 2017.

[32]. R. Fauzan et al., “A different approach on automated use case diagram semantic assessment,” Int. J. Intell. Eng. Syst., vol. 14, no. 1, pp. 496–505, 2021.

[33]. B. Dobing and J. Parsons, “How UML is used,” Commun. ACM, vol. 49, no. 5, pp. 109–113, 2006.

[34]. Z. Liu, X. Hu, D. Zhou, L. Li, X. Zhang, and Y. Xiang, “Code Generation From Flowcharts with Texts: A Benchmark Dataset and An Approach,” in Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022, pp. 6069–6077.

[35]. S. Sendall and W. Kozaczynski, “Model transformation: the heart and soul of model-driven software development,” IEEE Softw., vol. 20, no. 5, pp. 42–45, 2003.

[36]. J. Gaur, A. Goyal, T. Choudhury, and S. Sabitha, “A walkthrough of software testing techniques,” in 2016 International Conference System Modeling & Advancement in Research Trends (SMART), 2016.

ANNEXURE-I

I1. 7th Semester Outcome

Conference: Presented

Z.Jahan, M.Chauhan, N.Parween, M.Chhabra, "Peer-to-Peer Self-Driving Car Rental: A Case Study on the Development and Limitations of a Novel Transport System" 17th INDIACom; INDIACom-2023; IEEE Conference ID: 57626 2023 10th International Conference on Computing for Sustainable Global Development, [IEEE-SCOPUS]

Proceedings of the 17th INDIACom; INDIACom-2023; IEEE Conference ID: 57626
2023 10th International Conference on "Computing for Sustainable Global Development", 15th - 17th March, 2023
Bharati Vidyapeeth's Institute of Computer Applications and Management (BVICAM), New Delhi (INDIA)

Peer-to-Peer Self-Driving Car Rental: A Case Study on the Development and Limitations of a Novel Transport System

Zarak Jahan, Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, Greater Noida, India, zjahan025@gmail.com
Manav Chauhan, Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, Greater Noida, India, manavchauhan277@gmail.com
Nazia Parween, Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, Greater Noida, India, naziaparween633@gmail.com
Megha Chhabra, Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, Greater Noida, India, megha.chhabra@gmail.com

Abstract—Peer-to-peer (P2P) car-sharing scheme has been increasing recently through online platforms. Peer-to-peer car-sharing schemes can be utilized to put extensively available but still untapped resources of unutilized vehicles to use. The objective of this paper is to observe the impact of a peer-to-peer self-driving car rental system on the economy, its standard in the market and analyze the attitude of the people towards this scheme through a survey and give insights into the application being developed to address the gap analyzed in the peer to peer self-driving car sharing market. A survey was conducted with 141 participants, and the results allowed us to determine how much participants knew about the business model, how emotionally attached they were to their personal vehicles, and how much money would attract them towards this idea. Based on the collaborative consumption business concept, we also suggested an online platform to tackle the gap.

Keywords—Peer to Peer (P2P), Collaborative consumption, Economy, willingness, Business Model.

I. Introduction

In the past few years, peer-to-peer consumption has become an important socio-economic business model and has experienced drastic growth worldwide in virtual and networking technologies. One of the best examples of the successful implementation of the P2P business model is the car-sharing business. In this P2P business model, consumers can rent a vehicle owned by a company or private vehicle owners in exchange for monetary compensation as per the market rate. India as the fastest growing economy is facing the challenge of increasing traffic [1] which leads to worst congestion, wastage of fuel, pollution, less parking spaces [2] which not only frustrates the daily commuters but also causes a loss of 22 billion dollars annually based on wastage of fuel, loss of working hours, pollution as well as health hazards.

The concept of sharing economy was commercialized for underutilized goods and services for monetary compensation. Peer-to-peer business models have recently developed in India with a prominent focus on car sharing with the introduction of many car-sharing rental services like Uber, Ola, Zoom cars etc., but the idea of P2P car sharing is still an obscure idea of Indians due to lack of knowledge, trust and the sense of car being a very personal property which cannot be shared.

In this study, we researched based on previous studies how Peer to peer self-driving car rental business models affect the economy, what is the standard in the market and how willing are people to join this scheme. We also conducted a survey to know how people feel about this idea and categorized them as per age, education level, gender, no. of family members in the household, annual income and no. of vehicles owned. This study found that people are mostly not willing to opt for the idea of renting out their personal vehicle to people that are not known to them due to a lack of trust and knowledge of the P2P car-sharing system. Many participants also expressed concerns regarding the health of their vehicles and security concerns but those willing to join the collaboration consumption model tend to find a monetary benefit of around 5k to 10k per month decent for them to rent out their vehicles.

II. Literature Survey

The research on the P2P car rental system in India is very limited as this concept has not been introduced in the Indian market. This literature review relates to research or studies conducted in India as well as outside India where P2P self-driving car rental systems or other self-driving car rental systems made a big impact.

A) Willingness to rent the personal vehicle

Natalia et al. [3] identified that giving personal vehicles on rent in exchange for money was becoming popular in the United States. They conducted a survey between February 2018 and April 2018 to know about the willingness, perception, attitude and decision process people go through while offering their vehicle for P2P renting. The respondents of the survey were asked how willing are they to rent their car. The data collected from the survey was used to determine a pattern and was found that some variables were significant statistically in determining the willingness of renting a personal vehicle like age, livelihood, gender, the composition of the household, ownership of the vehicle, distance from the grocery store and use of other share base services. While estimating the results it was found that variables related to health and body mass index were insignificant in determining the model but they were formerly found to be statistically important in bike sharing and ride-sourcing behaviours concluding that lifestyle choices could affect rental willingness but the model results rejected this [4][5]. It was also found that females were less likely to rent their personal vehicles as compared to males as also found by [6]. It was also found that people 40 years of age and above showed heterogeneous behaviour but on average they were less likely to rent their personal vehicles than other age groups. Many researchers have already identified old age as the key factor in identifying the endorsement of shared economy infrastructure [7].

Paolo et al. [8] carried out a survey in the city of Milan to determine the various factors like behaviour and attitude of the people towards the peer-to-peer carsharing idea. In Italy, the Ministry of The Environment in the major cities advocated ICS (Inciviva. Car sharing) among which Milan was the most successful one in carsharing experiences. They conducted a survey in which the males and females in the age group of 18-60 were considered and characterized based on their working status and education level. They also collected data related to the size of the household, the number of transport vehicles in the family, the nature of the vehicles owned, the most frequently used transportation means, the average distance travelled by each person by their vehicle, nature of trips like shopping, visit relatives). A total of 1211 responses were collected among which only 25 declared no vehicle owned by their family. The majority of the responses provide positive feedback towards the peer-to-peer carsharing business model however [9] suggests that most people focus on sharing between well-established and trusted community members in order to beat the fear of sharing. It was observed that people would like to share their personal vehicle with a small set of people known to them and on the other hand, one more reason for the negative response was that people consider their car the most valuable object (36.3%) and expect it to be readily available (47.9%). Respondents were also asked how much money they desire to join the platform and a threshold figure of the desired price was obtained using the bisection method which predicted a threshold starting from 50 euros per month. People also tend to lend their vehicles more in the early morning than late in the afternoon.

The main reason for the reluctance to this concept was unwarranted car wear and tear. Ahuja et al. [10] presented a computational framework for assessing vehicle health according to decipherable machine-learning methods. For experimental assessment, they employed a vehicle evaluation dataset and examined it using a variety of black box models, augmented with model interpretability methodologies, which produced encouraging outcomes.

B) Self-drive car rental industry

Sarvalon and Malek [11] conducted research in which they identified and analyzed the current scenario of the self-drive car rental industry in India. Their study was based on three objectives. The first objective of understanding the market of self-drive car rental systems in India was found to still be a rudimentary idea as India is a developing country still facing issues of traffic congestion and limited spaces for parking. This idea attracts the younger generation from 25-35 years of age defying the traditional desire of owning a car. Since 2013 various startups have emerged in India to provide and satisfy the demands of this increasing trend like Zoomcar, Ola Rule, Revv, and Uber hire. However big and experienced international companies are considering India as a potential market and will evolve in the coming years. The second objective was to understand the working of the Industry its policies, what are eligibility criteria, fee structure, agreements, and security policies. Apart from this, the functioning of the system and the technology used like GPS, ETA, SOS button, Notifications, payment integration, Driver Behavior Monitoring (DBM) and On-Board Diagnostics (OBD) were studied. The last and third objective was to know the customer preference about which service provider they like the most, to do so 300 people participated in the survey conducted in the city of Madhya Pradesh. The survey resulted in that people preferred Zoomcar the most as compared to Ola Rentals, Myles etc.

Brenda and Christian [12] conducted research on various car-sharing business models, drivers and barriers, the behaviour of customers and the balancing of vehicles. The study enlightened the gravity of the association between stakeholders in the automotive sector for the determination of advancing their customer base and boosting applications and assets for managers. For the policy builders, their influence in aiding carsharing take-off is advertised with attention on equal support to various transport services to promote collective achievement. The methodology administered three stages: planning, conducting and reporting the research to keep up with the current awareness of carsharing, and creating a systematic review to integrate data and viewpoints while keeping the bias minimum [13]. Peer-to-peer carsharing business models are put under extensive term synergistic consumption which can be elaborated as using online marketplaces and social media to

Figure I1.1. Conference paper

I2. Certificate of Presentation

The conference paper was presented by Zarak Jahan (2019004410) on 16th March 2023 in online mode. Figure I2.1 shows the certificate of presentation.

Figure I2.1. Certificate of Presentation

I3. 8th Semester Outcome

Journal: Published

Zarak Jahan, Manav Chauhan, Nazia Parween, and Megha Chhabra. SERIGO: Development and Implementation of a Peer-to-Peer Self-Driving Car Rental App using Flutter Framework [J]. Int J Performability Eng, 2023, 19(3): 155-166.

doi: 10.23940/ijpe.23.03.p1.155166

URL: http://www.ijpe-online.com/EN/10.23940/ijpe.23.03.p1.155166

I4. Journal paper Published

The journal paper accepted on 23rd March 2023 and published on the website.

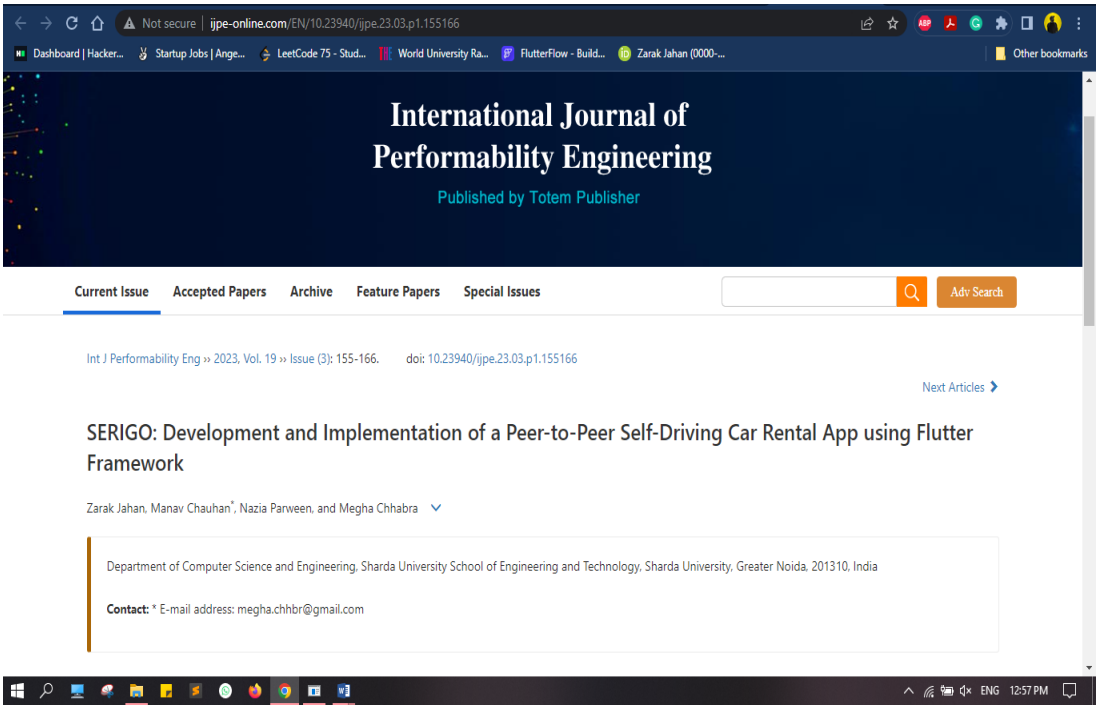
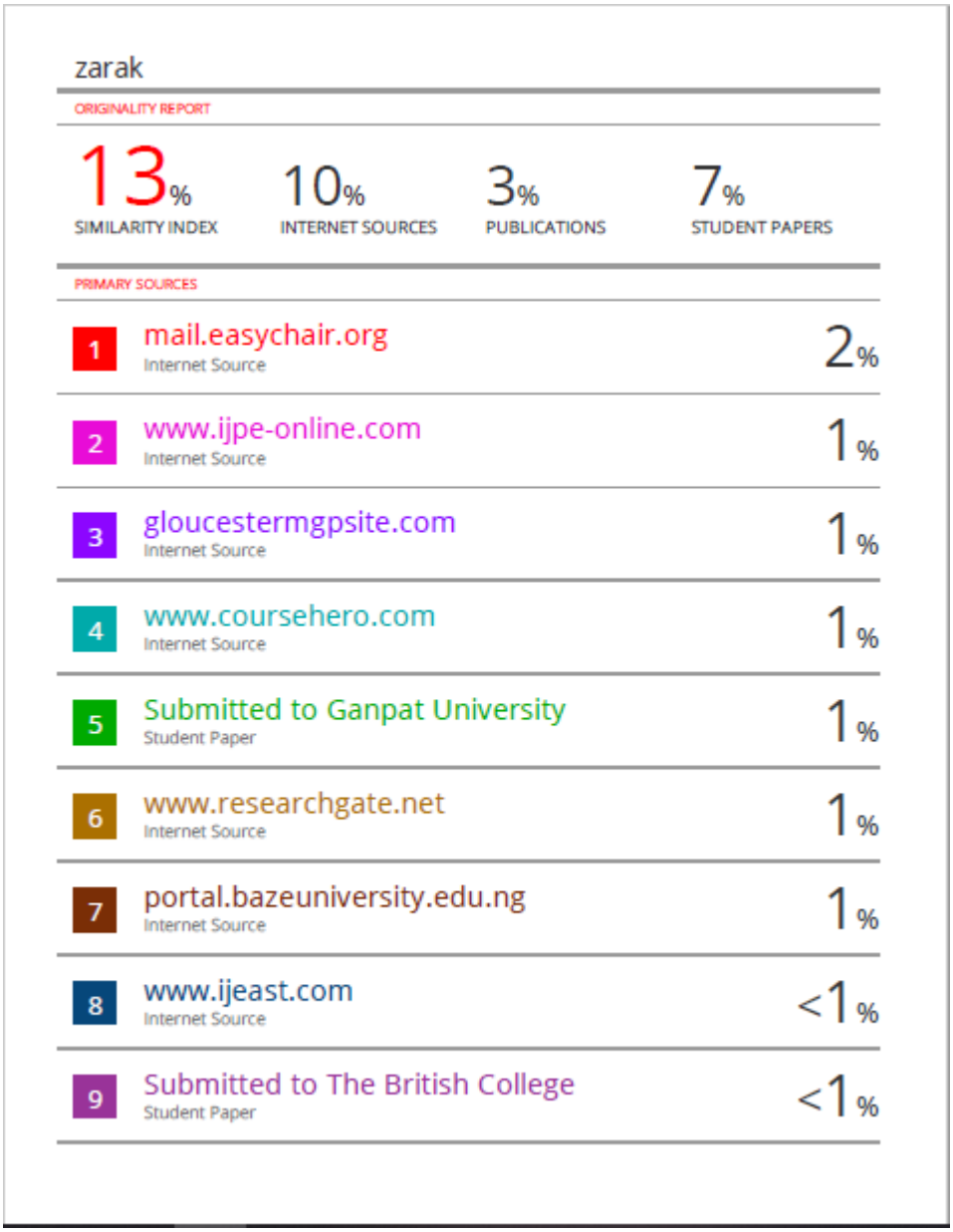


Figure I4. IJPE Published Screenshot

ANNEXURE-II

II2. Plagiarism report



zarak

ORIGINALITY REPORT

13%	10%	3%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	mail.easychair.org Internet Source	2%
2	www.ijpe-online.com Internet Source	1%
3	gloucestermgpsite.com Internet Source	1%
4	www.coursehero.com Internet Source	1%
5	Submitted to Ganpat University Student Paper	1%
6	www.researchgate.net Internet Source	1%
7	portal.bazeuniversity.edu.ng Internet Source	1%
8	www.ijeast.com Internet Source	<1%
9	Submitted to The British College Student Paper	<1%

Figure II2. 1 Plagiarism Report