A

Project Report

on

CARBUDDY: RIDE-SHARING PLATFORM.

Submitted in Partial Fulfillment of
the Requirements for the Degree
of

Bachelor of Engineering

in

Computer Engineering

to

Kavayitri Bahinabai Chaudhari
North Maharashtra University, Jalgaon

Submitted by

Sharayu Mahendra Banait
Priyanka Vijaysing Pardeshi
Chetan Jayram Mahajan
Mahendra Rajendra Bagul

Under the Guidance of

Dr. Manoj E. Patil

DEPARTMENT OF COMPUTER ENGINEERING
SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY,
BAMBHORI, JALGAON - 425 001 (MS)
2022 - 2023

SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY, BAMBHORI, JALGAON - 425 001 (MS)

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the project entitled *CarBuddy: Ride-sharing platform.*, submitted by

Sharayu Mahendra Banait
Priyanka Vijaysing Pardeshi
Chetan Jayram Mahajan
Mahendra Rajendra Bagul

in partial fulfillment of the degree of *Bachelor of Engineering* in *Computer Engineering* has been satisfactorily carried out under my guidance as per the requirement of Kavayitri Bahinabai Chaudhari North Maharashtra University, Jalgaon.

Date: March 30, 2023
Place: Jalgaon

Dr. Manoj E. Patil
Guide

Dr. Manoj E. Patil
Head

Prof. Dr. G.K.Patnaik
Principal

Acknowledgements

We would like to express deep gratitude and sincere thanks to all who helped to complete the project successfully. Also thanks to Prof. Dr. G.K. Patnaik, Principal, for providing the facilities to complete the project work. A deep gratitude goes to Prof. Dr. Manoj E. Patil, Head of the Department, for granting us opportunity to conduct project work and giving us a valuable guidance timely being our project guide. Sincere thanks to Mr. Akash D. Waghmare, Asst. Prof., Project Incharge and great thanks to our friends, project associates and all those who helped directly or indirectly for completion of the project. Last but not least thanks to our parents for supporting us.

Sharayu Mahendra Banait
Priyanka Vijaysing Pardeshi
Chetan Jayram Mahajan
Mahendra Rajendra Bagul

Contents

List of Tables

List of Figures

Abstract

Carpooling allows drivers to share rides with other passengers. This helps in reducing the passenger's fares and time, as well as traffic congestion and increases the income for drivers and car owners. In recent years, several carpooling systems have been proposed. However, all the existing systems are works well in metro cities but they are not available in local areas. Our system majorly focuses on local cities. Our system brings people who are traveling to the same destination, in a single car can decrease the number of vehicles on the road and thereby reduce pollution to a large extent. The system makes use of a pre-registration method, by which only identified people get into the carpool vehicles so that crime can be reduced. The Central Monitoring system used here intelligently handles carpool requests and switches them efficiently. The user can acquire details of his/her travel by sending an SMS from their mobile phone. Waiting centers are allocated at each part of the city and the users can enter the carpooled vehicles only from these waiting centers. also, this system has basic features like sharing of the journey, user rating, car location tracking, girl's safety measures, etc. Governments encourage carpooling to increase high-occupancy vehicle lanes rather than commuters. Carpooling is seen as a more environmentally friendly and sustainable way to travel as sharing journeys reduces carbon emissions, traffic congestion on the roads, and the need for parking spaces. Also, we will use cloud vision API for video checking of cars and google API to check the location of car owners and passengers.

Chapter 1

Introduction

Carpooling is an environment-friendly method where sharing of rides can reduce the number of vehicles on the road which in turn reduces the problems like environmental pollution, traffic congestion and lack of space for parking area. By having more people in a single vehicle, reduces the various costs of the journey. For example, combining three or more people who set off on the same direction in different vehicles, into one, reduces the costs of fuel needed for separate vehicles. Cost reduction by sharing and company of fellow travelers also reduces the stress of the driver. Most of the existing applications related to carpooling like BlaBlaCar, Quick rider, Pool My Car etc. provides sharing of the journey, user rating, feedback, payment, etc. But these applications are only available in metro cities they are not available for local areas.

The organization of this Chapter is as follows. Section 1.1 describes background of the project.Motivation is represented in Section 1.2.Section1.3 represents Problem definition of the project.Scope of the project is described in Section 1.4. Section 1.5 describes objective of the project.Selection of life cycle model is mentioned in section 1.6 and organization report is in section 1.7 Finally,the Summary of given chapter is described in last Section 1.8.

1.1 Background

Carpooling is defined as a non-profit common journey where drivers offer vacancies in their car while passengers share the cost of the trip with the driver. A successful carpooling route requires identifying the route and the time and the location of departure and arrival between passengers. Coordination is facilitated by carpooling platforms which are offered free of charge when operated by local government agencies or by paying an amount for use.

The benefits from carpooling are important both from the participants' perspective, who share the trip costs, and from the city's perspective, where the traffic congestion, parking demand, and gas emissions are reduced. However, the integration of carpooling services in the context of smart cities appears limited. its aim is to analyze the literature on the

knowledge area resulting from the intersection of smart carpooling value generation.

1.2 Motivation

The constant population and economical growth has caused an enormous increase in the number of private cars in cities worldwide. This phenomenon has lead to traffic congestions, parking problems, inordinate fuel consumption, and excessive pollution. While the average capacity in a car is 4 passengers, cars are often observed with one rider. In fact, most of the people drive alone to work. Because existing public transportation systems cannot be adapted in a timely manner or without major capital investments to address the growing needs of populations, developing social solutions, such as carpooling, where a driver and one or more passengers having totally or partly common routes share a private vehicle would be a green as well as a cost effective public solution to the daunting problem of traffic congestion. Carpooling stands out as an effective and social approach to exploit available transportation resources, i.e., fill the empty seats in private vehicles. It allows people to share a ride for similar departure and destination locations.

1.3 Problem Definition

The following gives an insight onto the problem statement:

Problem Statement Title: Development of system for carpooling and ride sharing for local cities.

Organization: Smart Cities

Description: CarBuddy is a carpooling and ride-sharing platform where the car owner can share their car with other passengers instead of going by car alone. It offers two different types of services. this rideshare web app offers a consumer to find a ride for a particular location. On the other hand, consumers are also allowed to offer a ride to the people.Here first the user has to enter their starting point, destination, as well as the date they wish to travel on. CarBuddy then presents you with all the available carpooling options for that particular search.

Complexity: Complicated

1.4 Scope

The proposed system is based on Visual Studio code which is a free and open-source source-code editor for developing websites. it can be used with a variety of programming languages.VS code offers more features that enhance your productivity when building web apps. This web app is developed using Python for backend we have used Django and SQLite.

Carpooling system is very effective means to reduce pollution and the congestion of vehicles in cities. It also provides an eco-friendly way to travel. It also provides an opportunity to meet new people. As today most people prefer private vehicle to travel due to delay caused in public transport system and luxuries provided by private vehicles. Pre-registration ensures security, as only identified people get into the vehicle so that trust can be established. The people registered can be allotted specific days on which they should take their private vehicle, so that no inconvenience is caused to its registered passengers for daily commute. Thus, the proposed carpooling system will be effective in reducing environment pollution.

It is a communication platform between car owners and passengers. Car owners will be able to post a notice announcing that he has been traveling between some particular locations regularly or just once, to search a travel-mate in order to reduce the ride costs.

The database that bests suits to our app is SQLite. SQLite is an open-source SQL database that stores data to a text file on a device.SQLite supports all the relational database features.

1.5 Objective

Expected outcomes from completing the modules include the following points:

1. To Save money by sharing the cost of driving one car.
2. To Promote alternative modes of transport.
3. TO Provide social connections in society.

1.6 Identification of software development Process Model

The four basic process activities of specification, development, validation and evolution are organized differently in different development processes. In the waterfall model, they are organized in sequence, while in incremental development they are interleaved. Software Development Life Cycle(SDLC) is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce a high quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

1.6.1 Waterfall Model - Design

After assessing various Software Development Life Cycle Models, the one that best suited to our application is Iterative Waterfall Model. After considering the users of our app and their requirements, our team's technical capacity and ability, associated risks, etc. referring to the final evaluation we adopted Iterative Waterfall Model. Using this approach, various phases in Life Cycle can be conducted in several cycles.

This Model has Five Phases: Requirements analysis and specification, design, implementation, unit testing, integration and system testing, and operation and maintenance. The phases always occur in this order and do not overlap. The developer completes each phase before the next phase begins. With every cycle of phases, a usable product is released, with each release providing additional feature. Using this model, requirements can be specified as many as possible and accordingly developers can implement those requirements. Overall, the software product produced isn't a prototype, it's a high quality code that is expanded into the final product.

Following is the diagram for Iterative Waterfall model:

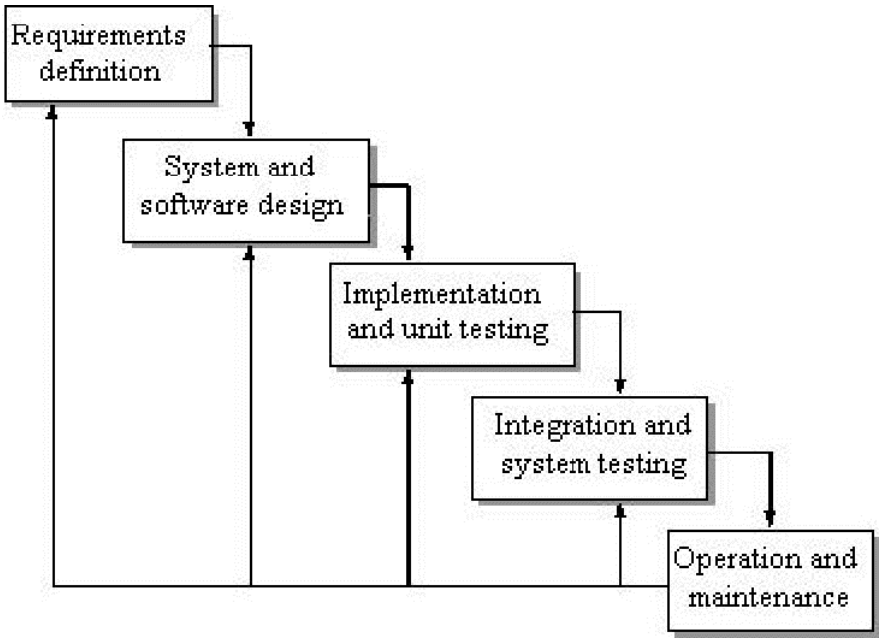

Figure 1.1: Waterfall model

- **Requirement Analysis:** All possible requirements of the system to be developed are captured in the phase and documented in a requirement specification document.
- **System Design:** The requirement specifications from first phase are studied in the phase and the system design is prepared. The system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

- **Implementation:** With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures. Deployment: Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.
- **Maintenance:** There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment. All these phases are cascaded to each other in which progress is seen as flowing steadily downwards(like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name Waterfall Model. In the model, phases do not overlap.The project under consideration is being developed using the above discussed model[5].

1.7 Organization of Report

- **Chapter 1:** entitled as Introduction describes the details about Background, Problem Definition, Scope and Objective of the project, Identification of Software Development Process Model and Organization of report.

- **Chapter 2:** entitled as Project Planning and Management consists of details about the Feasibility Study, Risk Analysis, Project Scheduling, Effort Allocation and Cost Estimation of the project.
- **Chapter 3:** entitled as Analysis describes in detail, the Requirement Collection and Identification, H/w and S/w Requirements, Functional and Non-Functional Requirements and a Software Requirements Specification(SRS).
- **Chapter 4:** includes design about System Architecture, Data Flow Diagram and various UML Diagrams.
- **Chapter 5:** consists of coding and implementation phases of the application.
- **Chapter 6:** briefs about various types of testing phases against which the application is tested.

- **Chapter 7:** describes the results or the outputs of application and also discussion about advantages as well as disadvantages of proposed system.
- **Chapter 8:** provides the conclusion and future work which can be done on the topic.

1.8 Summary

In the chapter all the details about problem definition, motivation, scope, objectives of the project and selection of software development model are described. In the next chapter, the Project Planning and Management is presented.

Chapter 2

Project Planning and Management

Scheduling in project management is the listing of activities, deliverables, and milestones within a project. A schedule also usually includes the planned start and finish date, duration, and resources assigned to each activity. Effective project scheduling is a critical component of successful time management.

This chapter is organized as follows. Section 2.1 describes Feasibility Study of the project. Risk Analysis of the project selection is represented in Section 2.2. Section 2.3 represents Project Scheduling of the project. Effort Allocation of the project is described in Section 2.4. Section 2.5 describes Cost Estimation of the project. Finally, the Summary is described in last Section 2.6

2.1 Feasibility Study

As the name implies, a feasibility analysis is used to determine the viability of an idea, such as ensuring a project is legally and technically feasible as well as economically justifiable. It tells us whether a project is worth the investment—in some cases, a project may not be doable. There can be many reasons for this, including requiring too many resources, which not only prevents those resources from performing other tasks but also may cost more than an organization would earn back by taking on a project that isn't profitable. There are few types of feasibility are exist, So developers should take care of this feasibility:

1. **Technical Feasibility**
2. **Economical Feasibility**
3. **Operational Feasibility**

2.1.1 Technical Feasibility

This assessment is based on an outline design of system requirements, to determine whether the company has the technical expertise to handle completion of the project. Technical feasibility involves the evaluation of the hardware, software, and other technical requirements of the proposed system. This portal is developed using development technologies such as Django, MySQL, Html, CSS, Bootstrap, JS. All the required hardware and software are easily available in the market. Hence the portal is technically feasible. There is no more hardware required other than the system.

2.1.2 Economical Feasibility

In the system, the organization is most satisfied by economic feasibility. Because, if the organization implements the system, it need not require any additional hardware resources as well as it will be saving lot of time. The proposed system that is educational forum is web platform. When the project will be a complete software functionality, people will have to pay lesser money, though it will be a proprietary one, still the money paid in comparison to the yields will be negligible. Therefore, the project will prove to be economically feasible.

2.1.3 Operational Feasibility

Operational feasibility is the ability to utilize, support and perform the necessary tasks of a system or program. It includes everyone who creates, operates or uses the system. Provide summary statistical information without disclosing individual's confidential data. This makes the system operationally feasible. The portal we will be developing will be based on web development technologies and the database provided by the organization. though technical skills are required for proper implementation and synchronization of freely available tools.

2.2 Risk Analysis

Risk analysis and management are a series of steps that help a software team to understand and manage uncertainty. Many problems can plague a software project. A risk is a potential problem might happen, it might not. But, regardless of the outcome, it is really a good idea to identify it, assess its probability of occurrence, estimate its impact, and establish a contingency plan should the problem actually occur.

2.3 Project Scheduling

Scheduling in project management is the listing of activities, deliverables, and milestones within a project. A schedule also usually includes the planned start and finish date, duration, and resources assigned to each activity. Effective project scheduling is a critical component of successful time management.

Task	Start	Days to Complete
Selection of project title	**06 sep 22**	**5**
Gathering the information	**11 sep 22**	**6**
Analyse the information	**17 sep 22**	**5**
Discuss with guide	**22 sep 22**	**3**
Made the problem statement	**26 sep 22**	**2**
Checked the Scope	**28 sep 22**	**4**
Checked Feasibility	**01 oct 22**	**4**
Design UI	**05 oct 22**	**6**
Design UML Diagram	**11 oct 22**	**12**
Design Frontend	**23 oct 22**	**10**
Adding Backend	**1 Jan 23**	**6**
Database Connection	**7 Jan 2023**	**3**
Project Completed	**10 Jan 2023**	**2**

Table 2.1: Project Scheduling

2.4 Effort Allocation

Project scheduling means teamwork. Project is developed by a combination of the effort of the team, So the whole project is divided into modules and number of modules is allotted to team members. After completion of each module, it will be link from one module to another module to form a complete project.

Task	Sharayu	Priyanka	Chetan	Mahendra
Identification of project and Gathering the information	✓	✓	✓	✓
Study of existing system	✓	✓	✓	✓
Learn skills that require for app development	✓	✓	✓	✓
Selection of software life cycle model	✓			✓
Identifying functional and non-functional requirement	✓		✓	✓
Design UML, UX and UI diagrams	✓	✓	✓	✓
Design Frontend	✓	✓	✓	✓
Adding Backend				✓
Database Connection	✓			

Table 2.2: Effort Allocation

2.5 Cost Estimation

For any software project, it is necessary to know the total cost require for development of project and development time requires. Cost Estimation is needed before development is initiated.

The cost of any software project is calculated by the formula:-

$$C = aLb \tag{2.1}$$

Where,

$$C = costofproject \tag{2.2}$$

$$a = 1.4(constant) \tag{2.3}$$

$$b = 0.93(constant) \tag{2.4}$$

$$L = sizeofcode \tag{2.5}$$

2.6 Summary

In this chapter, project planning and management is described in detail.In the next chapter Analysis is presented.

Chapter 3

Analysis

Project analysis can be used to estimate the economic or engineering viability of road investment projects by performing life cycle analysis of pavement performance, maintenance and/or improvement effects together with estimates of road user costs.

The organization of this Chapter is as follows. Section 3.1 describes requirements collection and identification for the project.Software Requirementś Specification (SRS) is represented in Section 3.2. having subsection 3.2.1 Product features,**??** operating environment, 3.2.3 assumptions,3.2.4Functional and 3.2.5 non-functional requirements. and finally,3.2.6 External Interfaces (User, Hardware, Software, Communication were described.Section 3.3 describes whole summary of the chapter analysis.

3.1 Requirements Collection and Identification

Requirement Analysis is the phase in which the entire requirement related to software are collected which helps the designer to design a actual system. Requirement Analysis enables the system engineer to specify software function and performance indicates software interface with other system elements and establish constraints that software must meet. Requirement Analysis provides the software designer with models that can be translated into data, architectural, interface, and the user with means to assess quality once project is built.

The app should show the destinations and routes available, so passengers will be able to book a destination and select a route. They will arrange pickup times through the app, which will also display the other passengers who are going to be sharing the journey with them.The make, size and model of each car should be displayed on the app.Users should be able to view the drivers profile before they choose to join the trip.After they select a driver a user will get a notification through their account to notify them that the driver is heading to their pickup destination.Drivers could register as a carpool driver online and post their car details such as license plates, make, model and size of car.After the customer views the driver and views the other people sharing the trip or journey, the app will show the time the

driver will arrive.Drivers will need to be able to view the passengers who are requesting to share the journey, and the driver will have the option to approve the passengers.Every time a passenger requests a trip or journey from a driver, the passenger's profile should be displayed to the driver.While they drive to the destination, a web mapping service like Google Maps will be open, to prevent drivers from getting lost.Users will be able to pay for the journeys through their accounts on the app.At the end of the journey the app should provide the passenger with an opportunity to rate the driver.If the driver receives a good review this will help the driver attract more passengers.The app could also display how much fuel was used throughout each carpool journey.This will show the drivers how much fuel they saved while carpooling.To make CarPoolMe user friendly, there should be instructions displayed on the app for users to create an account.A website will provide additional information and allow users to login and view and edit their details.Security will need to be implemented to prevent the app from getting hacked.There should be a login installed in the app for both passengers and drivers.

Every time a passenger requests a trip or journey from a driver, the passenger's profile should be displayed to the driver.While they drive to the destination, a web mapping service like Google Maps will be open, to prevent drivers from getting lost.Users will be able to pay for the journeys through their accounts on the app.At the end of the journey the app should provide the passenger with an opportunity to rate the driver.If the driver receives a good review this will help the driver attract more passengers.A website will provide additional information and allow users to login and view and edit their details.Security will need to be implemented to prevent the app from getting hacked.There should be a login installed in the app for both passengers and drivers.

3.2 Software Requirementś Specification (SRS)

A software requirements specification(SRS) is a document that captures complete description about how the system is expected to perform. It is usually signed of at the end of requirements engineering phase. Software Requirement Specification (SRS) is the starting point of the software development activity. It is a complete description of the behavior of a system which is to be developed. The SRS document enlists all necessary requirements for project development. The derive the requirements we need to have clear and through understanding of the product which is to be developed. This is prepared after detailed communication with project teams and the customer. A SRS is a comprehensive description of the intended purpose and environment fir software under development. The SRS fully describes what the software will do you it will be expected to perform. An SRS minimizes the time and required by developers to achieve desired goals and also minimizes the development cost. A

good defines how an applications will interact system hardware, other programs and human users in a wide variety of real-world situations.

Characteristics of SRS:

1. Correct: As SRS is correct if, and only if, every requirement stated therein is one that the software shall meet. Traceability makes this procedure easier and less prone to error.

2. Unambiguous: An SRS is unambiguous if, and only if, every requirements stated therein has only one interpretation. As a minimum this requires that each characteristics of the final product be described using a single unique term.

3. Verifiable: It is verifiable if there exists some finite cost-effective process with which a person or machine whether software product meets requirements

4. Consistent: Consistency refers to internal consistency. If an SRS does not agree with some higher level document such as a system requirements specification then it is not correct. An SRS is internally consistent if and only if, no subset of individual requirements described in it conflict.

5. Modifiable: SRS is said to be modifiable if its structure and style are such that any changes to the facilities the referencing of each requirement in future enhancement.

3.2.1 Product Features

Features for proposed product are mentioned as follows:

For Consumers :

- *Finding a Ride With Ease*

The moment a user sign-ups, the riders should be able to enter the pick-up and drop-off locations on the search box provided. This would, in turn, improve the ride search time and its ETA (Estimated time of arrival), irrespective of the location of other riders. The goal should be to offer timely results while ensuring the user-friendliness of the process.

- *Information about other passengers*

By maintaining transparency on the car sharing app, you would be able to gain your users' trust as it will offer them a sense of security. So, it should be made a priority to keep the passenger details visible among all the travelers. When you let them know about their co-passengers, they would know that you care.

- *The Split Payment System*

For it is a shared ride we are concerned about, the fare should also be equally distributed. The payment should depend upon the individual distance traveled in order to avoid any additional expense for a rider. This not only helps in making commuting affordable but also adds to the user experience big time.

For Drivers :

- *Ability to Accept/Reject Ride Request*

A driver should always be given the feel of being included. If a rider is important for the business, so is the driver. Take it as a responsibility to offer them an option to accept or reject rides as per their convenience. This will give them a reason to stick with your carpooling services.

- *Every Rider's Trip Information*

When there are multiple riders traveling together, drivers are likely to get confused. This is why; every rider's trip information should be clearly communicated over the driver's interface. It will further help them reach every rider within the estimated time while synchronizing the pick-ups and drop-offs with ease.

- *Easy Access to the Driver Dashboards*

Every driver should be having a well-maintained profile in your database. The moment a driver visits their respective dashboard, every essential detail should pop-up. Be it the driver's overall rating, rides completed, earnings or deduction details; the interface should maintain a see-through structure.

3.2.2 Operating Environment

Hardware Requirements of our project is as follows.

1. 8GB RAM
2. Upto 1TB HDD
3. Intel Core i5 or above / Ryzen 5 2400g
4. Internet Connectivity

Software Requirements of our project is as follows.

1. Operating System: Windows 10 or higher
2. visual studio code
3. MySQL
4. HTML

3.2.3 Assumptions

1. Reduces the number of cars on the road.
2. Provide social connections in society.
3. Taking care of girls' safety and privacy issues.
4. Car drivers insurance policy.
5. Reducing overall traffic congestion on the roads.

3.2.4 Functional Requirement

Few of its functional requirements are as follows:

General application requirements

- *Login*

Since all the operations that can be done using the application requires both the driver and passenger to be logged in, they can use the login forms of either Google Plus or Facebook. For this matter, the user is prompted to connect the app to his account and then proceed for sign in/up. After the user authorizes the application to access his social media account, the server retrieves his info. If he has never logged to the application before, a new account is created for him.

- *Modify profile information*

All users can modify their profile information. The profile information contain: name, phone number, email, type/color of car if any. The user can easily edit these information in order to be contacted and recognized.

- *Social media sharing*

In order to attract more users to the application and help users find passengers, users should be able to share their activity on the application on social media. A suggestion for sharing trips' creation, trips' registration or check in should pop-up whenever those previous actions are performed. The sharing should be authorized by the users and not done automatically by the application in order not to spam the users' account and gain the users' confidence.

- *Rate driver/passenger*

Both the driver and passenger can rate each other in other to gain reputation. The importance of the rating is to encourage users to be helpful and nice during the trip so that they gain popularity in the application. It is also a way to ensure users of who can be trusted or not. The ratings represent a relative guarantee for the users to trust each other.

Regular trips

- *Create new regular trip*

The driver can create a new trip to be displayed when passengers search for trips. The application will prompt the driver or information of the regular trip which consists of destination, origin, meeting point (which can be pointed in a map), departure time/date , estimated arrival time and traveling preferences (number of free spots, price, size of bags, smoking/non-smoking, pets, stops ...). After providing this information, the user publishes it in order to find passengers. Upon the creation of the trip, a user can share the trip he just created in social media to find passengers to drive with.

- *Search for regular trips and reservation*

When a passenger needs to find a driver for a destination, he can use a search form which asks for destination, origin, departure date/time. He can also specify the traveling preferences. When he finds a suitable trip, he can reserve a spot easily by tapping a button which will send a notification to the driver telling him that a passenger has reserved.

- *Check-in trip*

Whenever the driver or passenger arrive to the meeting point at the time agreed upon, he can check-in the meeting point in order to notify the other user and to show his punctuality. The application will use the devices GPS in order to make sure that the users are in the meeting point. When somebody checks in, a notification is sent to all the carpoolers saying

that somebody is in the meeting point.

Frequent trips

- *Add frequent trip*

The driver can create a frequent trip where they show the origin and destination, departure and return times in addition to the frequency (daily and weekly).

- *Search frequent trips*

A passenger can search for a frequent that he can join. The passenger should specify the departing neighborhood, destination, departure times and frequency. The application will try to match it with the best trip. If the passenger is satisfied, he can register to the frequent and will be given the contact of the other members.

3.2.5 Non-Functional Requirement

Non-Functional Requirements are the characteristics or attributes of the system that are necessary for the smooth operation of the system. Those requirements are listed below.

- *Performance*

The application has to offer a very quick response time as the meeting between the driver and passengers is done through notifications. In other words, the server should be able to treat notifications and propagate them instantly. The application should handle 1000 users sending queries at the same time.

- *Scalability*

The application should respond properly to a high increase of users. It should be able to handle from 10 000 users to 100 000 users. And also from 100 000 to one millions users.

- *Extensibility*

The application should be extensible in order to support multiple platforms including iOS, Windows Phone and Web.

- *Availability*

Since a lot of information about the trips and check in are available in the application, it has to be highly available and guarantees a good server up-time. The server should allow only 1 hour down time per year which is 99.99 percent up-time.

- *Privacy and Security*

The application should ensure the privacy of the users including the trips they take part in, their social media accounts and their accounts. The login system should also be robust where only authorized users can post and edit their own information.

- *Maintainability*

Since the application may be developed in the future by adding other features, it should be easily maintainable.

3.2.6 External Interfaces (User, Hardware, Software, Communication)

- *User Interface*

1. Login: System will check credentials against User and will create Login ID for each user.
2. Home Page: The homepage is design with colorful interface and intuitive layout for easy navigation so that anyone can easily navigate through the website.
3. Driver Dashboard: Driver has access to accept or reject the ride.
4. Consumer Dashboard: Consumer and request ride and make payment.
5. Chat: This Page will include Chat Forum facility which will allow driver and consumer to communicate easily with better performance.

- *Hardware Interface*

It is web based application. The hardware on which it resides will be any computer can have internet. The hardware requirement includes a system with following configurations:
The hardware requirement includes a system with following configurations:

1. Processor: Intel i3 or above
2. RAM: 1 GB

3. Input device: Standard Keyboard and Mouse

4. Output device : High Resolution Monitor

- *Software Interface*

This project is web-based application so only browser with internet connection is required from user standpoint. This product will utilize various software components for its web-based functionality. Web server require to host website from developers' standpoint.

1. Frontend: HTML, CSS, JS
2. Backend: Django
3. Database: MySQL

- *Communication Interface*

As a part of its core functionalities this product will require HTTP OR HTTPS communication interface with client device. It will also require to communicate with SQL DATABASE.

3.3 Summary

In this chapter , Analysis is described. In the next chapter, Design is presented.

Chapter 4

Design

System Design chapter provides graphical structure of the project by using various UML diagrams. System design provides the understanding and procedural details necessary for implementing the system recommended in the system study. Design is a meaningful engineering representation of something that is to be built. It can be traced to a customer's requirements and at the same time assessed for quality against a set of predefined criteria for good design. In the software engineering context, design focuses on four major areas of concern are data, architecture, interfaces and components.

Section 4.1 describes the System Architecture which includes. Description about the Data Flow Diagram is provided in Section 4.2. Section 4.3 describes various UML diagrams related to the architecture. Section 4.4 provides a short summary.

4.1 System Architecture

A System Architecture is the conceptual model that defines the structure, behaviour, and more views of a system and architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and Assistant architecture can comprise system components, the extremely visible properties of those components, the relationship i.e the behaviour between them. It can provide a platform in which system can be procured, and systems developed, that will work together to implement the overall system.

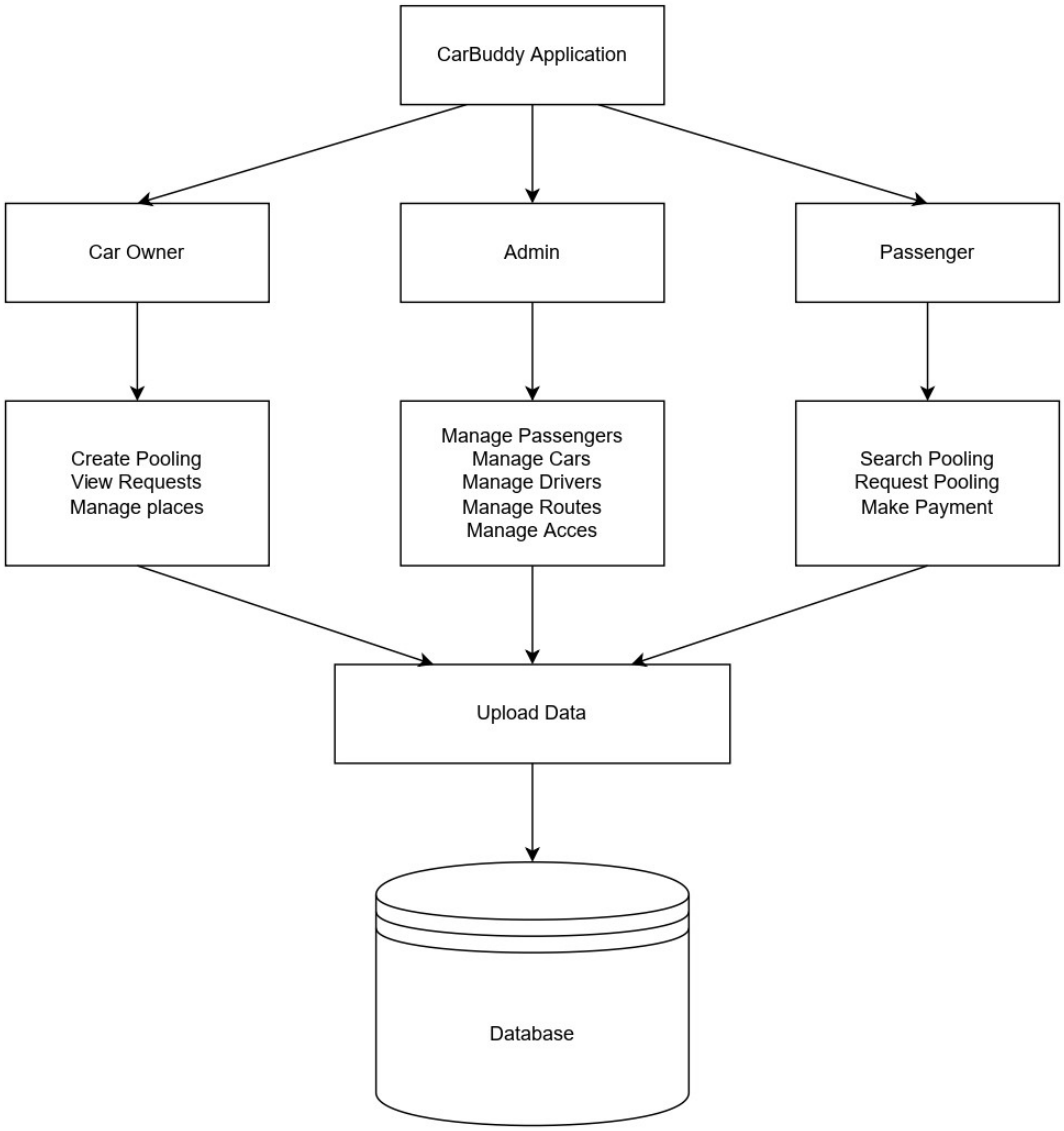

Figure 4.1: System Architecture

4.2 Data Flow Diagram

A data flow diagram is a flowchart can help to visualize the data pipeline of a system user can trace happens to the data as it moves between components. It is a great to find redundancies and optimize the speed and responsiveness of software. A DFD is often used as a preliminary step to create an overview of the system going into great detail, it can later be elaborated. DFDs are used for the visualization of data processing (structured design). A DFD show kind of information input to and output from the system, the data will advance through the system, and it the data will be stored. It represented information of process timing processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses

on control flow, or a UML activity workflow diagram, which presents both control and data, flows as a unified model.

A data flow diagram can dive into progressively more detail by using levels and layers,zeroing in on a particular piece. DFD levels are numbered 0 or 1, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

1. **DFD Level 0:**

   DFD Level 0 is also called a Context Diagram. It is a basic overview of the whole system or process being analyzed or modeled. It is designed to be an at a glance view, showing the system as a single high-level process, with its relationship to external entities. It easily understood a wide audience, including stakeholders, business analysts, data analysts and developers. In the Figure 4.2 the data flow diagram level0 is described.

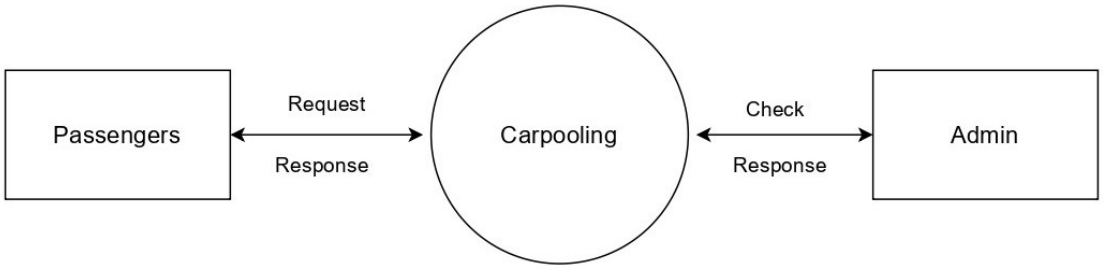

Figure 4.2: DFD Level 0

2. **DFD Level 1:**

   DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. It highlight the main functions carried out by the system, it break down the high-level process of the Context Diagram into its sub processes. In the Figure 4.3 the data flow diagram level 1 is described.

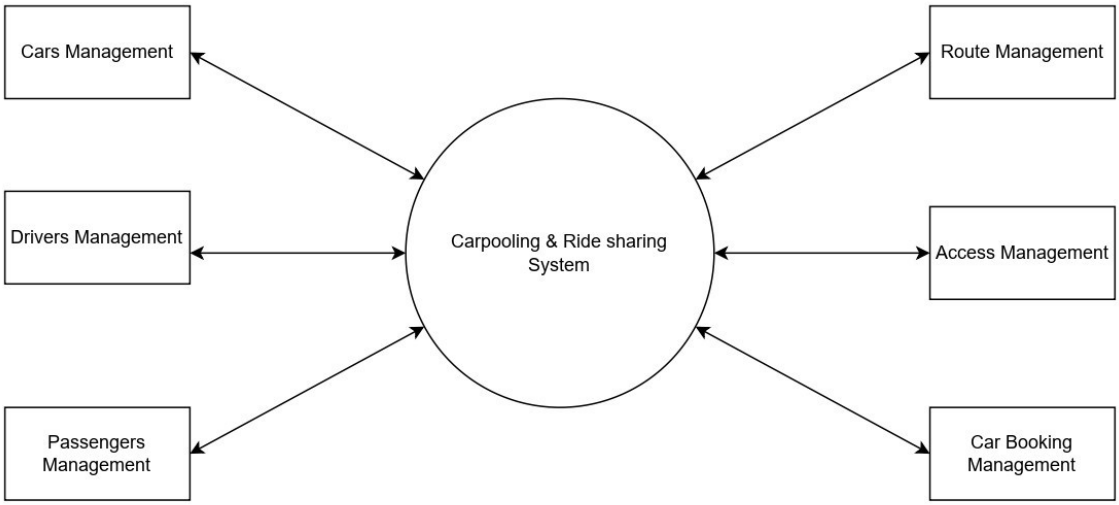

Figure 4.3: DFD Level 1

3. **DFD Level 2:**

   DFD Level 2 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail the system is functioning. In the Figure 4.4 and Figure 4.5 deep information of level 1 function.

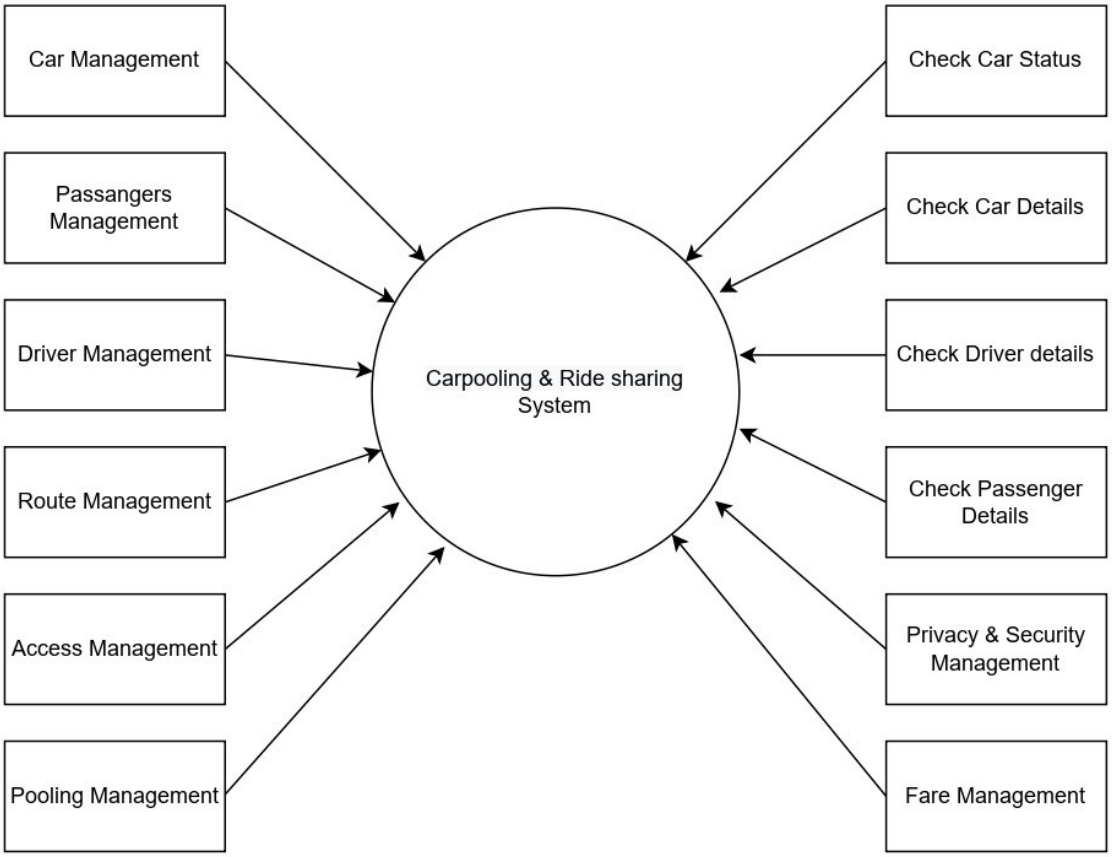

Figure 4.4: DFD Level 2

4.3 UML Diagram

Unified Modeling language (UML) is a standardized modeling language enabling developers to specify, visualize, construct and document artifacts of a software system. Thus, UML makes these artifacts scalable, secure and robust in execution. UML is an important aspect involved in object-oriented software development. It uses graphic notation to create visual models of software systems.

4.3.1 Use Case Diagram:

A Use Case diagram shows the interaction between the system and entities external to the system. These entities are called actors which have specific role in the system. The figure shows the use case diagram for proposed system. Purpose of Use Case Diagram is to know or show functionality of the system.

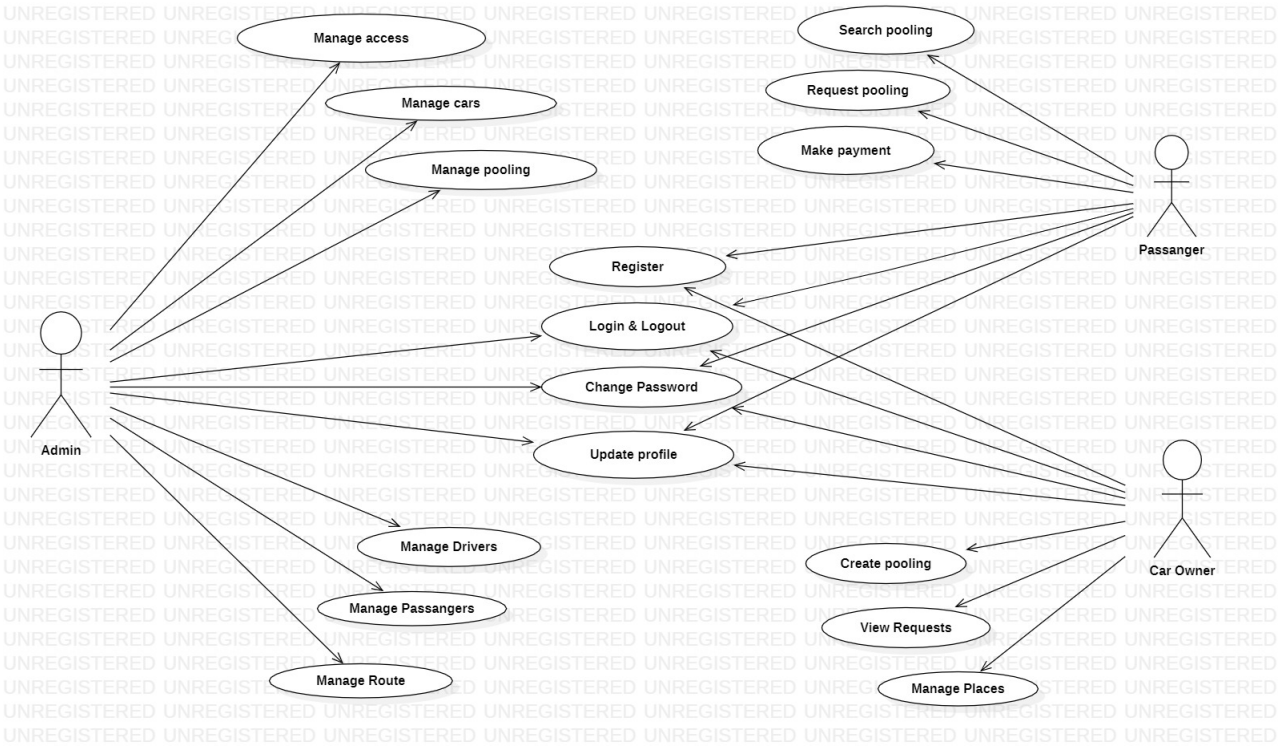

Figure 4.5: Use Case Diagram

4.3.2 Sequence Diagram:

A sequence diagram simply depicts interaction between objects in a sequential manner. Purpose of Sequence Diagram is to show flow of functionality. A sequence diagram generally consists of objects, messages exchanged between objects lifeline of each object and focus of control for each message.

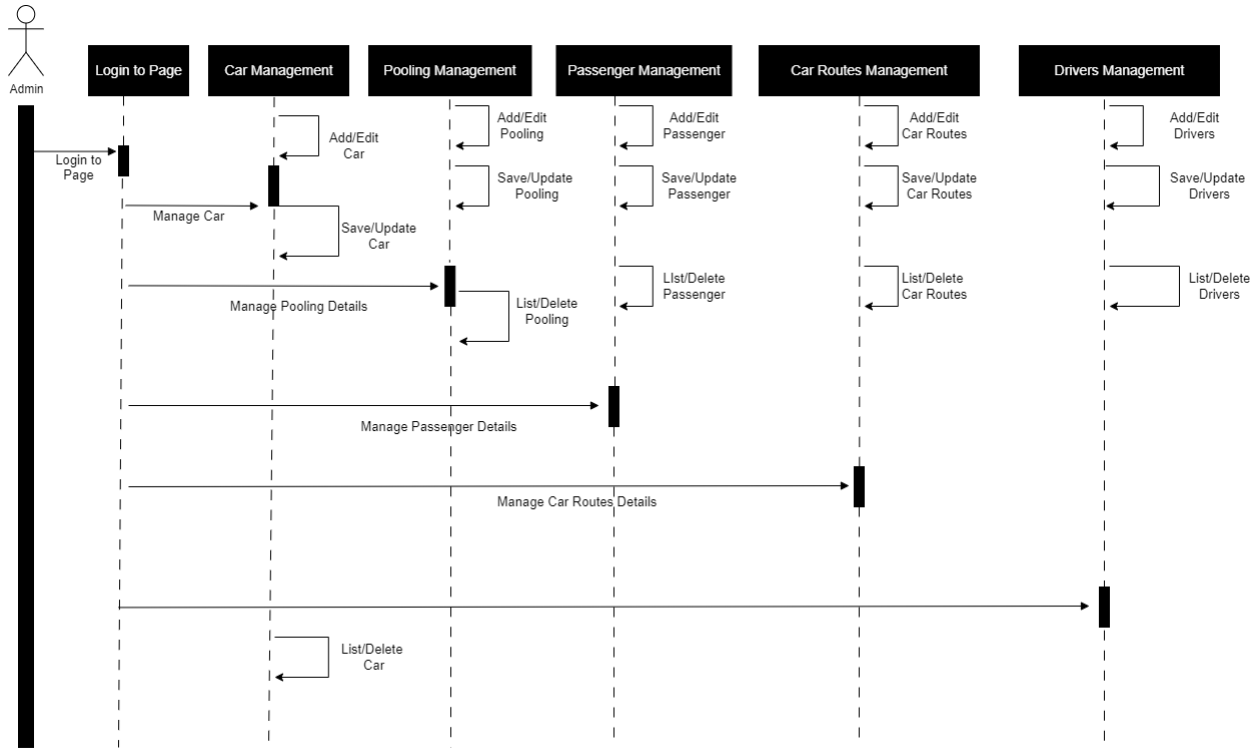

Figure 4.6: Sequence Diagram

4.3.3 Class Diagram:

A Class diagram is used to represent the static view of the system. It mainly used classes, interfaces and their relationships. The Figure shows the class diagram for proposed system. Purpose of Class Diagram is to show structural aspects of the system.

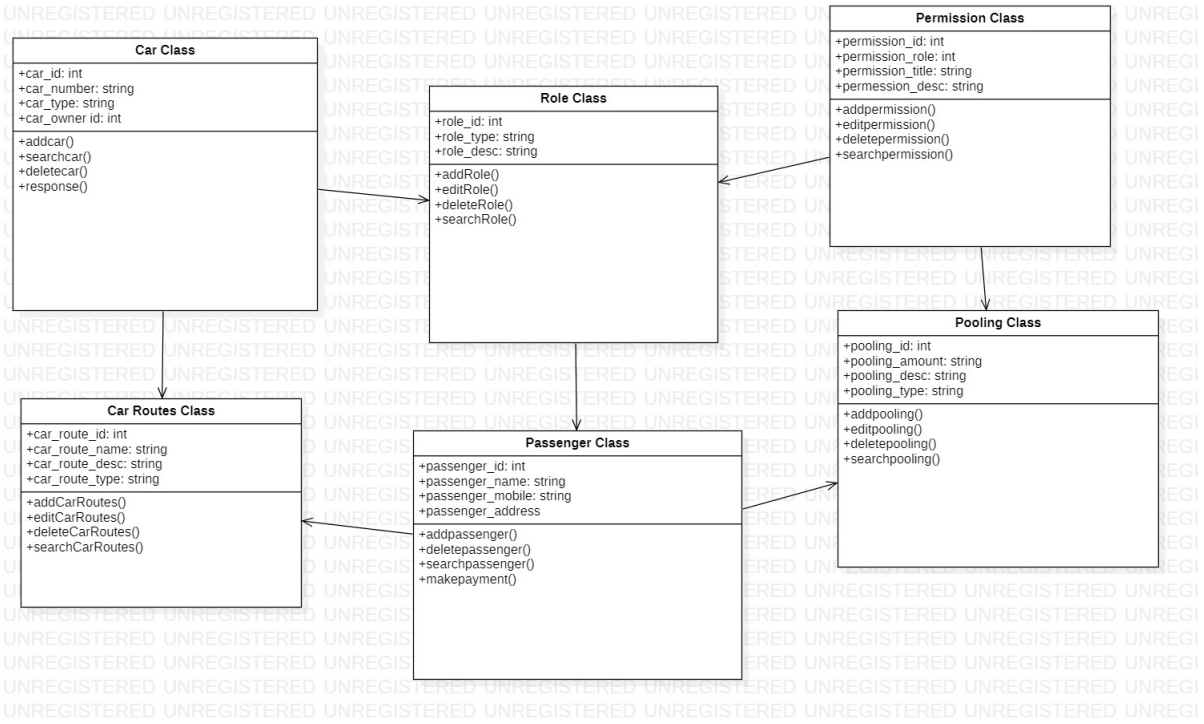

Figure 4.7: Class Diagram

4.3.4 Component Diagram:

A component diagram is generally used to show a set of components and their relationships. Graphically a component diagram is a collection of vertices and arcs where components are the vertices and the relationships form the arcs. Particularly components are connected with the help of dependency relationship that shows the dependencies among various components.

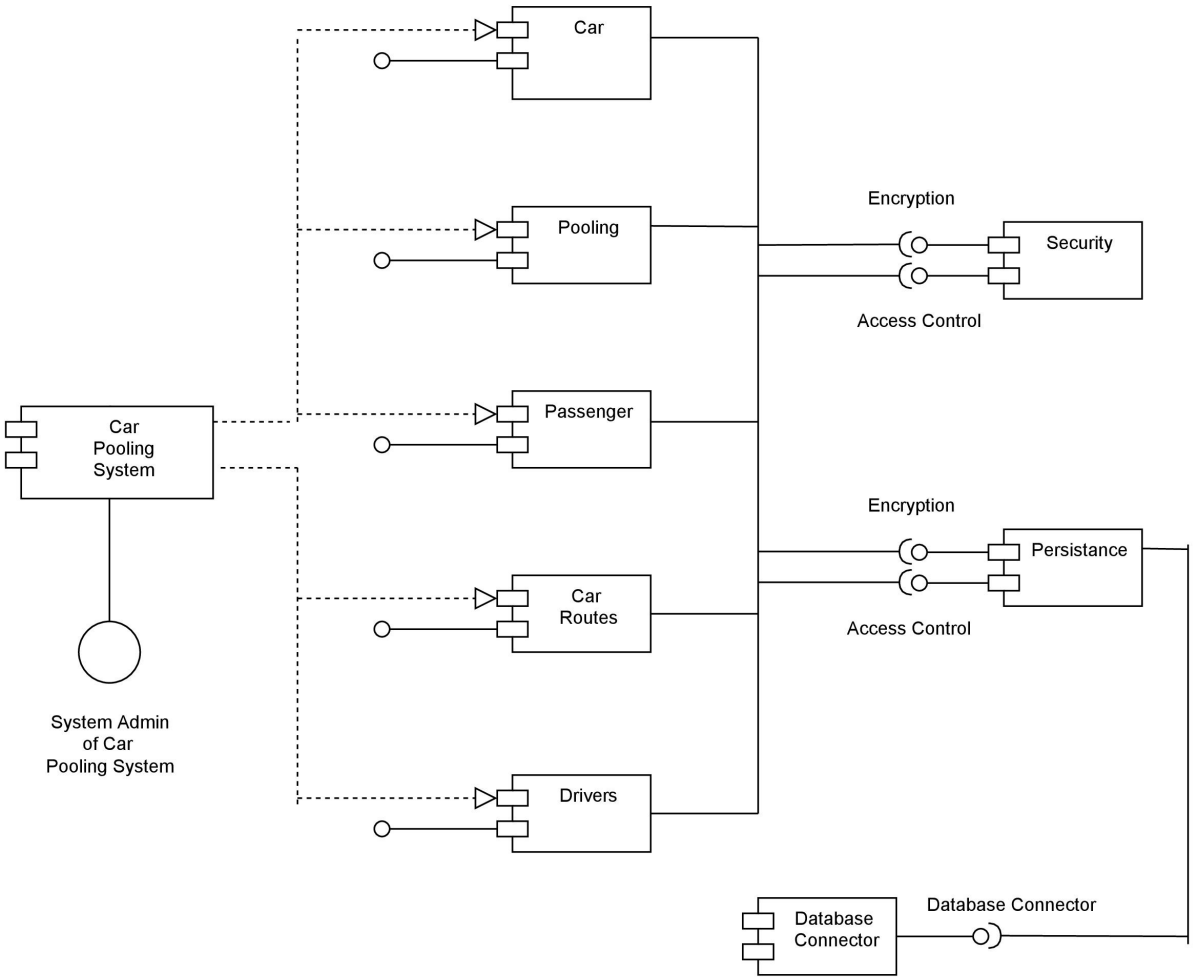

Figure 4.8: Component Diagram

4.3.5 Deployment Diagram:

A deployment diagram shows the configuration of the run-time processing nodes and the components that reside on them. Deployment diagram addresses the static deployment view of a system. It is related to component diagram where a node typically encloses one or more components.

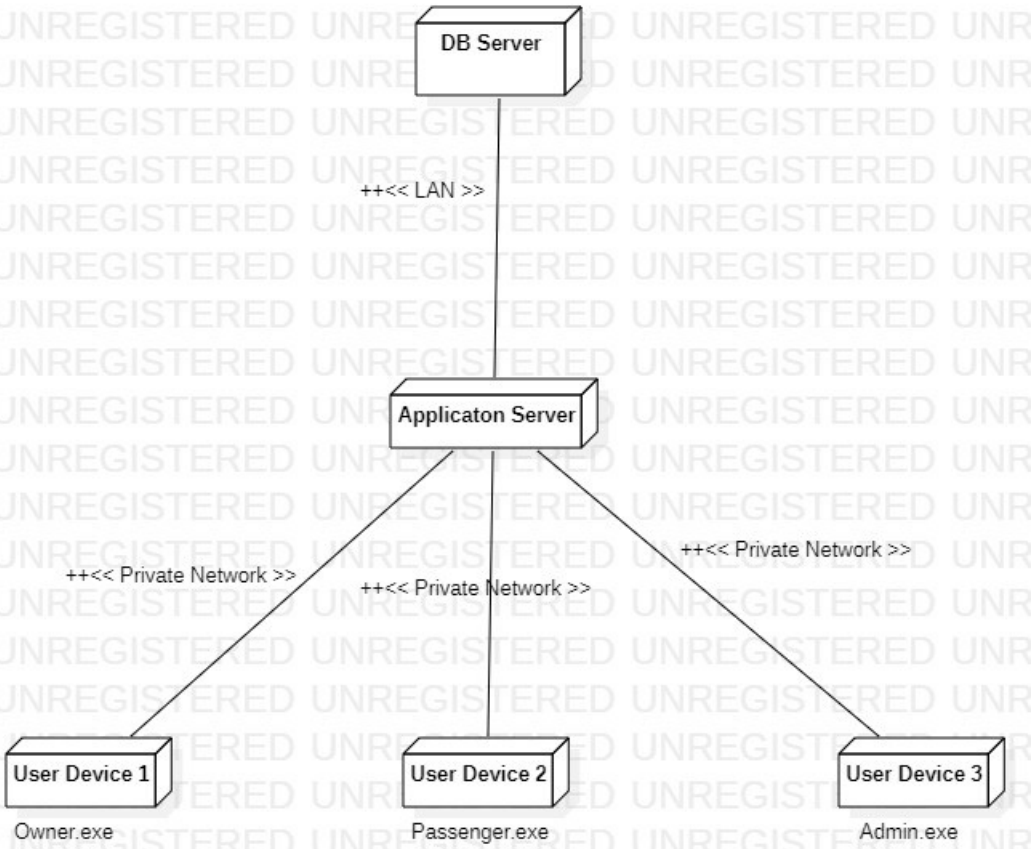

Figure 4.9: Deployment Diagram

4.3.6 Activity Diagram:

An activity diagram shows the basic activities between two intermediate states of a state chart diagram. Activity diagram shows the flow from activity to activity. An activity is an ongoing non-atomic execution with a state machine. Activity ultimately results in some action which is made up of executable atomic computation that results in a change in state of system or the return of value. The activity diagram is a collection of vertices and arcs and forking, joining operations.

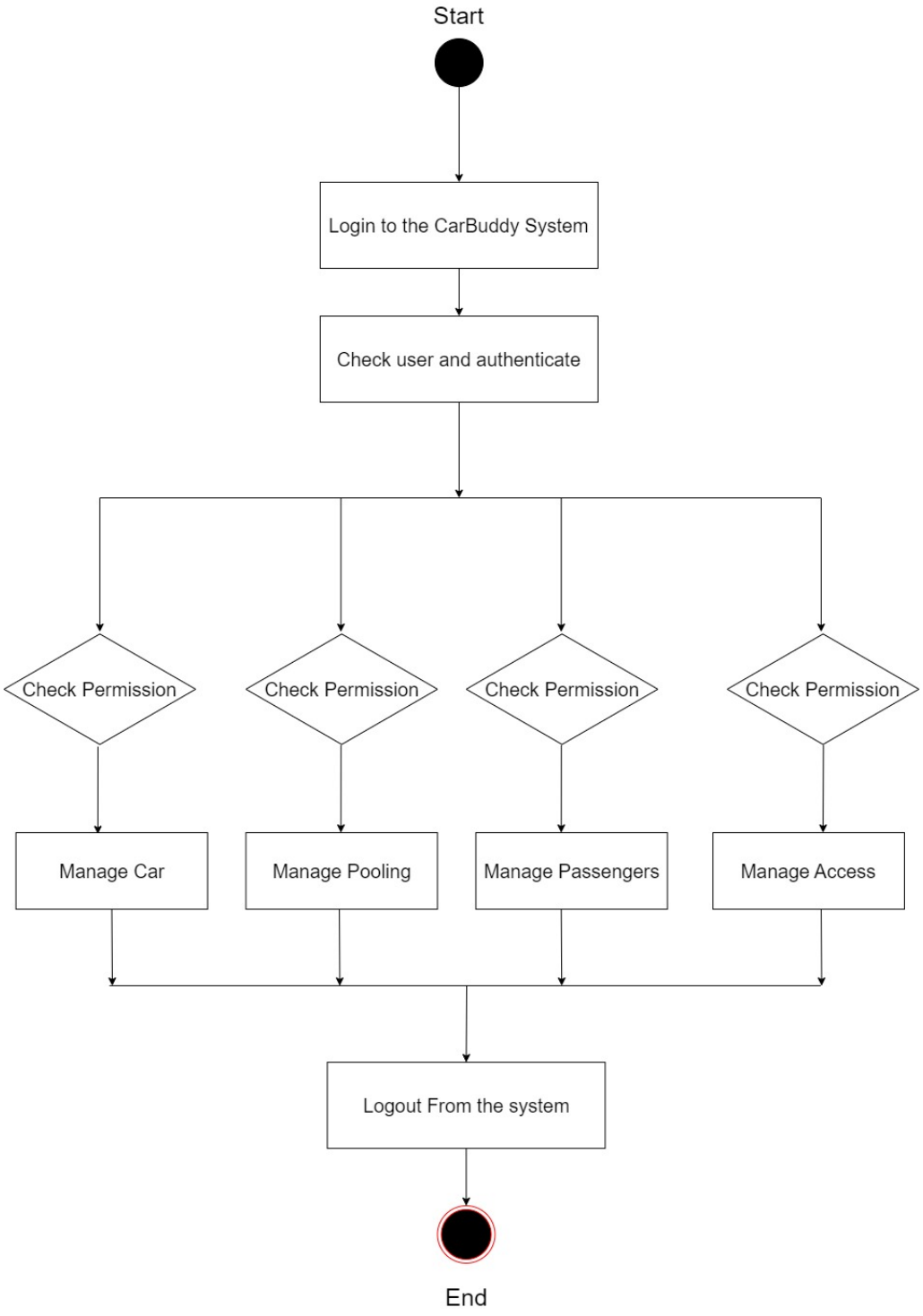

Figure 4.10: Activity Diagram

4.4 Summary

In this chapter , Design is described. In the next chapter, Conclusion and Future Work is presented.

Chapter 5

Implementation

Implementation phase is longest and most important phase in software development. When the designing of the software is completed, then a group of developers starts coding of the design using a programming language. The interface of the software and all its internal working according to design phase is implemented in implementation phase.

This chapter mainly contains following sections: Section 5.1 describes the Algorithm/Steps, Section 5.2 describes the Hardware and Software Requirements, Section 5.3 describes the Project Modules, Section 5.4 describes the Summary.

5.1 Algorithm/Steps

1. Collection of requirement analysis.

2. Creation of UI in Figma Software.

3. Designing of API endpoints.

4. Start developing UI/Frontend of website.

5. Creation of database.

6. Start developing backend apis.

7. Integrate UI with Backend(APIs).

8. Testing.

9. Project Complete.

5.2 Software and Hardware for development in detail

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product.

- WINDOWS 7 or higher.
- Python 3.7.0 or higher.
- Visual Studio Code.
- 2GB RAM (minimum).
- Postgresql Database
- 100GB HDD (minimum).
- Intel 1.66 GHz Processor Pentium 4 (minimum) .
- Internet Connectivity.

5.3 Modules in Project

In our Application there are many modules so we develop each module separately. When all modules are ready then integrated all the modules into one application. This section briefly describe all the modules and the functionality of these modules.

- Flask (website backend python framework).
- Passlib(for hashing passwords).
- Flask-wtf(for handling html forms at ease).
- psycopg2(database for data handling).

5.4 Summary

In this chapter, we described the topics like Implementation details, Hardware and software requirements in details, Modules of project and summary. In the next chapter, Testing and Test cases are presented.

Chapter 6

Testing

Testing goes side by side with the implementation is aimed at ensuring the system works accurately before the live operation is performed. The common view of testing held by the user is to ensure they are no errors in a program. Testing usually means the process of executing a program with explicit intention of handling errors. It depends on the process and the associated stakeholders of the project. The chapter mainly contains following sections:- Section 6.1 describes the Black Box Testing. And the White Box Testing is described in the Section 6.2. Summery of the chapter is described in the section 6.3.

6.1 Black Box Testing

Black Box testing also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. Tests can be functional or non-functional, though usually functional. The method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. such method attempts to find errors in the following categories:

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external database access.
- Behavior or performance errors.
- Initialization and termination errors.

6.1.1 White Box Testing

White Box Testing is also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing. It is a software

testing method in which the internal structure, design, implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a 41 system. The method is named so because the software program, in the eyes of the tester, is like a white or transparent box; inside which one clearly sees.

6.2 Manual/Automated Testing

A test case is a set of conditions under which tester will determine whether an application, software system or one of its features are working as it was originally established for it to do software system or one of its features are working as it was originally established for it to do testing.

6.3 Test Cases Identification and Execution

A test case is a set of conditions under which tester will determine whether an application, software system or one of its features are working as it was originally established for it to do software system or one of its features are working as it was originally established for it to do testing.

6.4 Summary

In this chapter we discussed the sections named testing details, testing types, test case identification and execution. In next chapter result and discussion chapter are covered.

Chapter 7

Results

The results section is a section containing a description about the main findings of a research,Section 7.1 describes the Result Section 7.2 web applications snapshots and Section 7.3 contain Summary.

7.1 Result

After doing the design and implementation of the above system it is founded that after following the software development life cycle thoroughly it can help us to create project with ease.Also with the help of testing, the software can be bugs or errors free.

7.2 Snapshots of Web Application

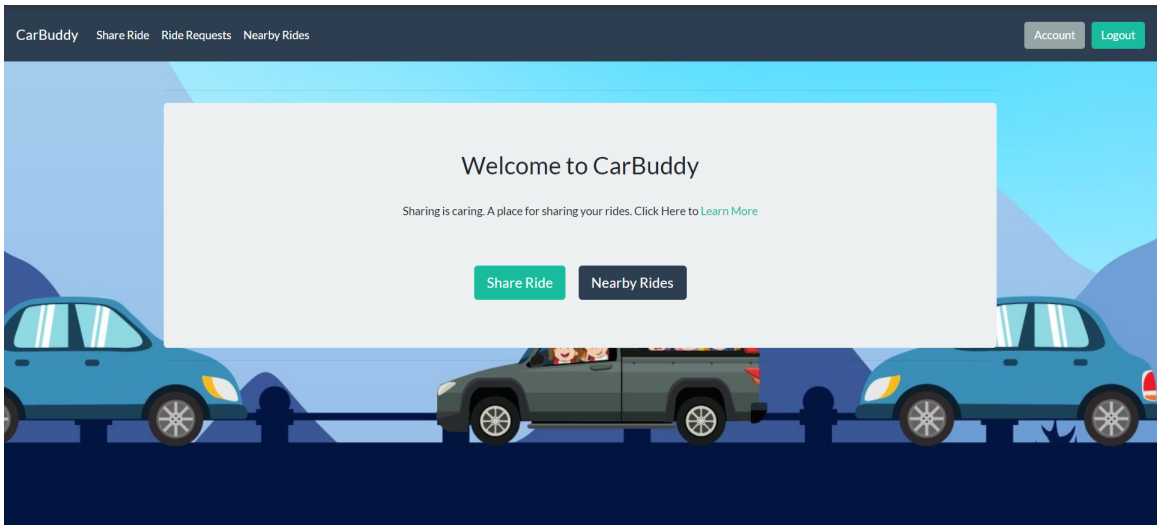

Figure 7.1: Homepage

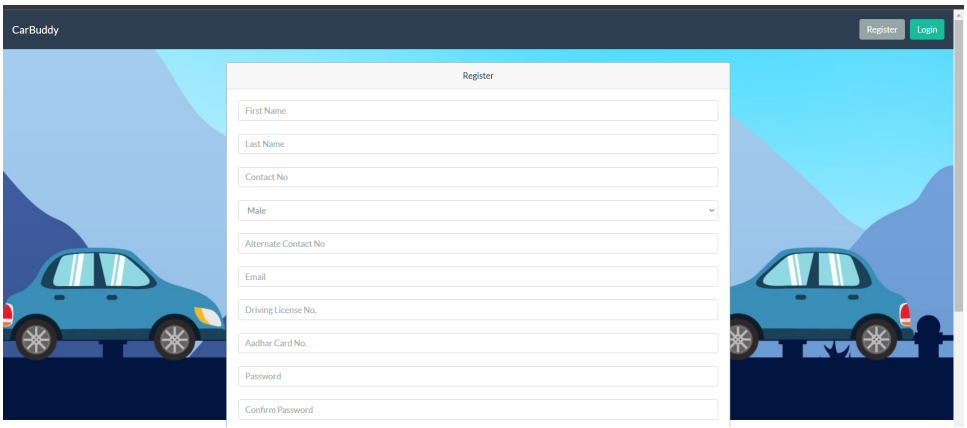

Figure 7.2: Registration

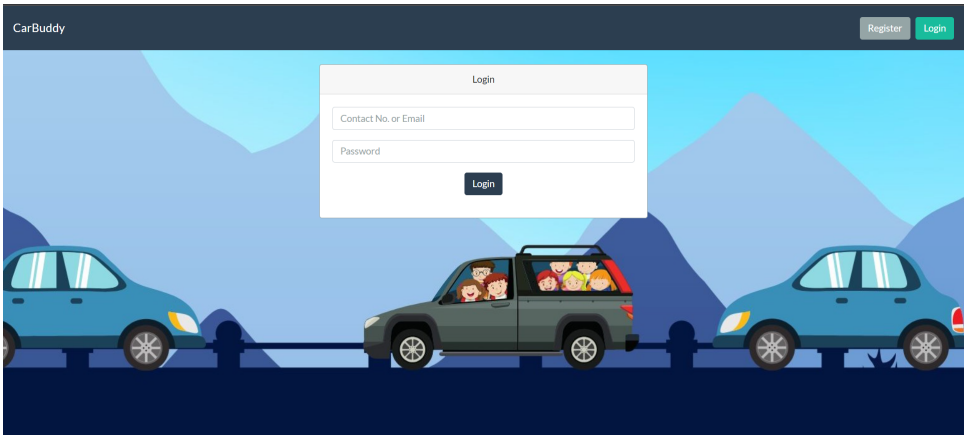

Figure 7.3: Login

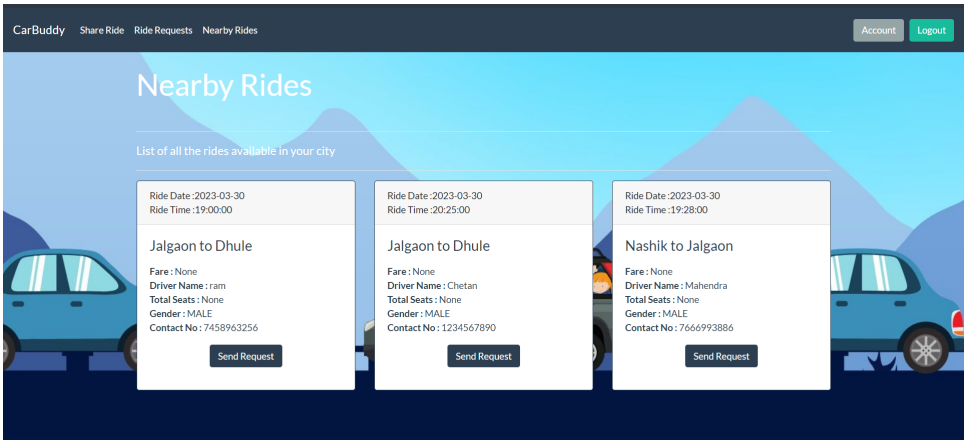

Figure 7.4: Nearby Rides

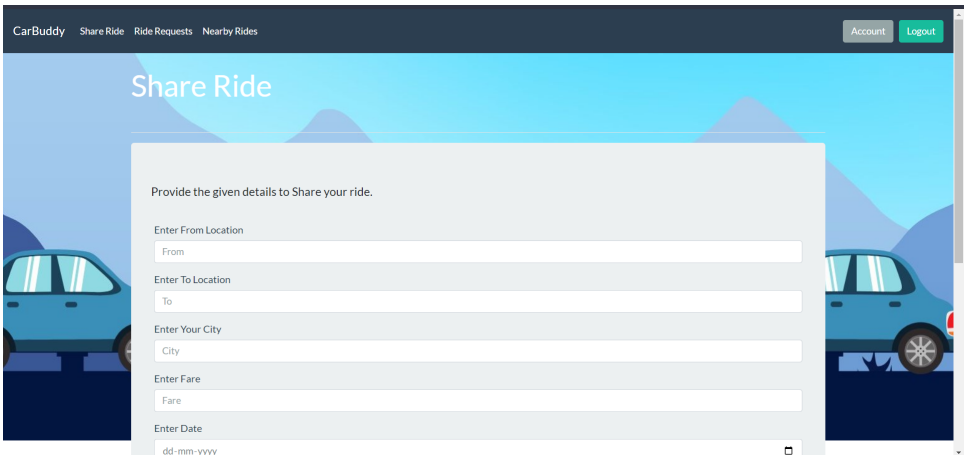

Figure 7.5: Share Rides

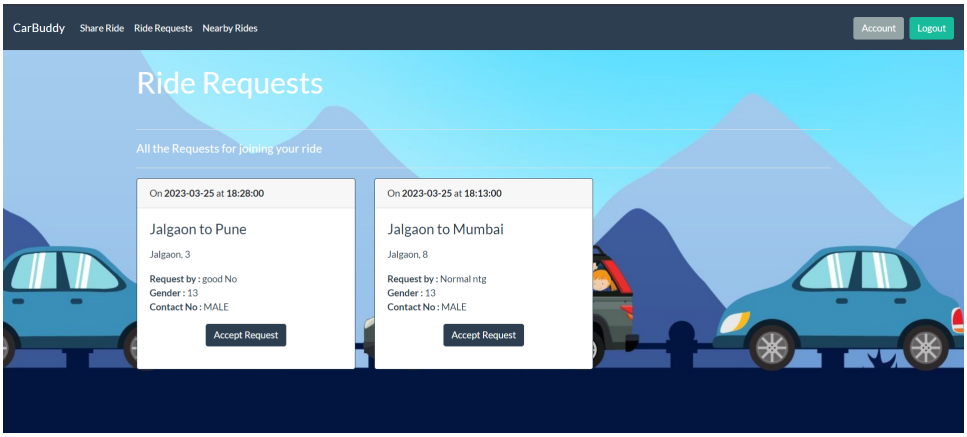

Figure 7.6: Ride Requests

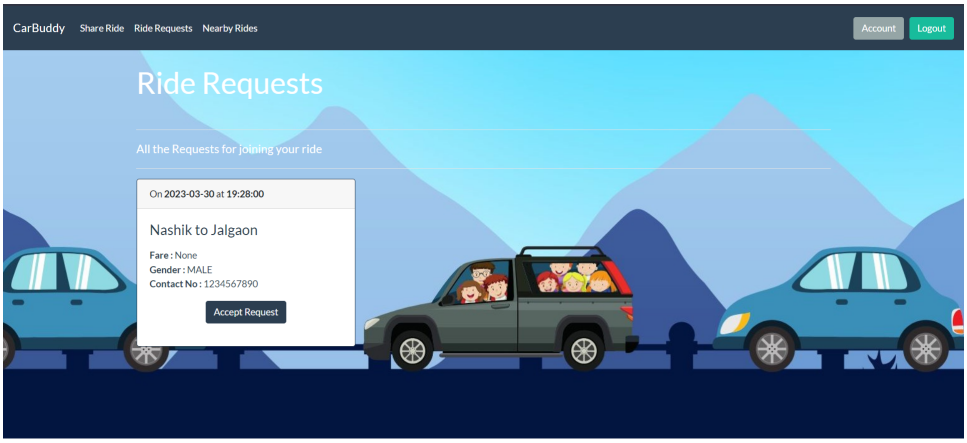

Figure 7.7: Accept Request

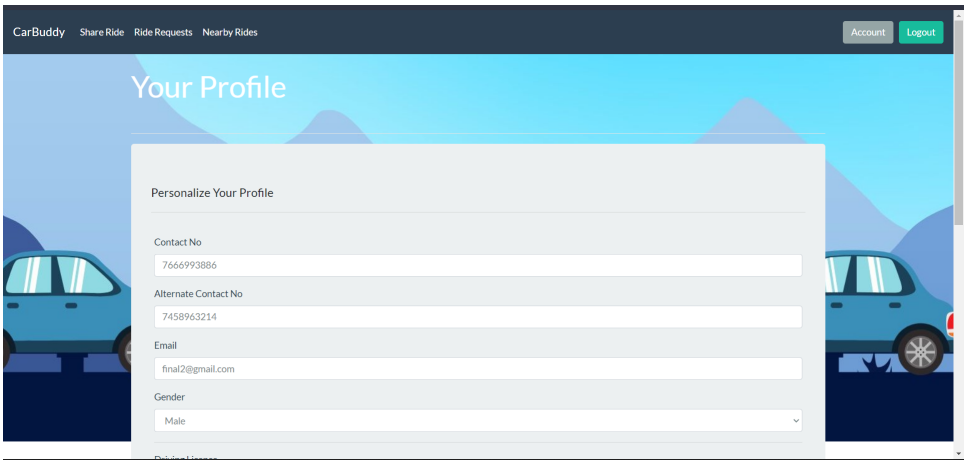

Figure 7.8: User Profile

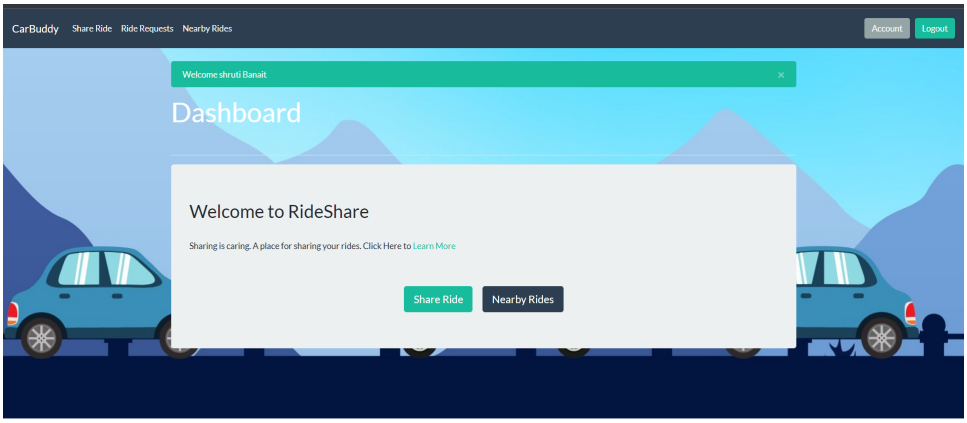

Figure 7.9: Dashboard

7.3 Summary

In this Chapter we discussed about the Result of the project and snapshots of application is presented. In the next chapter, Conclusion and future work is described.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Carpooling helps in reducing environmental pollution, traffic congestion and lack of space for parking areas. So, it is an environment-friendly social web application and also helps people to reduce their journey time. It save thousands of amount spend on, fuel expense, vechicle wear, parking Tools,etc. It helps to meet new people. The more you carpool, the more your environment will thank you. So drive less and make a positive contribution to your environment. With the help of this system many people will get benefited with it. as it will be cost effective and also it will follow all safety parameters before final deployment.

8.2 Future Work

As part of future work, inclusion of GPS system, new tracking and monitoring methodology along with various others algorithm for localization, stipulated journey time calculation and driver to user mapping can be performed. also intend to build the similar application for different domains like cargo pooling etc.

Bibliography

[1] *Neo L F.* (Singapore Med J) [*Working Toward Best carpooling system*].

[2] *Manorama Mienam.* (Assam) [*Driver- passenger Communication and Passenger Satisfaction: A Sociological Study*].

[3] Medical Apps
https://blog.capterra.com/carpooling system/

[4] Applications
https://cliniciantoday.com/the-3-best-apps-for-carpooling/

[5] *Latex Templates.*
https://www.overleaf.com/latex/templates/

[6] *UML Diagrams*
https://online.visual-paradigm.com/

[7] *Images*
https://www.google.co.in/imghp?hl=en&tab=ri0&ogbl

[8] *Software*
https://sites.google.com/site/himfinalreport/chapter-6-feasibility-study

[9] *Feasibility Information*
http://www.tutorialsspace.com/Software-Engineering/SE-2nd-Unit/02-Requirement-Engineering-Phases-Feasibility-Study.aspx

Index