DECENTRALIZED CARPOOL

By

KUDAKWASHE EXSTAFF KOTI

(H190573E)

HIT400 Capstone project Submitted in Partial Fulfillment of the

Requirements of the degree of

Bachelor of Technology

In

Software Engineering

In the

School of Information Sciences and Technology

Harare Institute of Technology

Zimbabwe

Supervisor

MISS CHIBAYA

05/23

HIT 400 /200 Project Documentation Marking Guide

ITEM	**TOTAL MARK /%**	**ACQUIRED/%**
PRESENTATION- Format-Times Roman 12 for ordinary text, Main headings Times Roman 14, spacing 1.5. Chapters and sub-chapters, tables and diagrams should be numbered. Document should be in report form. Range of document pages. Between 50 and 100.Work should be clear and neat	**5**	
Pre-Chapter Section Abstract, Preface, Acknowledgements, Dedication & Declaration	**5**	
Chapter One-Introduction Background, Problem Statement, Objectives – smart, clearly measurable from your system. Always start with a TO… Hypothesis, Justification, Proposed Tools Feasibility study: Technical, Economic & Operational Project plan –Time plan, Gantt chart	**10**	
Chapter Two-Literature Review Introduction, Related work & Conclusion	**10**	
Chapter Three –Analysis Information Gathering Tools, Description of system Data analysis –Using UML context diagrams, DFD of existing system Evaluation of Alternatives Systems, Functional Analysis of Proposed System-Functional and Non-functional Requirements, User Case Diagrams	**15**	

Chapter Four –Design Systems Diagrams –Using UML Context diagrams, DFD, Activity diagrams Architectural Design-hardware, networking Database Design –ER diagrams, Normalized Databases Program Design-Class diagrams, Sequence diagrams, Package diagrams, Pseudo code Interface Design-Screenshots of user interface	**20**	
Chapter Five-Implementation & Testing Pseudo code of major modules /Sample of real code can be written here Software Testing-Unit, Module, Integration, System, Database & Acceptance	**20**	
Chapter Six –Conclusions and Recommendations Results and summary, Recommendations & Future Works	**10**	
Bibliography –Proper numbering should be used Appendices –templates of data collection tools, user manual of the working system, sample code, research papers	**5**	
	100	**/100**

Certificate of Declaration

This is to certify that work entitled Decentralized Carpool *is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology (Hons) in Software Engineering ,Harare Institute of Technology .It is further certified that no part of research has been submitted to any university for the award of any other degree .*

(Supervisor)	Signature……………………………..	Date……………………….
(Mentor)	Signature……………………………..	Date………………………
(Chairman)	Signature………………………………..	Date………………………..

ABSTRACT

Existing carpooling optimization techniques based on a centralized approach serve policy makers' goals but neglect the reality of participants. Moreover, without strict enforcement, participants often ignore centralized solutions and maximize their own savings. We present a new heuristic, formulated and tested on real and simulated carpooling problem cases, that mimics the decentralized carpool self-organization process. Our findings reveal system-wide savings similar to centralized models and a potential strategy for improving carpooling use. Decentralized ridesharing is a system in which individuals can arrange rides with each other without the need for a centralized organization or platform. Instead, participants use a peer-to-peer network to connect with other riders or drivers traveling in the same direction. This system enables more efficient use of resources, reduces traffic congestion and carbon emissions, and provides a more affordable alternative to traditional modes of transportation. The decentralized nature of the system also provides greater flexibility and autonomy for participants to arrange rides on their own terms without relying on a third-party platform or service. Overall, decentralized ridesharing has the potential to change the way we think about transportation and contribute to a more sustainable and equitable future. The system proposed by this study combines route customization, i.e seamlessly integrated with GPS, Google Maps, Android mobile app and Firebase database.

PREFACE

Transport is a vital aspect of modern society, providing mobility and access to resources, services and opportunities. However, traditional modes of transport, such as single-occupant vehicles, can be inefficient, costly and harmful to the environment. In recent years, there has been a growing movement towards more sustainable and fairer transport solutions, including carpooling.

Carpooling is a simple but effective way to reduce traffic congestion, carbon emissions and transport costs by sharing rides with others traveling in the same direction. However, traditional ridesharing systems are often centralized and rely on third-party platforms or organizations to connect drivers and riders. This can limit the flexibility and autonomy of participants and create unnecessary barriers to entry.

Decentralized carpool is a new approach to carpooling that uses the power of peer-to-peer networks to connect drivers and riders directly. By removing the need for a centralized organization, decentralized ridesharing provides greater flexibility, autonomy and cost-effectiveness for participants, while contributing to a more sustainable and equitable transport system.

The purpose of this project is to explore the concept of decentralized ridesharing and its potential to change the way we think about transportation. We will discuss the benefits and challenges of this approach, explore real-world examples of decentralized ridesharing in action, and provide practical advice to individuals and organizations interested in implementing a decentralized ridesharing system. I hope to inspire a new generation of transportation solutions that prioritize sustainability, equity and community engagement.

ACKNOWLEDGMENTS

Glory be to God Almighty, for with Him all things are possible. I am extremely grateful to him for giving me the strength to work on this project in difficult times. Thanks to all my friends and relatives who helped me. I would like to thank Miss Chibaya, my supervisor, for her tremendous support, helpful comments, notes, feedback and involvement in the learning process of this research project. I am grateful for all his support during this journey. I was also lucky to have the help of the lectures who wrote me useful suggestions and recommendations. I would also like to thank the University Of Harare Institute Of Technology for providing me with adequate resources and knowledge needed to develop my project. Furthermore, I would also like to express my deep gratitude to my colleagues Adrian Nzvimbo, Christopher Meki, Tinotenda Katsamba and classmates for their generous help. Last but not least, I thank my parents, my sister and family for showing me such unwavering love and support. I am so grateful for all the sacrifices you have made to ensure my happiness and success.

DEDICATION

To all commuters who are tired of sitting in traffic, wasting time and money on their daily commute, this project is dedicated to you.

I believe that ride-sharing can be an effective solution to the challenges of modern transportation, and I am determined to make it simpler and more efficient than ever before.

This project is also dedicated to the countless individuals and organizations working to build a more sustainable future. By reducing the number of cars on the road and promoting shared transport, I hope to help reduce carbon emissions and create a cleaner and healthier environment for everyone.

Finally, I would like to dedicate this project to the many volunteers, collaborators and supporters who have helped me along the way. Without your enthusiasm, determination and hard work, this project would not be possible. I am grateful for your contributions and look forward to working together to make ridesharing a more viable and attractive option for commuters around the world.

Table of Contents

CHAPTER 1

Introduction

1.0 Introduction

The car-sharing market is constantly growing and recently it has become popular more than car ownership. This classic car sharing system is based on a centralized database server which can often lead to hacker attacks or password leaks. Moreover, in a classic car-sharing system the owners of the car can misuse customers' data. Nowadays we have come to see from a lot of use cases, that the best solution is to use blockchain technology. The blockchain technology is decentralized, immutable, public ledger provides the customers with security that is impossible to tamper with. The aim of the proposed system is to create and implement peer-to-peer short term car-sharing application that is based on decentralized just like the blockchain technology.

1.1 Background

In recent years, the issues of worldwide warming and therefore the energy crisis have aroused widespread public concern. One recommended solution for reducing the harmful factors resulting in such problems is decentralized carpooling system. Carpooling has gained much attention due to the rise of environmental concerns and also the congestion of roads. It has also gained tons of attention due it being environmental friendly and also a cheap means of traveling. Carpooling is when people share a ride in one among their personal cars. Carpooling reduces pollution since we have less cars on the road. The proposed system reduces pollution since we have less cars on the road. Travelling alone could also be stressful, so having other people with you on a journey reduces the strain and is additionally the occasion to socialize and make the trip funnier. What people do not understand is that sharing a ride provides a chance to scale back pollution, cuts down on gas and oil consumption, and trims down the value of commuting. The purpose of the application is to supply a platform to bring like-minded people together, promote people to require up the thought of carpooling, help them coordinate, improve transparency, ensure data protection, and prevents anyone to tamper with the system. The platform encourages carpooling which lets people use the same vehicle thereby taking a few vehicles off the road, meaning less traffic, commuting time becomes quick and also many free parking spots. Since the platform is decentralized it also makes sure that the users' information can't be tampered with, the service becomes practically impossible to tamper and increases transparency.

1.2 PROBLEM STATEMENT

Lately there is a problem of traffic on our roads and therefore the increasing fuel costs boost the misery of daily users of private vehicles. The number of vehicles plying on the road is rising as each day passes due to the low-cost cheap imports from Japan. This gives rise to few prime factors that square measure pollution, depletion and rising rates of fossil fuels and massive traffic congestion. The current system well known as mushika shika has been deemed illegal, with touts being suspected of rowdy behavior which has resulted in the inconvenience of travelers. Other solutions at hand include taxi management system, in this system the taxi driver, passenger and management system are important entities. These components interact with one other in existing taxi management systems. Because of the widespread acceptance and use of smart phones in the real world, we assume that each 1 passenger and taxi driver is equipped with a smart portable communication device, such as a smart phone or a computer. Each passenger contributes to the cab management centre, he/she must communicate his/her present position and desired destination. The taxi management centre must be established, send the taxi to carry the passengers efficiently from their designated starting points to the intended destinations. Taxi drivers who are available to convey the passenger will react to the taxi management centre to express their desire to pick up the passenger. The cab control centre then moves on to the next passenger's request. . The previously described practice of existing taxi management systems has various drawbacks. For example, passengers in a single cab cannot readily share a journey with one another to reduce costs. The transportation system is inefficient. As a result, there is no way to minimize air pollution and carbon emissions. Carpooling entails two or more passengers traveling in the same direction in a private car along a partially shared route. The idea of carpooling is that some individual passengers' journey time is sacrificed in order to receive additional benefits such as sharing driving costs, parking fees, reduced traffic congestion, or lower carbon emissions. Unfortunately providing carpooling with general systems to users brings a lot of security flaws, first of all is overall stability of service. So decentralization will add transparency and ensures data protection as it is termed an unchangeable method.

1.3 OBJECTIVES

The objective of the project is to present a decentralized android mobile application which provides a communication platform between the car owners and passengers. It will allow the car owners to post a notice notifying the other users about a route they will user whether regularly or just once so as to search for travel-mate in order to reduce ride costs. It will be implemented in phones with the android software. The application will try to cover:

- To create a decentralized android application where people will find car owners using the same routes as they are.
- To use google maps to get current location for a user
- To include notifications which will alert car pooler

1.4 HYPOTHESIS

Privacy policy breaches, global warming and energy issues have sparkled significant concern in the past years. Decentralized carpool is one of the recommended solutions that are capable of reducing these problems which has led to public cries. Decentralized Carpooling attempts to provide the solutions to the common needs of moving from one location to another and also providing data privacy. In the routes that are far away from towns or the routes that many people use there is inadequate transportation networks meaning there is no public transit at all during a certain period of time or the existing choices are way too expensive. It all leads to increased travelling time periods there by wasting unnecessary time traveling or people paying an extra amount more to a viable solution. For registering to carpool systems will use their private data which can later be manipulated by other people for their own use meaning their data is at risk if they are to register.

The decentralized carpool transportation service can make a significantly huge impact if it is done on a larger scale by the government or other major organizations with many branches. The schemes of using a decentralized carpooling are particularly made so as to encourage car owners to share traveling expenses and resources with customers whilst also ensuring that users' private information is safe. The solution to the need of traveling whilst reducing global warming, congestions and energy issues is carpooling whereby individuals traveling to the same destination in the same route use the same vehicle whilst dividing their costs in a more convenient manner. The solution to data privacy and transparency is to use a decentralized carpooling. Since today there is such availability from a users' view with smart phones it becomes possible to have such a technological feasible solution.

1.5 JUSTIFICATION

This project put forward a technological response that facilitates and promotes decentralized carpool to interested parties at the same time providing a better ecological and health environment, promoting social engagement, ensuring data privacy and increasing transparency of the system. Creating and launching a decentralized carpooling system is the goal. The proposed system is not trying to replace the current transportation networks nor is it the solution to our problems but rather it is there to act as a complimentary method of achieving the same goals of lowering average wait times and prices and ensuring data privacy. The aim is to provide an internet based platform where drivers can share their automobiles and customers may look for vacant seats whilst ensuring that their data is kept private. The proposed system will try to address the flaws of the existing system. The system will give immutable information on the car owner and the vehicle he or she is using so as to preserve transparency among system users. The location of the customers in the pool will be tracked through GPS Navigation system. The platform will aim to be user friendly and will be accessible anywhere on an android application as long as the user has a smart phone and an active smart phone. The intent is to use the best software development practices and more advanced technologies.

1.6 PROPOSED TOOLS

Android:

The developed android application run on top of Android Application framework of the Android Operating system.

Google Play Services:

Provides an interface to the application that it can use to access Google powered features such as Google maps.

Firebase:

Firebase is a platform developed by Google for creating mobile and web applications Pc with RAM 4Gb, disk 500Gb

1.7 Feasibility study

A feasibility study is an analysis that takes all of a project's relevant factors into account including economic, technical, legal, and scheduling considerations to ascertain the likelihood of completing the project successfully. Feasibility study is used to discern the pros and cons of undertaking a project before they invest a lot of time and money into it.

1.7.0 Technical feasibility

Technical feasibility aims at analyzing whether the required technology is available or not and also the availability of the required resources. Technical feasibility of the proposed system may be done by just looking at the software and hardware available and analyzing them without necessarily building the system. The technology for the proposed system is shown below:

1.7.0.1 Hardware requirements

Table 1 hardware required

DESCRIPTION	QUANTITY	AVAILABILITY
Laptop/desktop	1	yes
Visa card	1	yes

As shown on the table above the hardware required to complete this project is readily available, therefore making the proposed system technically feasible.

1.7.0.2 Software requirements

Table 2 software requirements

DESCRIPTION	QUANTITY	AVAILABILITY
Windows Operating System	1	yes
Visual Studio code	1	Yes
Google Chrome	1	yes

Metamask account	1	yes
Android studio	1	yes
Google account	1	yes
Alchemy account	1	yes

1.7.1 Economic feasibility

The focus here is on checking if the cost of developing the whole system won't surpass the proposed budget. Cost–benefit analysis showed that it was possible for the development of the application to progress. Most developing software I will be using are open source hence they will be available free of charge. The hardware with required processing power is also already available hence the cost is covered hence the project is economically feasible

1.7.2 Operational feasibility

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. It answers question "will the system work?" and how well the system would fit into the current business structures and once implemented how useable would it be. Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. After analyzing these operational requirements, we see that the proposed system is operationally feasible.

1.8 Project Plan

A project plan is a formal document designed to guide the control and execution of a project. A project plan is the key to a successful project and is the most important document that needs to be created when starting any business project.

Table 3 Project plan

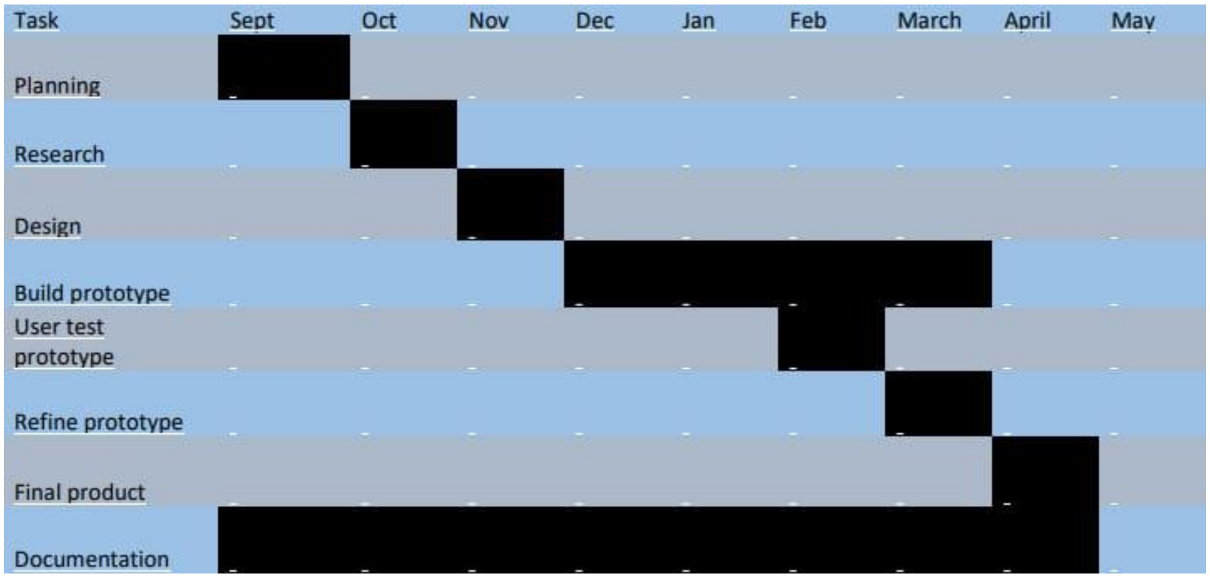
PHASE	DURATION(days)	STARTING DATE	ENDING DATE

Requirements analysis and definition	35 days	1 September 2022	5 October 2022
System design and software design	112 days	6 October 2022	25 January 2023
Implementation and unit testing	20 days	26 January 2023	14 February 2023
Integration and system testing	29 days	15 February 2023	15 March 2023

1.8.0 Gantt chart

Gantt charts convey this information visually. They outline all of the tasks involved in a project, and their order, shown against a timescale. This gives you an instant overview of a project, its associated tasks, and when these need to be finished.

Table 4 Gantt chart



Task	Sept	Oct	Nov	Dec	Jan	Feb	March	April	May
Planning	■								
Research		■							
Design			■						
Build prototype				■	■	■	■		
User test prototype						■			
Refine prototype							■		
Final product								■	
Documentation	■	■	■	■	■	■	■	■	

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

Carpooling is the whereby people or commuters traveling to the same destination or en route in the same direction share the same vehicle to fulfil their journeys. It can relate to carpooling, vehicle sharing, ride sharing and so on but the concept is the same. In Zimbabwe we do not currently have any online carpooling that is decentralized. Vaya is the only solution that is close to carpooling and it is not decentralized. Worldwide there many carpooling systems using blockchain technology or that are decentralized for example Uber in USA is now running on blockchain. Many studies being conducted are giving an analysis of the benefits of using a decentralized carpooling system. The most important purpose of using a decentralized carpooling system is to reduce travel time and transportation expenses whilst maintaining privacy policy. Decentralized carpool provides a tamper-proof booking system and makes anyone become part of the chain. Decentralized carpool system is seen as an excellent alternative to public transit and taxi services. Carpooling also provides the chance of traveling with people who have similar interests to yourself making traveling a fun way to kill time. It also deals with driving stress and also cuts down overall trip time. Carpooling is a way for communities to minimize traffic congestion and pollution. Some studies identified environmental responsibility as a motivation for carpooling. Others, on the other hand, emphasized the social benefits of carpooling: reduced fuel use, less CO2 emission, less traffic congestion, and greater social contact. Some studies identified the need to maintain privacy policy in carpooling as a motivation to use blockchain technology or use a decentralization technique.

2.2 RELATED WORK

HireGo

It is built on Ethereum using smart contracts as a distributor of virtual fungible tokens (standard ERC-20) [1] called HGO. If one wants to rent a car using HireGo, he or she has to exchange ETH to HGO tokens. HireGo's market places has three contracts. The first contract of HireGo provides HGO tokens and the ERC-721 car tokens are provided by the second contract and lastly the third contract provides rental contract. The rental contract is responsible for Vehicle token between two parties and also acts as an escrow. So what happens is when one wants to rent a car, he or she sends HGO tokens to rental smart contract which locks car token and HGO tokens for selected period. After this the contract then manages the transfer of the vehicle token from owner to us. The rental solution also comes up with big security issue. If we rent a car for a day trip and then we lose the phone with Ethereum address, the car will be stuck for the rest of time and the owner is unable to help us because he/she does not own that vehicle at the time of renting.

Helbiz

Its philosophy is to create and integrate transport ecosystem where users are rewarded for using this system. The project is using well-known implementation of blockchain car-sharing platform by providing ERC-20 tokens called HBZ. If the user does not have enough tokens, the application offers the user to but the HBZ tokens through the use of credit card.

DAV

Its main goal is to connect autonomous vehicles with users. The use the DAV token for communication and this DAV token also acts as an access token, Services in the DAV network can be bought with virtual tokens based on Ethereum. If a person is owning a vehicle or charging station which he or she is providing to other users in the network he or she is rewarded with DAVE tokens [2]. Autonomous vehicles automatically interact with environment and also can fulfil user’s commands.

FFQuest

Its main strength is the use of their own chain called Distributed FFQ Ledger based on Ethereum Virtual Machine. All the transaction details between car companies, drivers and customers such as payment conditions, assets, availability and reservations by using their own utility token type ERC-20 called FFQ are passes by the ledger. This project uses central backend to mirror Distributed FFQ Ledger on a server and also for storing videos and images [3].

WONO

It provided a decentralized market for renting services such as cars and working man days. They used a set of smart contracts that work as a bridge between Ethereum main net and their blockchain. Their blockchain uses Proof-of-stake algorithm to ensure consensus in the network. The IPFS file system which is a decentralized system is used to store users; profile data represented as JSON. Data properties which contain confidential information are encrypted on a client's app side by a public key of the User's address and can be decrypted only by the privatekey [4].

Meshkani

With blockchain technology emerging quickly, significant efforts are being made to incorporate this technology in the transportation sector. Meshkani [5] suggested an empirical ride sharing method where individual users can share their ride with only a single user. Lei [6] and Abbas [7] have suggested decentralized data management systems that are integrated with blockchain and IoT to provide better transparency, data sharing and tracking.

Wang

Wand [8] introduced a secure and privacy-preserving decentralized collection of traffic information on the blockchain, TrafficChain, by taking advantage of edge computing. For making payments and transactions easier in blockchain, [9] proposed a Decentralized conditional Anonymous currency and transaction method, furthering the concept of a cashless economy. A similar effort was made by [10] to ensure that the third party services in the travel industry are eradicated and a transparent system based on using smart contracts and blockchain can be implemented.

S. C. -K. Chau [11] sheds light on the principles of decentralized ride-sharing and vehicle-pooling mechanisms based on stable matching and showed how several fair costsharing mechanisms can achieve high social optimality. Taking this study further with practical implementation, this paper aims to create a decentralized ride-sharing system using blockchain.

Kelley introduced a term "enhanced casual carpooling", referring to the casual carpooling with the incorporation of modern technology. A data-driven technique for Green and social carpooling(GRAAL) was provided by Berlingerio and collegues.Optimizing a carpooling system by reducing the number of automobiles whilst increasing the enjoymen of people sharing a journey is what the authors are attempting to do. A person's enjoyment is mainly due to his hobbies, social connections and proclivity to interact with people who share similar interests. GRAAL computes the enjoyment within a set of users from crowd-sourced data (starting from Twitter data), and then uses it on real world datasets to optimize a weighted linear combination of number of cars and enjoyment. A geo-social data to recommend users to join their friends during trips by using home location models and users' similarities is being used by Elbery et al. Cici et al. derive data using 3G Call Description Records and Twitter. The users' home and work locations, as well as their social links, are utilized to construct an algorithm for connecting people with comparable mobility patterns. Bicocchi and Mamei created a ridesharing program using a clustering method applied to tagged trajectories. Byon et al. create a Facebook10 based carpooling application and suggest Twitter-based traffic monitoring and Flickr-based incident reporting. Zahng et al. suggest a carpool service (coRide) in a large-scale taxicab network based on an analysis of the reduction in circulating taxis in the presence of ride sharing. Cloud servers dispatch costeffective carpool routes for taxicab drivers, resulting in cheaper fees for individual customers. Knapen et al. provide a system (GCPMS—Global CarpoolingMatching Service) that delivers carpooling advice by maximizing the anticipated value for successful bargaining.

Montes et al. created Teranga Go, a collaborative online consumer community based on the Senegalese population traveling by car from Europe to Africa. Ridesharing connections are built on a sense of real community, social gatherings between users, and connection to technology, with self-confidence as the core concept. To reflect experts' opinions, the authors used a multi-criteria multi-expert decision-making model using hesitant fuzzy linguistic concepts. Globally, there are cases of commercial applications that explore the concept of carpooling; the trips most of them assume are one-off long trips, but shorter recurring trips are not excluded. The most popular commercial applications are: Blablacar, the largest Polish and European carpooling solution; Amovens compartir; and ZimRide, the largest ridesharing solution in the United States. Caballero-Gil et al. [describe a mobile application that implements a trust-based social structure. The author examines many interpretations of carpooling and recommends a trust-based one that combines reputation and privacy protection. Bonhomme et al. are a multi-agent platform with an emphasis on security services that allow commuters to authenticate each other. Megalingam et al. describe an Automated Wireless Carpooling System, addressing safety and security issues.

Ronald et al. proposed an agent-based model focused on negotiation tactics. The proposed model contains a well-defined and organized interaction protocol that integrates the transport and social layers. Based on the individual and combined properties, a utility function is designed. Agents negotiate the type, location and timing of the social gathering. Nourinejad and Roorda developed centralized (binary integer programming) and decentralized (multi-agent dynamic auction) optimization methods to match passengers and drivers. Galland briefly describes the conceptual design of a carpooling app originally proposed by Cho et al., Bellemans et al. and Galland et al. On the Janus multi-agent platform, this design uses an agent-based approach. The platform allows users to choose the best mode of transportation for their needs, maintain a social network, negotiate carpools, and match car drivers and passengers. Hussain explores the organizational paradigm for long-term carpooling. The authors argue that to initiate the carpooling process, individuals' goals and intentions should match. Moreover, they emphasize that success 12 depends on time preferences, route optimization, and the effect of limiting activities. They used a multi-agent system with a social network based on the operational Flander paradigm. From the simulation results, run on the Janus multi-agent platform, they concluded that the simulation time is unacceptable for more than 40,960 agents. In addition, they offered a thorough description of temporal and geographical constraints [13]. In Net Logo, an initial version of the Dynamic Carpooling System is created for people who usually use the same route. Kothari explains the development of the Genghiz multi-agent carpooling system. The system is built using the Java Agent Development (JADE) multi-agent framework.

Ride-sharing services are gaining popularity these days for travel comfort. But carpooling services are mostly cloud-based and most services face some problems such as unnecessary communication delays, the risk of revealing users' privacy, so researchers focus on blockchain's integration with these ride-sharing platforms. [12] tried to reduce communication delay and privacy risks of users of the ride-sharing platform. New blochchain model named "CoRide" for the ride-sharing platform was introduced in [13]. [14] proposed to incorporate blockchain with carpooling system. In the proposed carpooling system, users could share vehicles with proper privacy and security.

2.3 POTENTIAL OF BLOCKCHAIN TECHNOLOGY AND DECENTRALIZATION

The key technological enablers of recent developments in distributed transaction and ledger systems are blockchain technology and the underlying distributed database technologies. Financial instruments such as payments, trading records and smart contracts can be built on blockchain technology which prevents adverse behaviour and repercussions such as forgeries and false disputes. It can also be used for legal and public records such as birth certificates, voting or court records. It can also be used to create smart property in which case blockchain becomes an inventory, tracking and buy-sell mechanism for hard assets like diamonds or cars. It can also be used for tracing the product creation for socially responsible business. Blockchain can be utilized as a transactional mechanism for "sharing economy" services [8], as it naturally solves trusted recording of large-scale peer-to-peer activities. The importance of such a transactional mechanism increases with the emerging "Programmable World" 8 where an increasing amount of physical things become programmable and get connected to the Internet.

While the technical community (including both computer science scholars and practitioners) have addressed and continue to address blockchain as a technology, our knowledge on its application beyond descriptive accounts and anecdotal evidence is quite thin. Particularly the opportunities and risks from business and societal (rather than technical) perspectives are not well understood.

2.4 CONCLUSION

We now realize that mobile technology is a broad and deep subject to master, that there are multiple ways to leverage it, and that app development is challenging and requires extensive planning and optimization. Decentralized carpooling is an effective way to reduce air pollution, parking problems, fuel consumption, commuting costs based on shared use of private cars or vehicles, mismanagement of user data, and also to increase the transparency of the system. In this paper, we study the carpooling problem and develop a decentralized carpooling prototype to implement ridesharing on the smartphone platform and Google Map API. Our proposed decentralized mobile ridesharing app aims to generate a simple ridesharing app that can help passengers and drivers and ensure no privacy violations. Finally, and after understanding the mobile technologies and solutions available for use, and understanding how car sharing work, we can determine the best options to use to build our app.

CHAPTER 3

ANALYSIS

3.0 INFORMATION GATHERING TOOLS

Information was gathered and analysed through various methodologies so as to accurately and comprehensively specify the system. This activity was done so as to ensure that each activity to be done is done completely and successfully so as to meet the expectation of the users.

3.0.1 INTERVIEWS

This data collection method was used by the developer to get information on the challenges being faced by customers as they travel day in day out to and from their work or to a destination of their own choice. I visited bus terminals in Chitungwiza and engaged travellers to get to know the loopholes of the current system and their fears of the proposed system. We used their information to truly understand the travellers' opinions and their point of view.

3.0.1.0 Advantages of interviews

- Interviews allows clarity of thoughts as the interviewer can ask further questions on anything that seems unclear
- Interviews are interactive in nature therefore they save time as compared to other methods
- Interviews allows the evaluation of facial expression which may strengthen one's facts
- There was a direct communication with individuals who will use the system hence their recommendations and needs were collected first hand

3.0.1.1 Disadvantages of Interviews

- The interviewer maybe biased and may interpret the information incorrectly in favour of his preconceived perceptions
- Information given by some interviewees may have been biased in order to protect themselves
- Some interviews failed to come through

3.0.2 Questionnaires

Questionnaires were helpful in instances where the people who were supposed to participate in an interview were not available or occupied at the time. Questionnaires were mainly distributed to the people commutes daily to work. These questionnaires were filled without any form of supervision and anonymity was ensured however the respondents would sometimes just fill in the answers without understanding the question.

3.0.2.0 Advantages of Questionnaires

- Economy of time. Several subjects are addressed simultaneously
- Economy of financial resource as postage is cheaper than travelling to subjects
- Analysis of data from closed-ended is fairly easy
- Data gathering is not influenced by personal attributes
- Respondents are not usually stampeded in providing responses
- Respondents are guaranteed anonymity and are freer to give information without fear of reprisals

3.0.2.1 Disadvantages of Questionnaires

- They provide no room to clarify ambiguities and rephrase questions. This may result in nil returns which present analysis problems
- They may suffer from a delayed return rate
- They may suffer from a low rate of return
- Their construction is time consuming
- They provide no room to verify authenticity of information given

To carter for the disadvantages of each technique I have used the concept of triangulation which is to use both techniques so as they counter each other’s disadvantages.

3.1 DATA ANALYSIS

The analysis phase was a crucial activity and the goal was to provide a clear understanding of how the current system works, and its shortfalls. In this chapter, efforts were made to outline possibly all the requirements that the proposed system had to meet. Of concern was the analysis of existing systems, diagrammatic representation, strengths and weaknesses and come up with possible alternatives. The study helped to produce a system that addressed the shortcomings of those previous existing system.

3.2 DESCRIPTION OF THE SYSTEM

3.2.0 Users

Two types of users that are involved in this proposed system are the driver or the owner and the passenger.

Passengers:

A passenger is anyone who doesn’t own a car and wants to join a rider in a ride.

Driver:

A driver or owner is one who has the car and is looking for people to share a ride with. The driver can create a new ride to be displayed on the system for the users to see. The application will prompt the information of the ride which consists of destination, origin, meeting, departure Time/date, estimated arrival time and travelling preferences. If the passenger meets the requirements needed by the user then the driver can accept his or her as a passenger.

3.2.1 Find ride and choose a ride

When a passenger needs to find their way to a destination, they can use a search form to request it using destination, origin, date/time of departure. It can also specify travel preferences. When he or she finds a suitable trip, he or she can easily book a seat and the app will send a notification to the rider which the passenger booked. Profile registration System users have their own profiles where they save a description and have important information. Available information is description provided by the user, name, photo to be recognized, workplace or educational institution the user and vehicle information is entered if it is a driver. At the same time, these profiles, depending on what information is available, give other users some validation and trust when it comes to sharing trip.

3.2.3 Notification

When important actions occur, users will receive notification alerts for those, for example, that their status is in the lift has been changed, either because its request to join the lift has been accepted or rejected, or the elevator itself was cancelled. This is important because smartphone users do not have time to check for updates to all their apps, instead the app needs to notify the user if such updates exist, otherwise they will go unnoticed.

3.3 DATA ANALYSIS

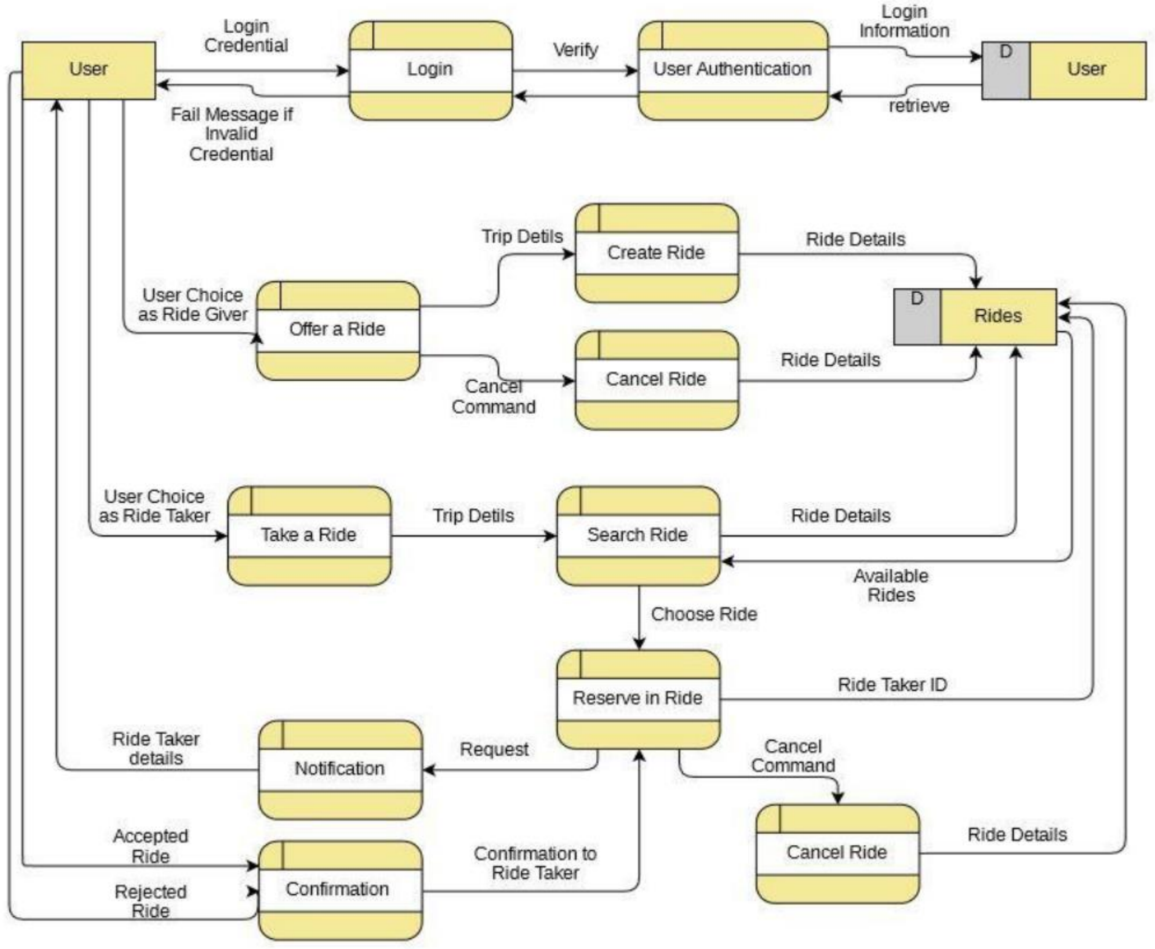


Figure 1: Dfd for existing system)Level 0 and 1)

<u>DFD level 0 of an existing system</u>

Figure shows that admin, owner and user must register to access the system. All the users which are owner and user can manage their detail and all the information will be stored in user and owner table respectively. Owner can manage their car and enter the details of the car into the system and the details will be stored in data store car. When user book the car, then every booking detail will be stored in booking data store and it will display to the owner about the booking. Then, admin can view report from every car detail, booking detail and detail of user and owner

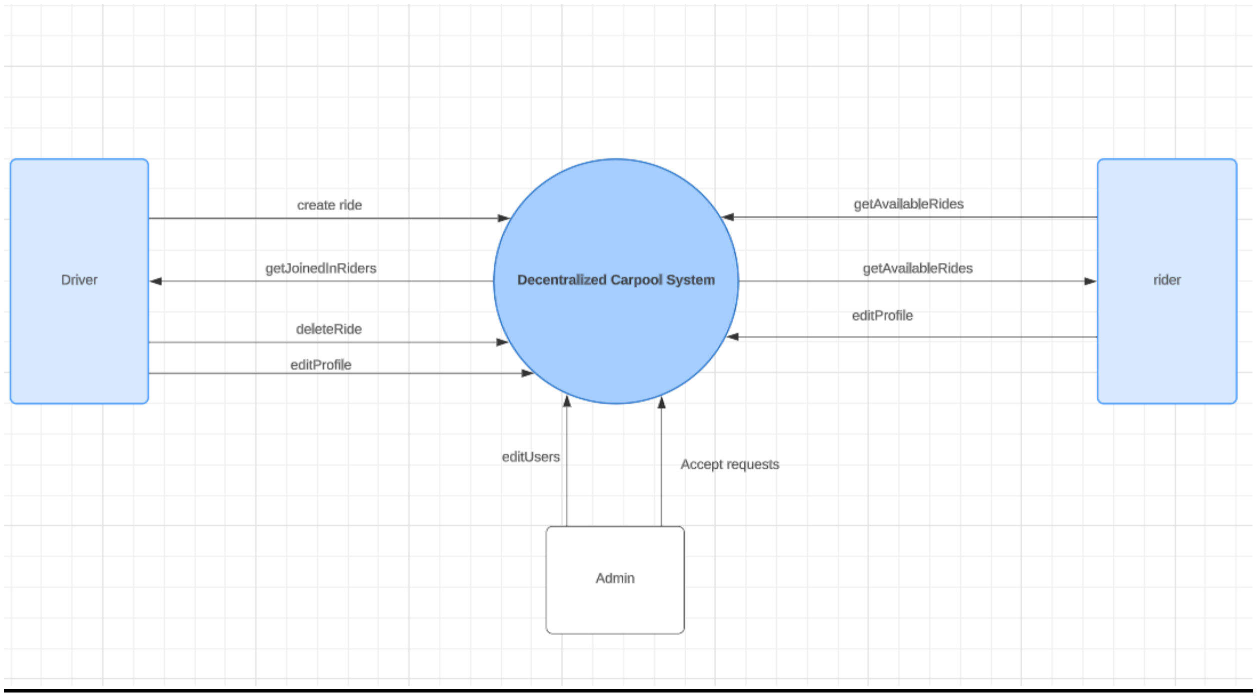


Figure 2: DFD for existing system (Level 0)

<u>**DFD LEVEL ONE**</u>

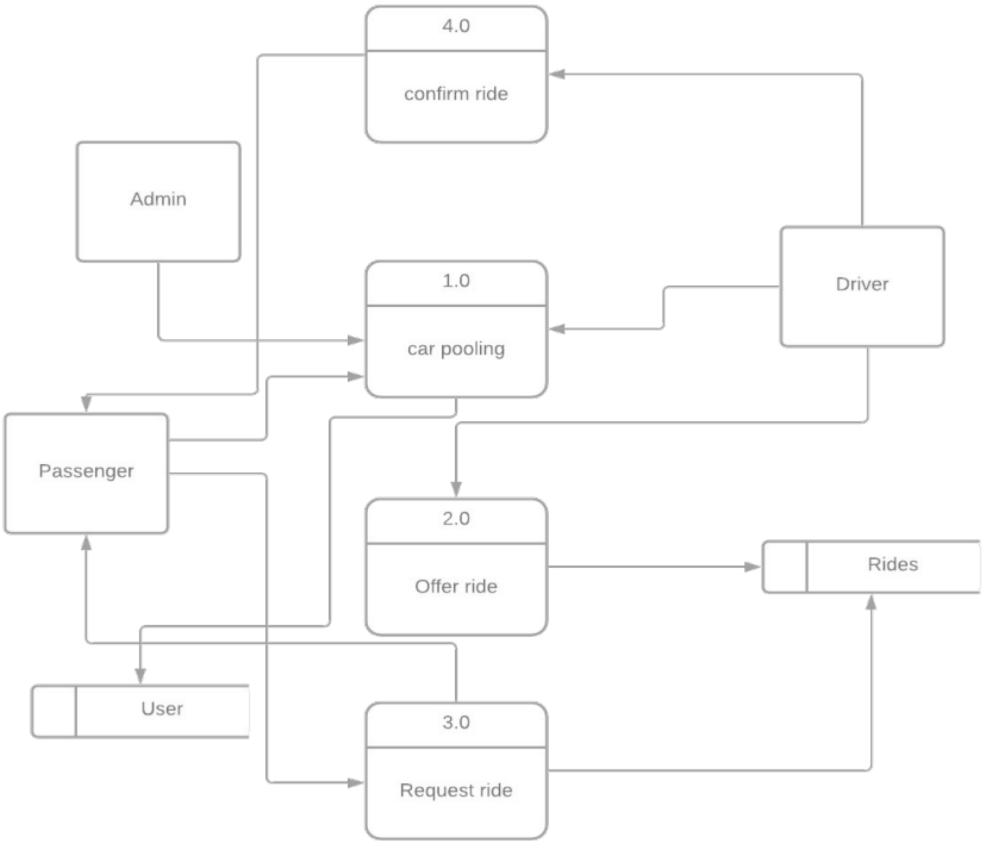


Figure 3: DFD for existing system (Level 1)

Use case diagram for existing system

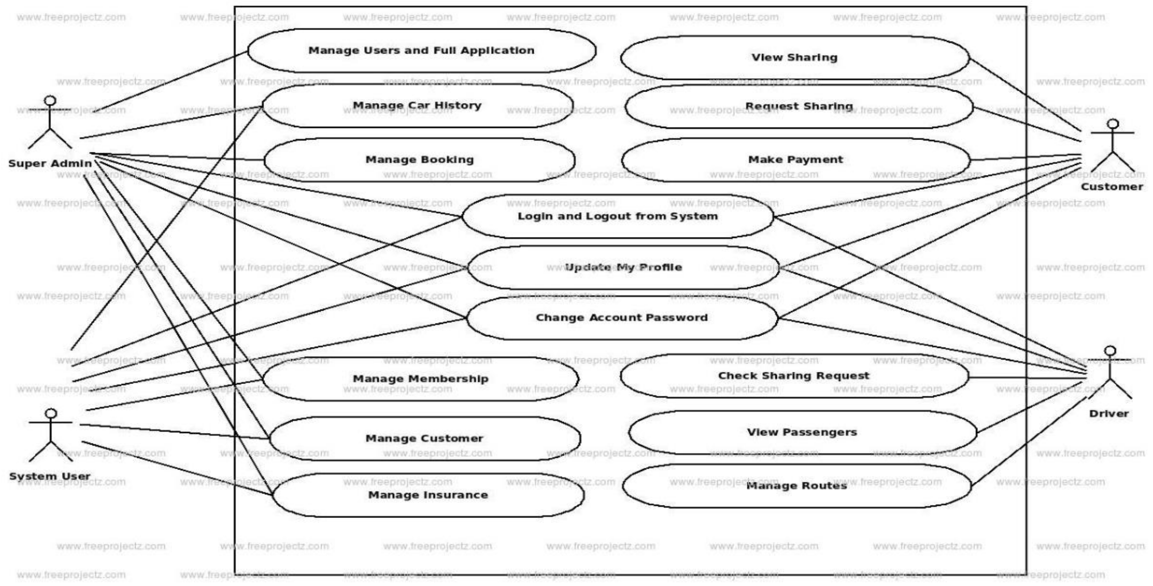


Figure 4: Use case for existing system

Activity Diagram for existing system

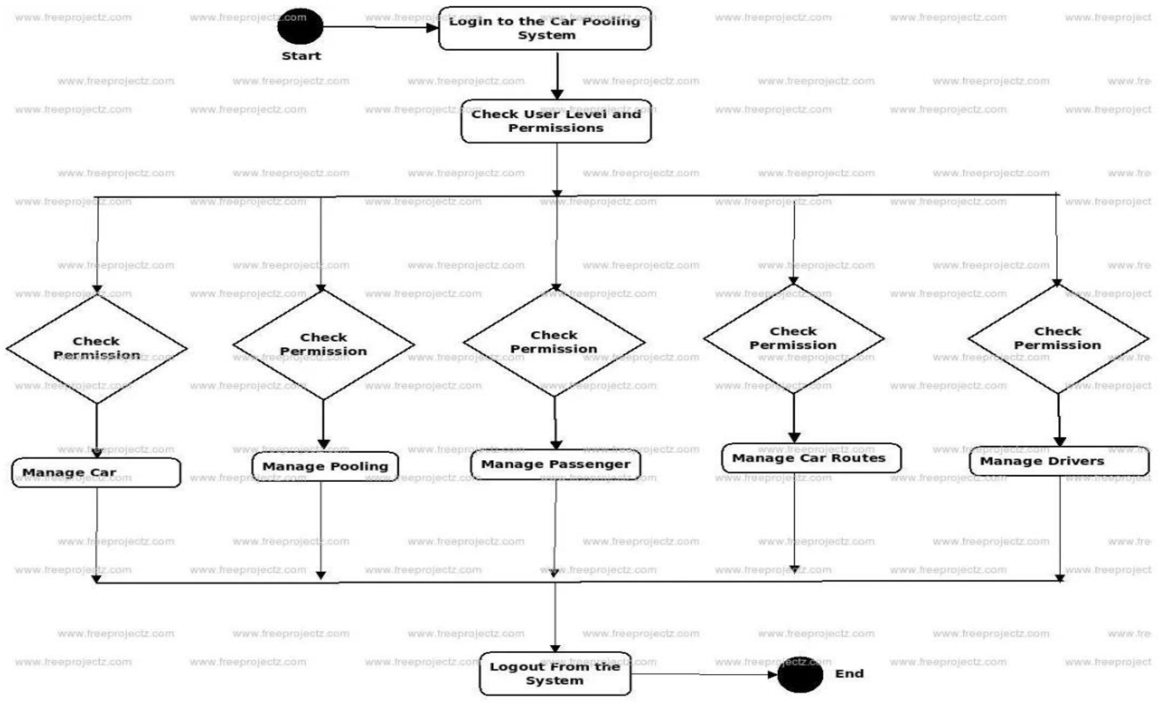


Figure 5: Activity Diagram for existing system

3.4 PROBLEM OF THE EXISTING

- The current carpooling can be hacked and users' data can fall into wrong hands
- The Current system is prone to DDoS attacks
- The taxi system is expensive since a single can hire the taxi
- The manual system of "mushika shika" is illegal.
- It leads to many detour rides.

3.5 FUNCTIONAL ANALYSIS OF THE PROPOSED SYSTEM

3.5.1 Functional Requirements

All users can:

• Create an account.

• Login.

• Request for a ride

-After identification drivers can:

• Submit a ride with specifications.

• Cancel a ride while notifying passengers.

• Accept or Decline a ride request from a passenger.

In addition passengers can:

• Search for a ride.

• View available rides

• Request a ride.

3.5.2 Non-Functional Requirements

a) 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

b) 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.

c) Extensibility

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

d) Availability

   Since the application is decentralized, it has to be highly available and guarantees a good server up-time. The application guarantees 99.999% availability.

e) Privacy and security

   The application should ensure the privacy of the users meaning no one can be able to alter their information. The login system should also be robust where only authorized users can access the application's functionality.

f) Maintainability

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

3.6 Use case description of requesting pooling

<u>Introduction</u>

This use case documents the steps that must be followed in order for user to requesting a pooling that is to request a ride in the carpooling system.

Actors

Passenger

Pre-Condition

The actor must have successfully logged in and must have a token

Post-Condition

If the use case is successful, the actor can view the rides searched, if not the system remains unchanged

Basic flow

System checks if the actor is authenticated and have some tokens

The actor navigate to find ride page

The system prompts user to enter trip details

The actor enter user details

System searches using user details

Alternative flow

If the actor is unauthenticated he or she is redirected to the login page

If actor doesn't have tokens he or she is redirected to purchase tokens page

If an error occurs the system displays error message

Special requirement

Internet Connection

Associated use case

None

CHAPTER 4

DESIGN

4.0 INTRODUCTION

This chapter will highlight the design and methodology of the proposed solution, i.e. the development, configuration and deployment platform. The chapter will cover the Unified Modeling Language (UML), which is a collection of notations used to document specifications and design. Systems architecture and system diagrams such as UML activity diagram, UML class diagram, UML sequence diagram and UML deployment diagram will be used to achieve this goal.

4.1 System Diagrams

A system diagram is a diagram that defines the boundary between the system, or part of a system, and its environment, showing entities that interact with it. These diagrams help to get to an agreement on the scope of the project hence they are used in the early stages of a project life cycle. System diagrams are helpful in showing how a single change may affect the whole system. They are models used to express dynamic forces acting upon the components of a system and their interactions.

4.1.0 Data Flow Diagram

Data flow diagrams maps out the movement (flow) of information in a system. It uses defined symbols like rectangle, circles, arrows and short text.

- Circles represent processes
- Rectangles represent external entities
- Arrows show the movement of information
- Short text describe the nature of information being moved

<u>Context diagram of proposed system</u>

A context diagram is the most basic data flow diagram that provides an overview of how data flows in the system easily but with less detail. A context diagram can also be called a level-zero data flow diagram

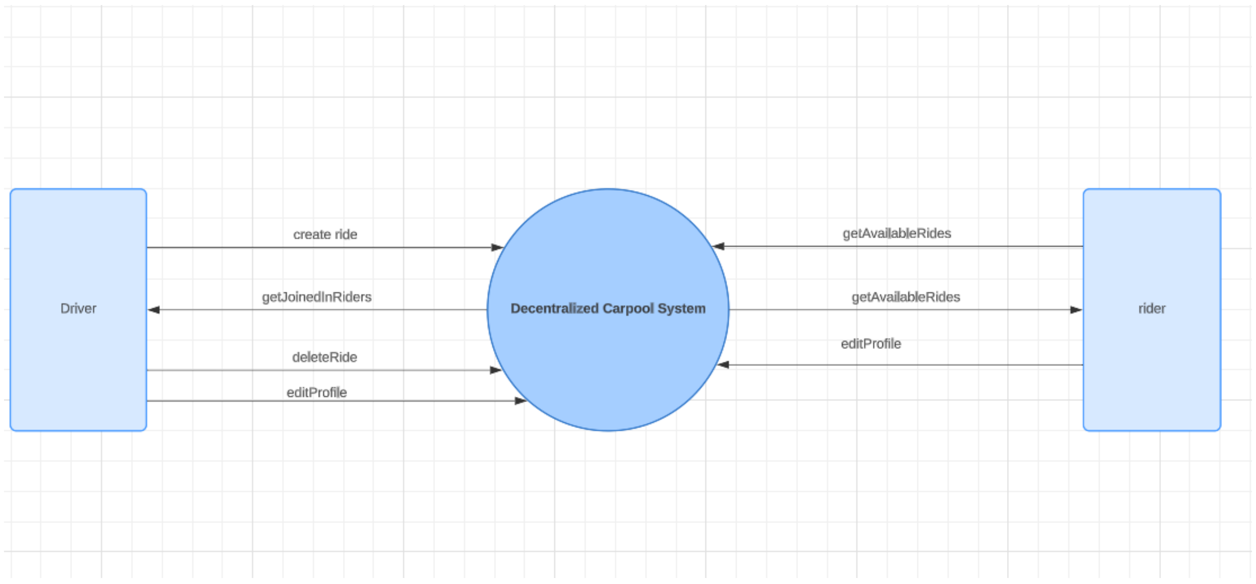


Figure 6: DFD for proposed system (Level 0)

Level-One data flow diagram of proposed system

Level one data flow diagram provides much more details on the processes of the system. It breaks down the main processes into smaller sub processes that can be analyzed and improved on a deeper level

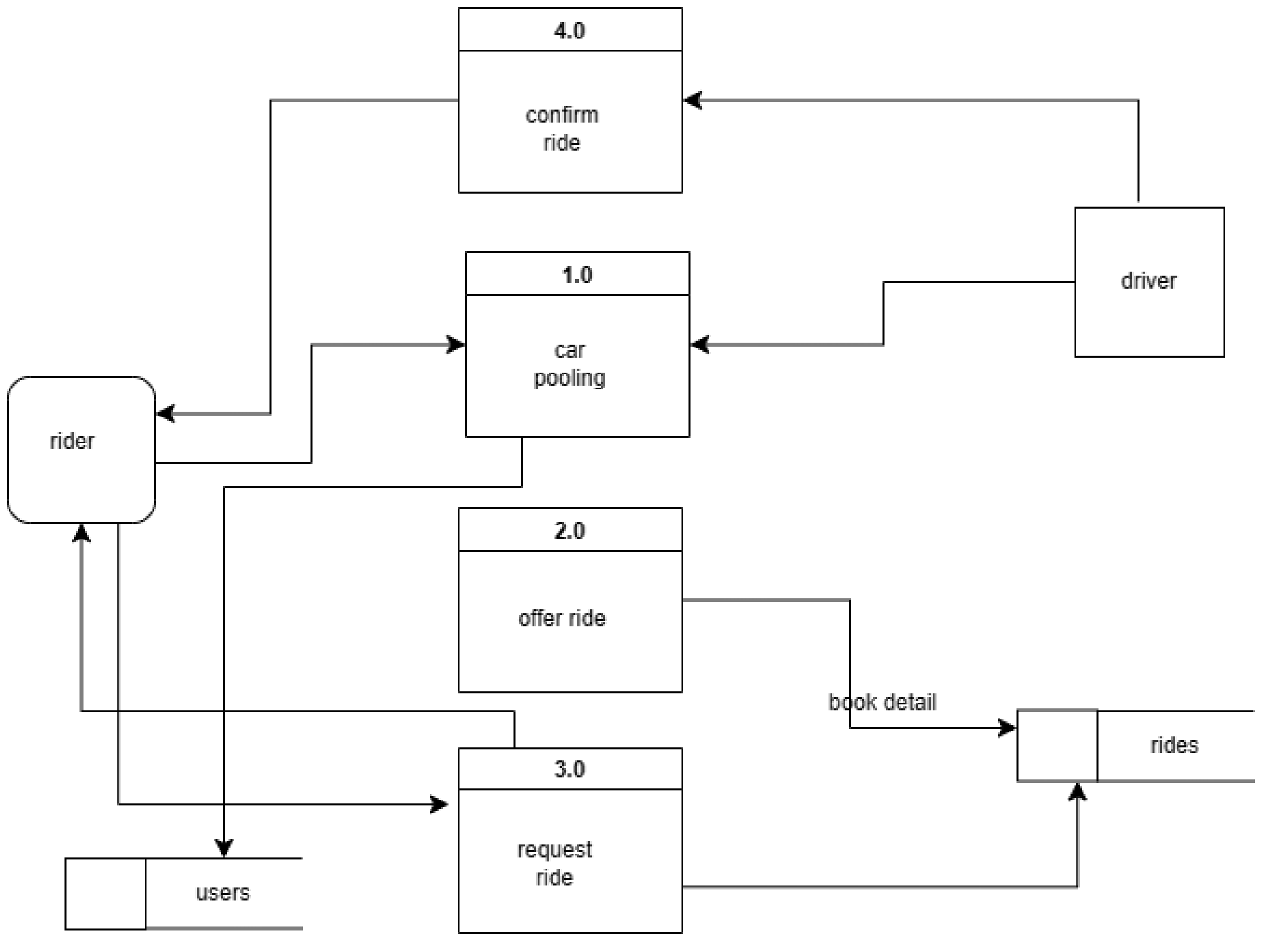


Figure 7: DFD for proposed system (Level 1)

4.1.1 Activity diagram

An activity diagram is a graphical notation of flow chart which shows flow of one activity to another. Below activity diagram shows the activities, associations and conditions involved in information inquiry

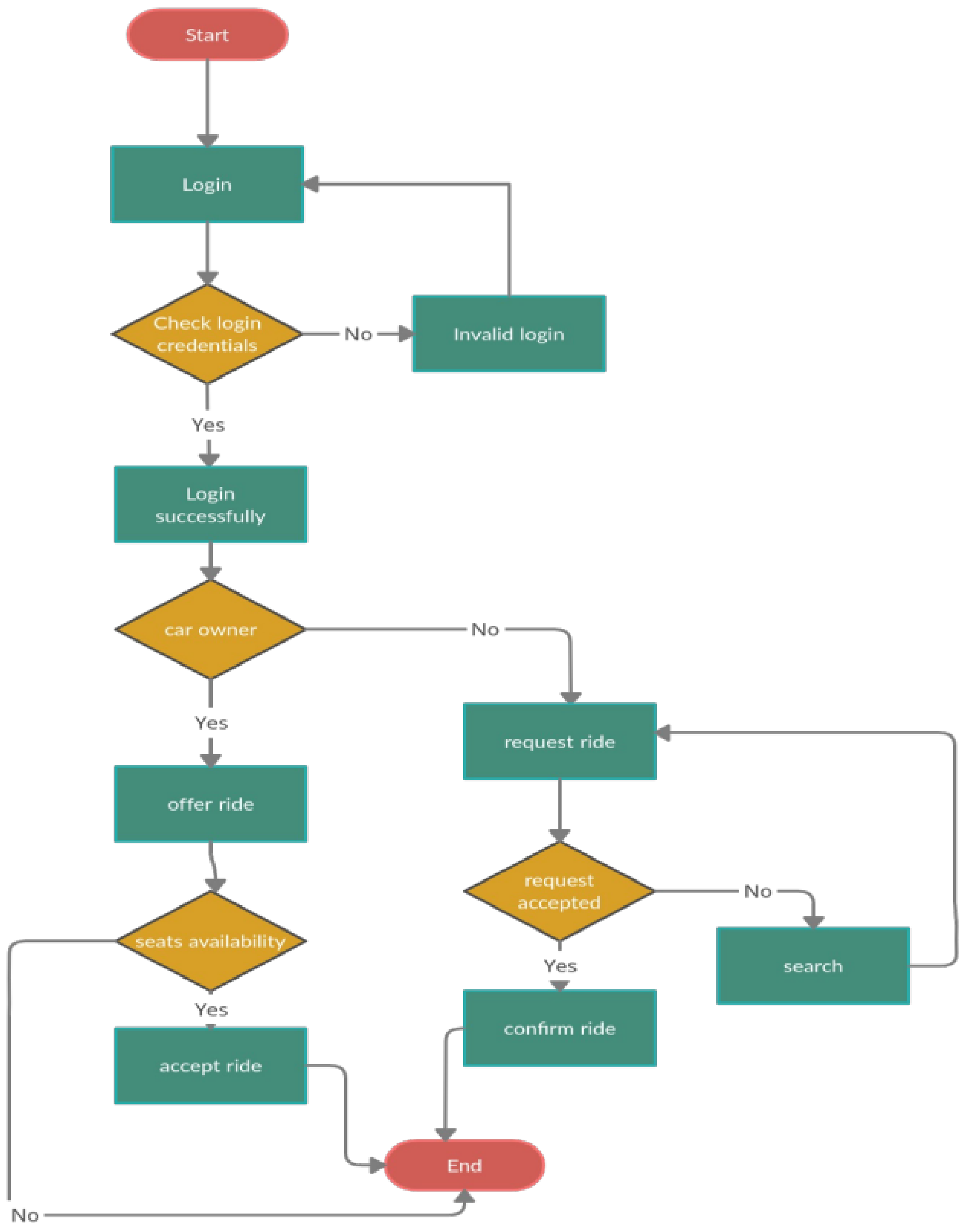


Figure 8: Activity Diagram for proposed system

4.1.2 Architectural Design

The following are the components of the decentralized carpool system.

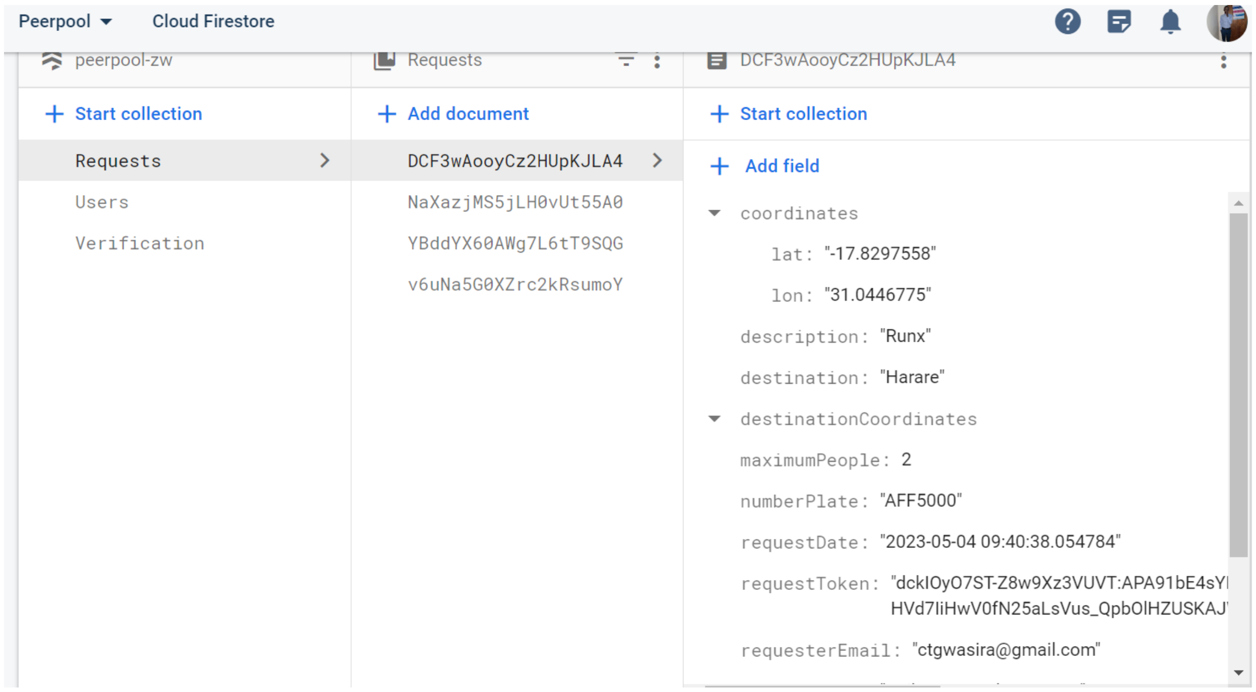
Users:

Two types of users are involved in the proposed system: driver and the rider. Rider is a person requesting carpooling services and driver the person providing carpooling services using his own car. Typically, drivers and riders are going in the same direction and are all in good books with the idea of sharing a ride. Decentralized Carpooling System:

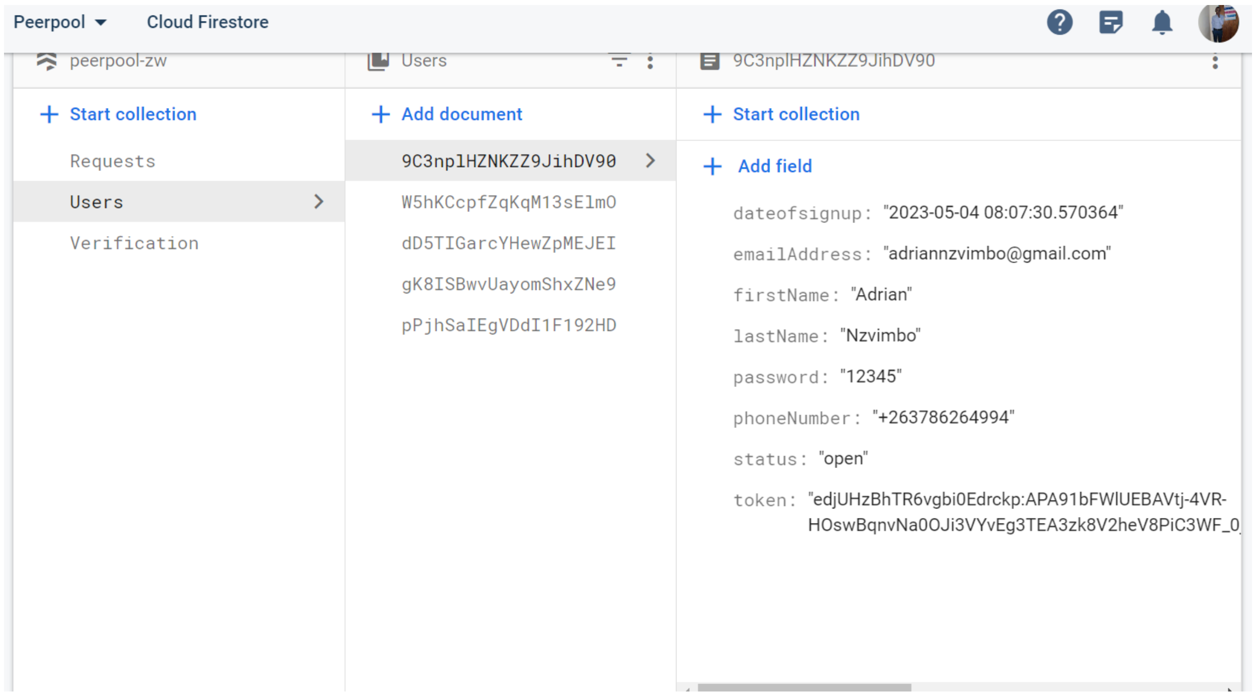
A decentralized android mobile application which is the most important component of the application system. It includes the front end which the users interact with and databases and business logic. For backend firebase was used as it uses real time database technology. n. Firebase is based on the JavaScript Object Notation (JSON) principle, in which data is saved as a node with a reference key. Furthermore, NoSQL is used for the suggested decentralized carpooling application since it gives a more flexible supported manner for scalable database than standard relational SQL.The system also uses Google Maps API to get a user’s current location and also used Google Maps autocomplete for easy location selection.

4.1.2.0 Database Design

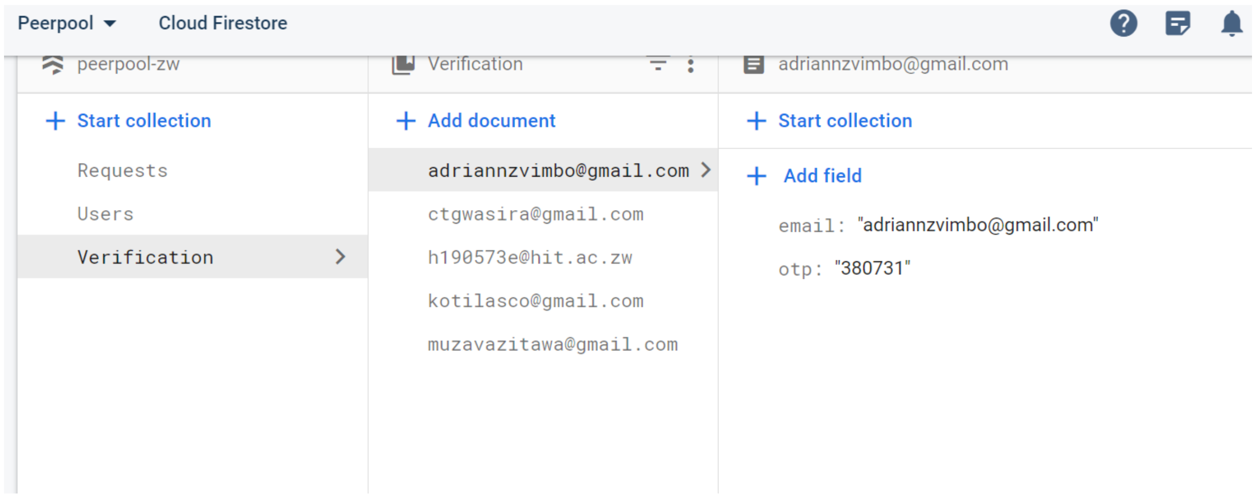
The system uses the Firestone database from firebase as mentioned at the beginning of this document. It is a NoSql database. My database has collections and these collections have documents whose data is stored in fields. Normalization was done to reduce information reputation and data redundancy. This technique has non-repetitive and reliable information, reducing information conflicts. The database is decentralized.



```
coordinates
    lat: "-17.8297558"
    lon: "31.0446775"
description: "Runx"
destination: "Harare"
destinationCoordinates
maximumPeople: 2
numberPlate: "AFF5000"
requestDate: "2023-05-04 09:40:38.054784"
requestToken: "dckIOyO7ST-Z8w9Xz3VUVT:APA91bE4sYI
              HVd7IiHwV0fN25aLsVus_QpbOlHZUSKAJ'
requesterEmail: "ctgwasira@gmail.com"
```



```
dateofsignup: "2023-05-04 08:07:30.570364"
emailAddress: "adriannzvimbo@gmail.com"
firstName: "Adrian"
lastName: "Nzvimbo"
password: "12345"
phoneNumber: "+263786264994"
status: "open"
token: "edjUHzBhTR6vgbi0Edrckp:APA91bFWlUEBAVtj-4VR-
       HOswBqnvNa0OJi3VYvEg3TEA3zk8V2heV8PiC3WF_0
```



Peerpool Cloud Firestore

peerpool-zw | Verification | adriannzvimbo@gmail.com

Start collection | Add document | Start collection

Requests
Users
Verification

adriannzvimbo@gmail.com
ctgwasira@gmail.com
h190573e@hit.ac.zw
kotilasco@gmail.com
muzavazitawa@gmail.com

Add field

email: "adriannzvimbo@gmail.com"
otp: "380731"

Figure 9: Firestore Database for the system

4.1.3 Entity Relationship Diagram.

An organizational relationship diagram shows the entities included in the system, their descriptio ns, and relationships as an easy way to build a conceptual model.

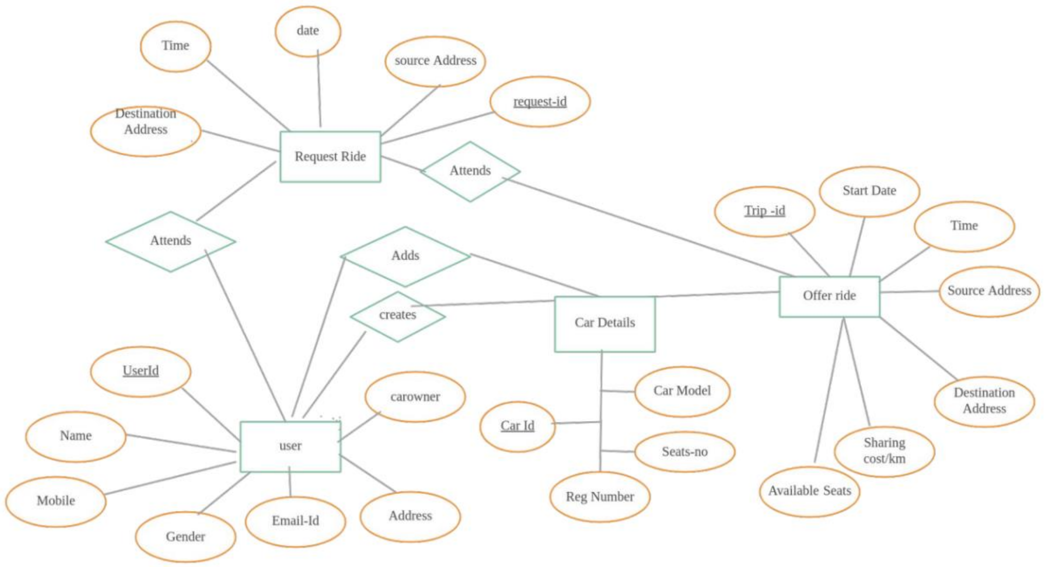


Figure 10: ERD for the system

4.1.4 PROGRAM DESIGN

Methodology:

Steps of the proposed system:

a) The driver logs on to the system to pick the start-up data and time, pickup spot which is his or her location, destination, car details and seat availability.
b) The passenger makes a request to the ride he or she wants
c) A confirmation is sent to the driver if a request is made through notifications
d) The number of available seats decreases as the request are made and if the seats are now full then no further request can be made

The Haversine formula is used to calculate the distance between the driver and the rider.

The Haversine formula is used to calculate the great-circle distance between two points on a sphere (such as the Earth). The formula is:

d = 2 * r * arcsin(sqrt(sin^2((lat2 - lat1)/2) + cos(lat1) * cos(lat2) * sin^2((lon2 - lon1)/2)))

where:

- d is the distance between the two points
- r is the radius of the sphere (in the case of the Earth, this is typically taken to be 6,371 kilometers or 3,959 miles)
- lat1, lon1 are the latitude and longitude, respectively, of the first point
- lat2, lon2 are the latitude and longitude, respectively, of the second point

The latitude and longitude values should be in radians. If they are in degrees, they need to be converted to radians first by multiplying by pi/180.

Note that this formula assumes that the Earth is a perfect sphere, which is not entirely accurate.

4.1.5 Package diagram

A package diagram is a type of UML diagram that shows the dependencies between software packages or modules

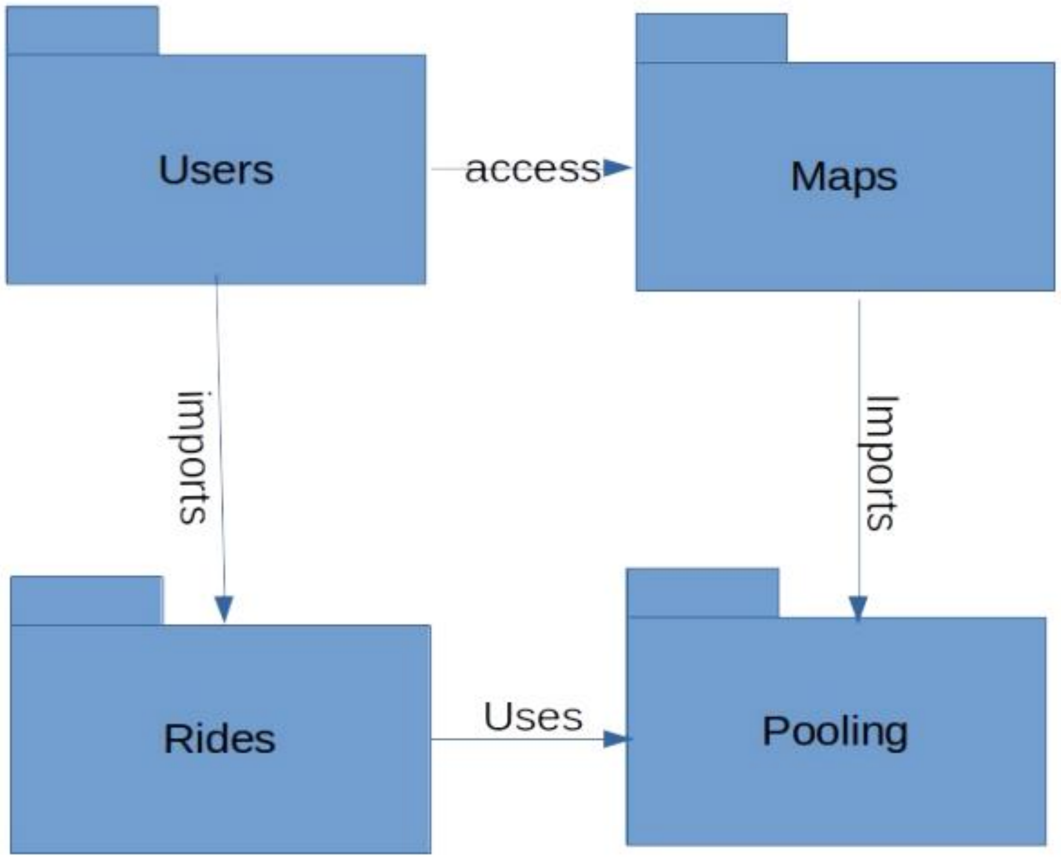


Figure 11: Package Diagram for the system

4.1.6 Class Diagram

The following is the class diagram for the system, demonstrating the main classes and their methods. It demonstrates the analysis of the static view of the system.

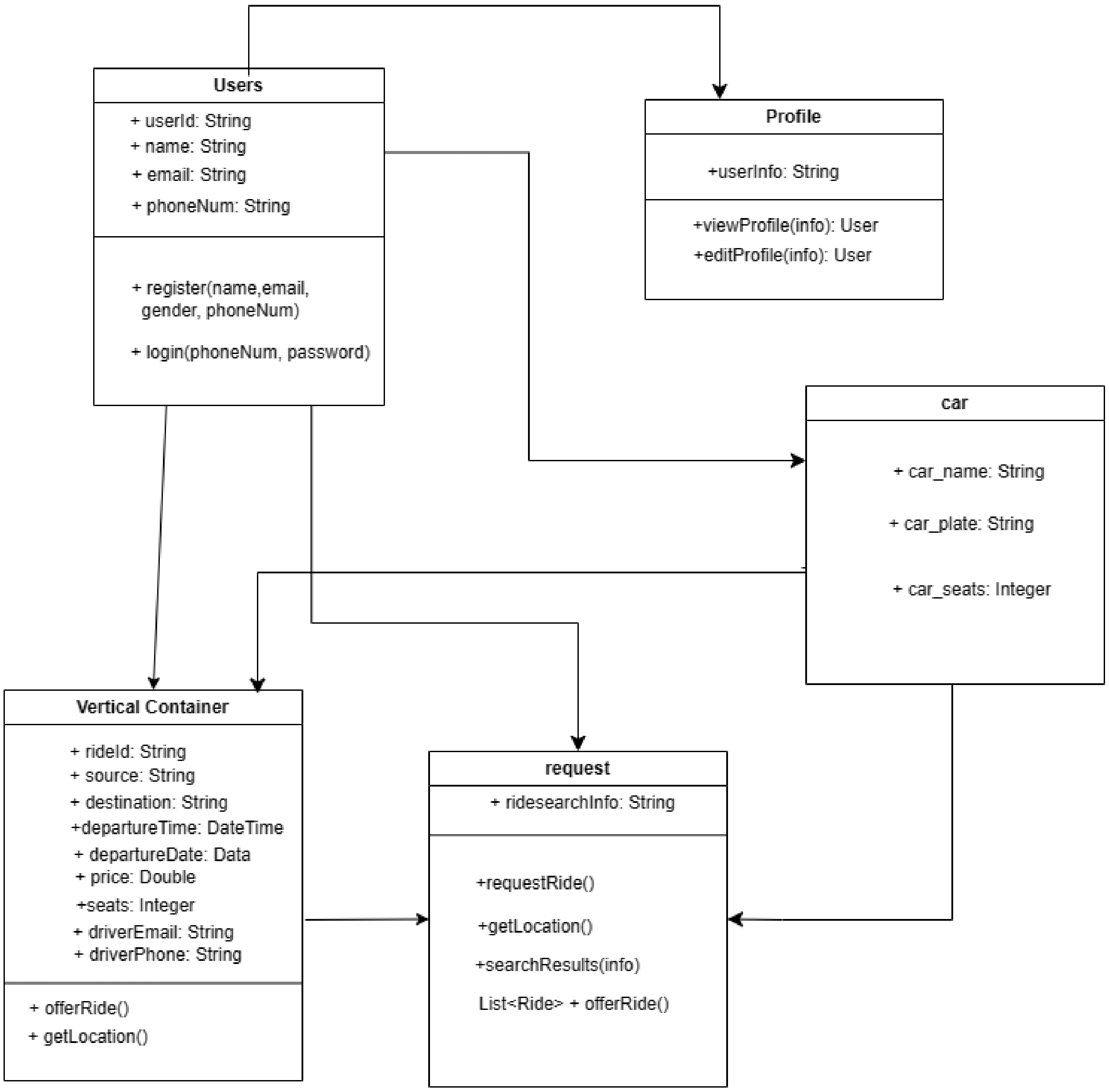


Figure 12: Class Diagram for the system

4.1.7 Sequence diagram

A sequence diagram is a type of UML diagram that represents the interactions between objects or components in a system or process. It shows the flow of messages or events between objects or components in a chronological order, typically from top to bottom. Sequence diagrams are useful for understanding the interactions between objects or components in a system or process and can be used to design, analyze and document systems or processes.

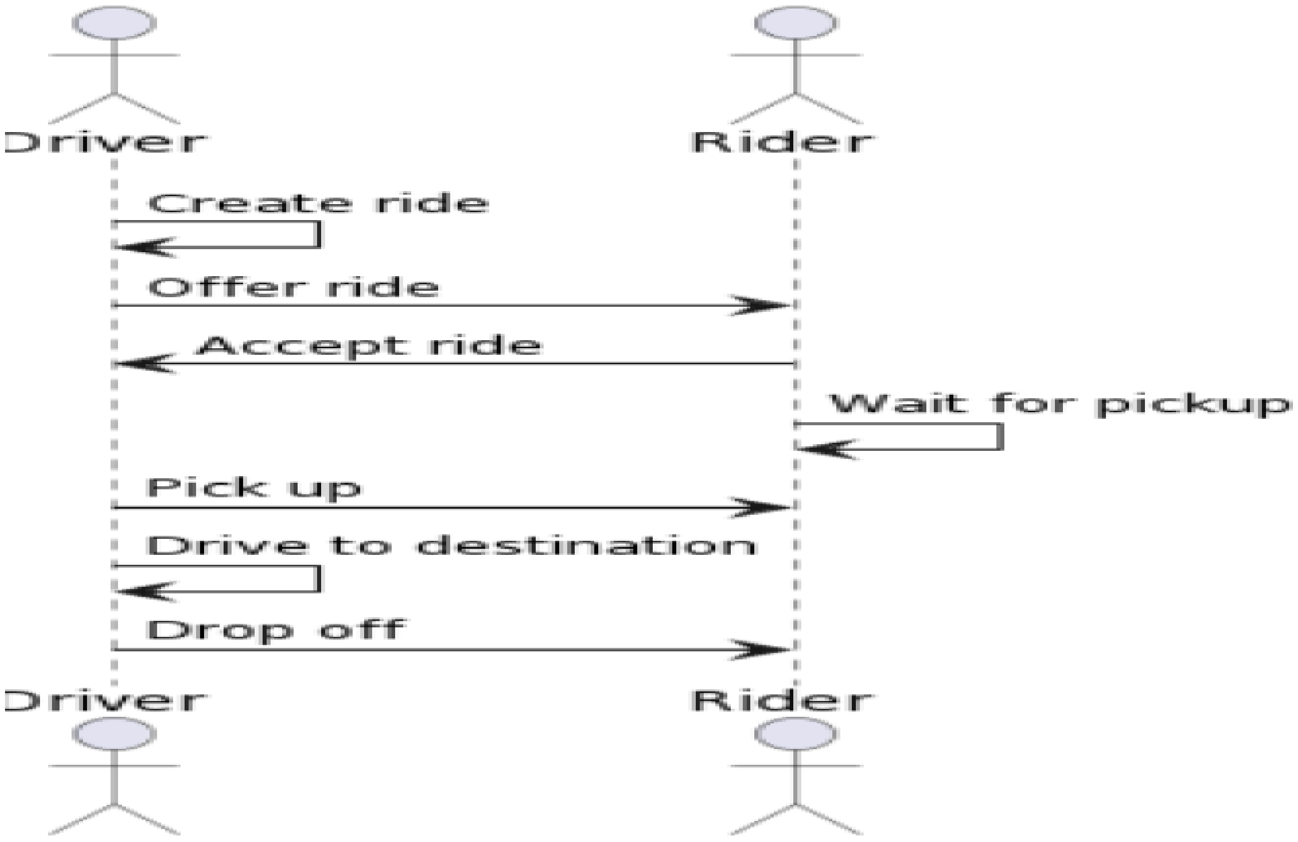
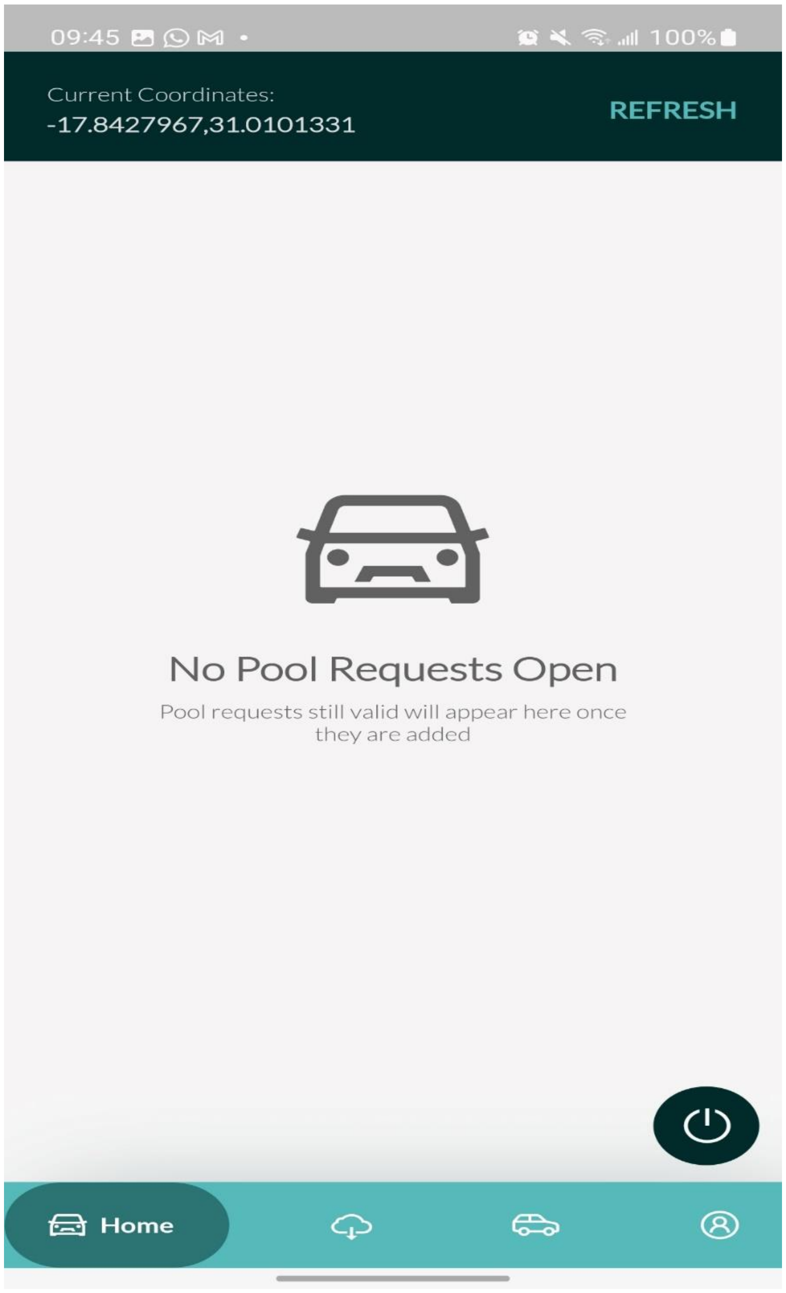


Figure 13: Sequence Diagram for the system

4.2 INTERFACE DESIGN

4.2.0 Home page

It contains main navigation system of the application and the menu to accessory pages of the application. It contains all the rides in the system.



Current Coordinates:
-17.8427967,31.0101331

REFRESH

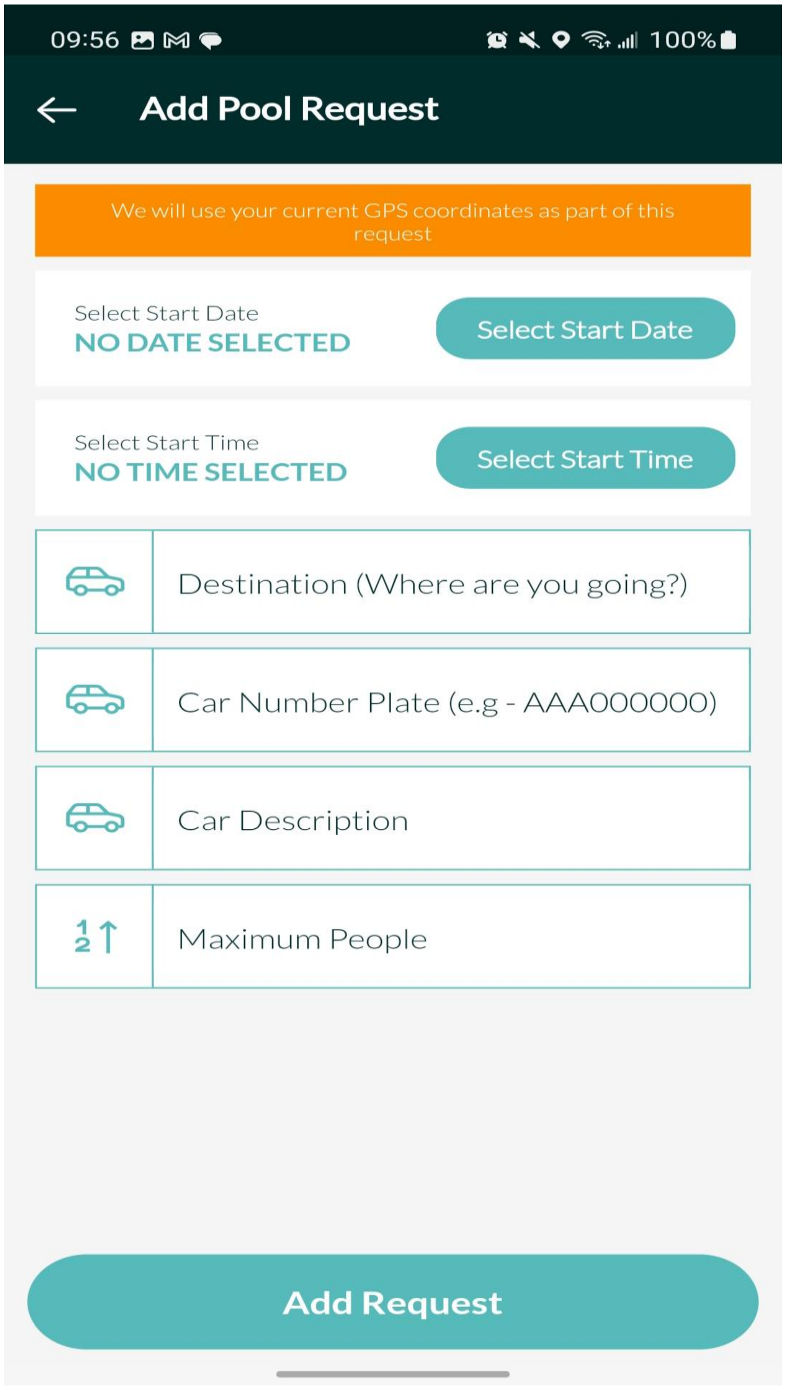
No Pool Requests Open

Pool requests still valid will appear here once they are added

Figure 14: Home Screen

4.2.1 Offer ride page

This the page the driver inputs his ride details and are saved to firestore database


09:56 100%

Add Pool Request

We will use your current GPS coordinates as part of this request

Select Start Date
NO DATE SELECTED

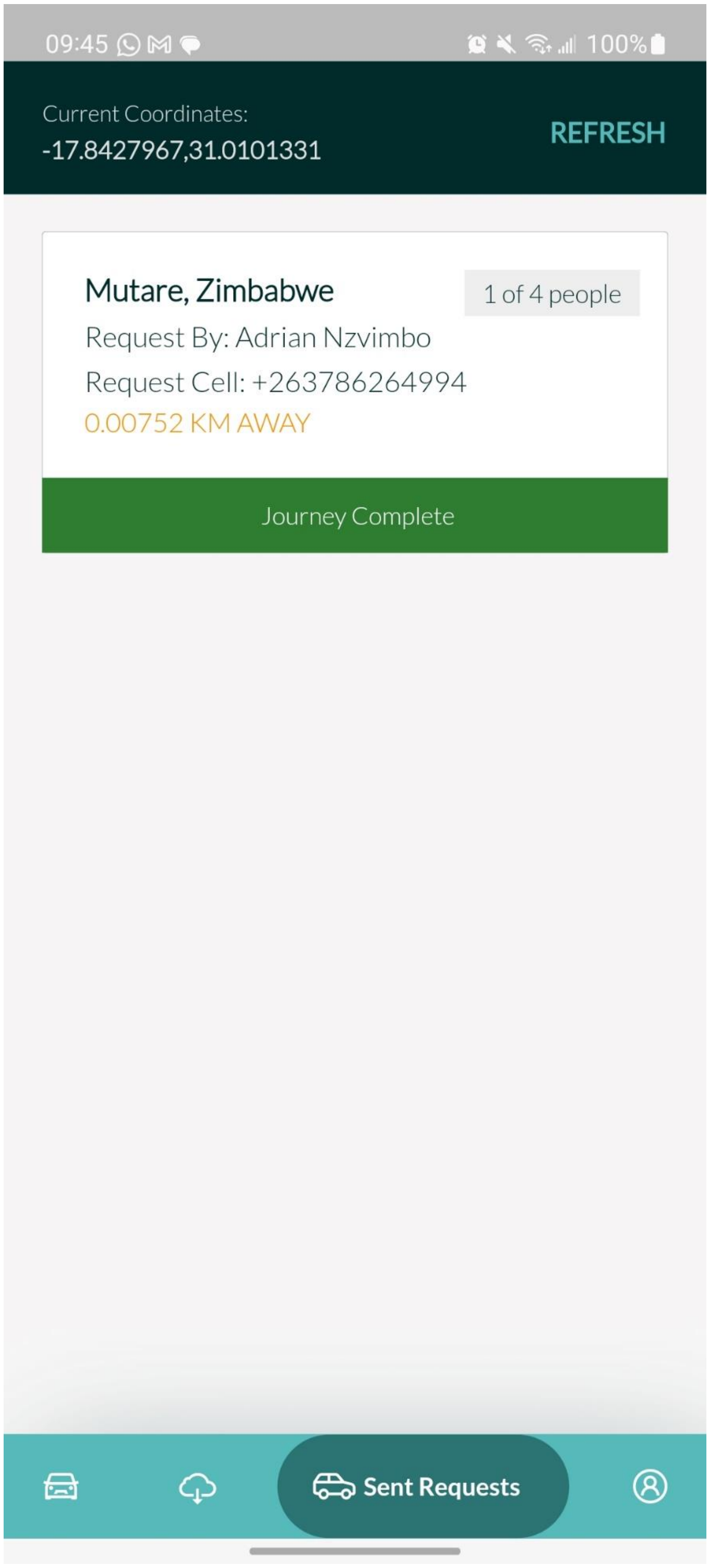
Select Start Time
NO TIME SELECTED

Select Start Time
Destination (Where are you going?)
Car Number Plate (e.g - AAA000000)
Car Description
Maximum People
Add Request

Figure 15: Offer Screen

4.2.2 Request ride page

This is the page where the rider uses to book a ride.


09:45 100%

Current Coordinates:
-17.8427967,31.0101331

REFRESH

Mutare, Zimbabwe

1 of 4 people

Request By: Adrian Nzvimbo

Request Cell: +263786264994

0.00752 KM AWAY

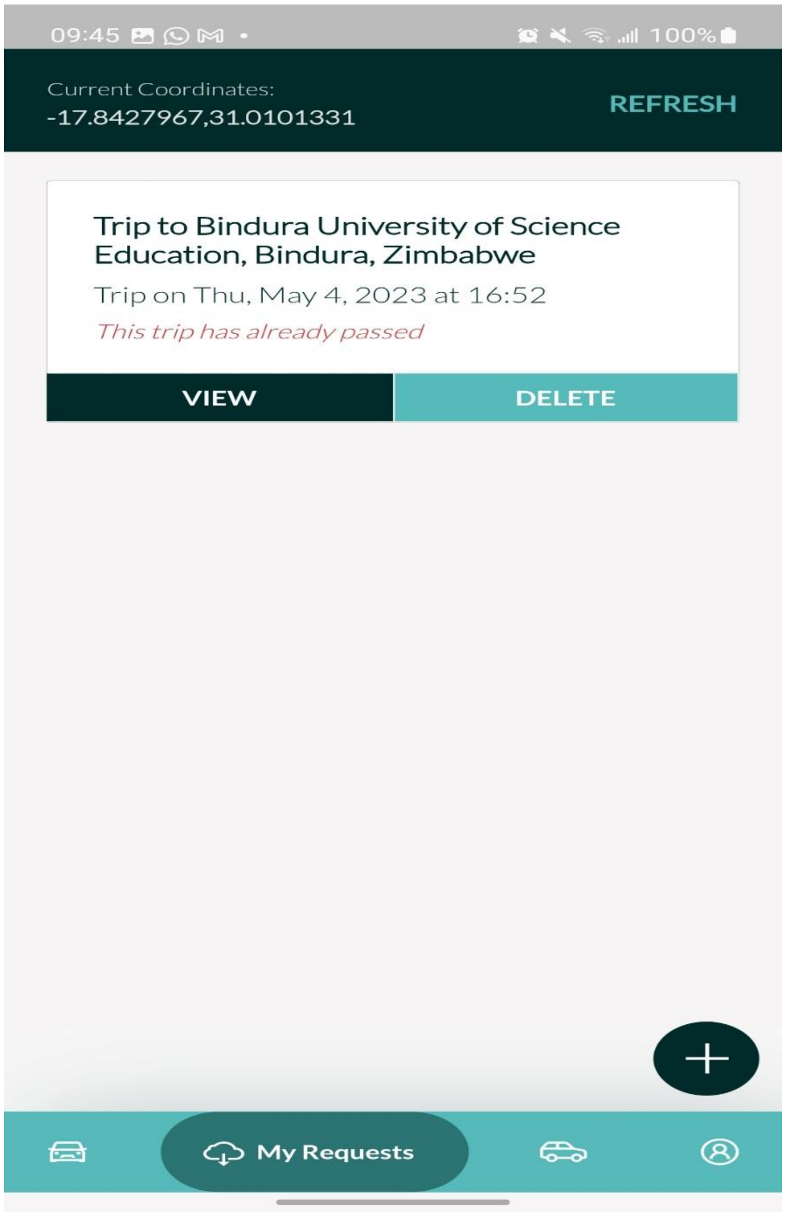
Journey Complete

Figure 16: Offer Screen

Figure 16: Request Ride Screen

4.2.3 Driver Page

This shows all the rides a driver has created and the riders who have requested his ride.



Trip to Bindura University of Science Education, Bindura, Zimbabwe

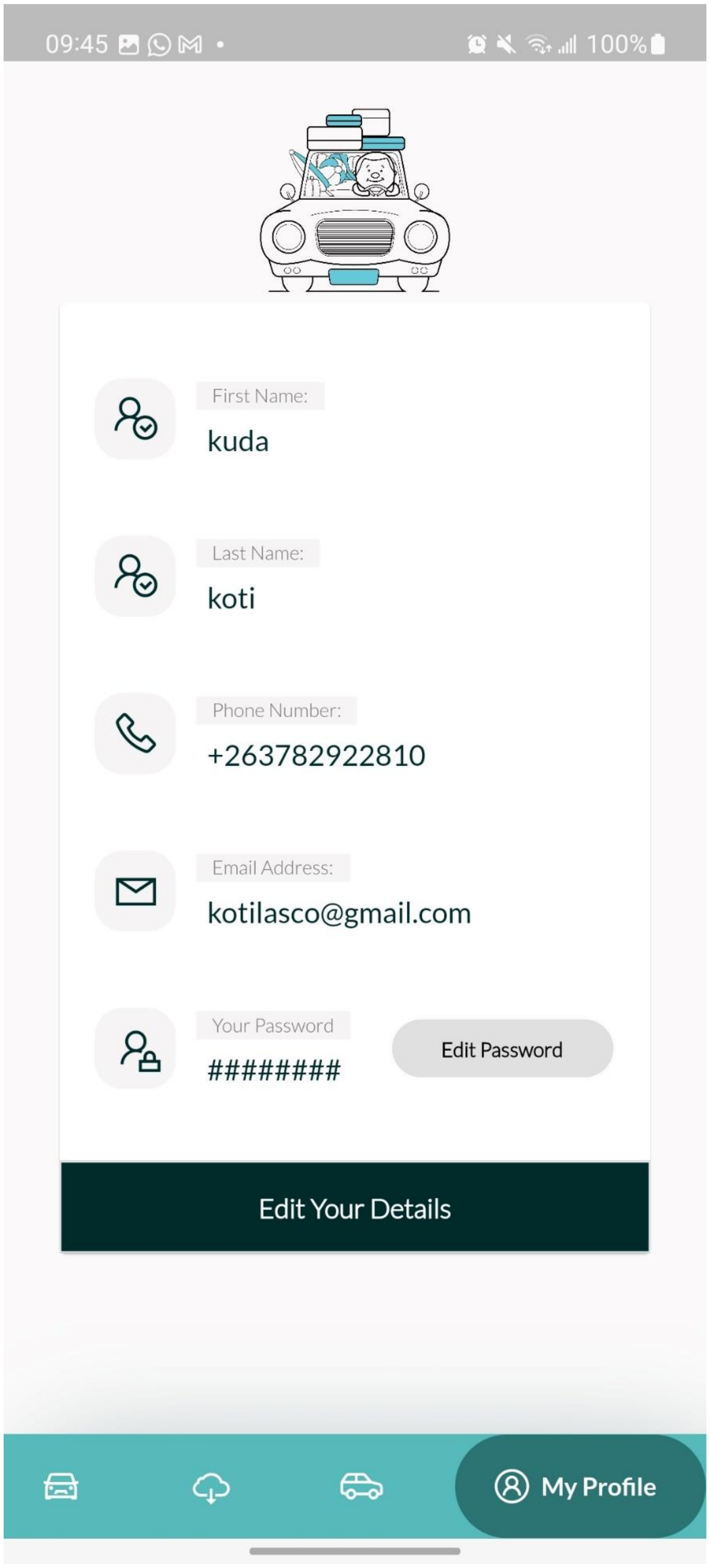
Trip on Thu, May 4, 2023 at 16:52

This trip has already passed

Figure 17: My Request Screen

4.2.4 Profile page

Shows profile page of the user and can use update button to update the profiler and user details


09:45 100%

First Name:
kuda

Last Name:
koti

Phone Number:
+263782922810

Email Address:
kotilasco@gmail.com

Your Password
########

Figure 18: Profile Screen

4.3 Conclusion

UML class diagrams and project specifications are useful when modelling data. By accurately modelling attributes and associations of class entities, we can easily map these class diagram specifications to better understand them. Furthermore it’s hard to map all the project’s features before doing all the planning of the features because some changes may occur during the project's development some features will be added depending on the situation and availability but these specifications and modelling are the core of the application and it will be made with these specifications as the core design. Modelling Data is a very important step of the project realization and by turning our idea into a design concept we are going to make the implementation more easy and organized. After designing our project we will turn the concept into real functions which would be the implementation to an android application

Chapter 5

Implementation and Testing

5.0 Introduction

In this chapter the implementation and evaluation process conducted in decentralized carpooling system is outlined it gives more insight on how the system was built. In terms of implementation the development environment, tools, development platform, algorithms and the results of the system testing are discussed. This phase will also check if the developed system is in line with the stated objectives and will give walkthrough of programming tools and aspects using code snippets. For testing, the type of testing procedure used, participants and analysis made on the results for testing procedure are explained. This chapter covers more on various issues related to programming concepts, rules followed and testing which is the most critical part in implementation.

5.1 Coding styles and conventions

These are rules contributing to writing of flutter code for android application project. The project was developed using flutter and firebase for back end.

- Always specify types
- Using upperCamelCase for naming classes, enums, typedefs and extensions
- Use of null safety features
- Use of comments
- Use of strict analysis rules

5.2 Coding review

Systematic study of the source code in order to detect defects and evaluate the code's quality known as code review. Code review also helps in maintaining consistency in system design and execution. The code created during the implementation of the project was reviewed to correct errors that were ignored during the original development phase. This review was done by several of them my co-workers.

Over-the-shoulder is the approach used for code review. It is among the simplest and most natural ways to participate in peer code review. After the system was completed, there were two qualified colleagues asked to evaluate the code while explaining why it was developed in such a way. Considerations considered during code review are included in the table below along with the initials of the individuals who participated in the review process.

Implementation and assessment of the proposed system was carried out. All objectives were met and all coding rules were followed for the various languages used. The coding technique used involved considerable preparation for most sophisticated modules and trial and error for other secondary components of the system. This was done in order to create a high quality, fully functional system. The code has also been reviewed by a number of individuals to ensure its quality.

5.3 Pseudo code and code snippets

Decentralized carpooling system is a dynamic system which depends on two underneath sources of information: which includes rides offered uploaded by drivers and ride selection and registration by passengers. The user (driver) who is going to travel by his/her vehicle will mention source, destination along with the capacity of vehicle. The user (passenger) who finds the path convenient can register for the trip. This section will cover code snippets of main modules of this process.

5.3.1 Registration and Login

It is used for verifying the user credential to decide whether or not a given user can access the system. The user has to register first creating account if he has an account already he / she should navigate to the login page for authentication which takes phone number and password. The login module uses firebase authentication package for flutter and is used to get access to a valid User Object which will be tested to determine whether it’s a valid user or not. If authentication is successful it calls function that navigates the user to the home screen.

Registration pseudo code

```
If user has no account
        Initialize variables
                Get user input
```

```
            Navigate to the second registration page
            Get user's phone number and password
            Check if user already exist
    else
            Authenticate else Navigate to login page
```

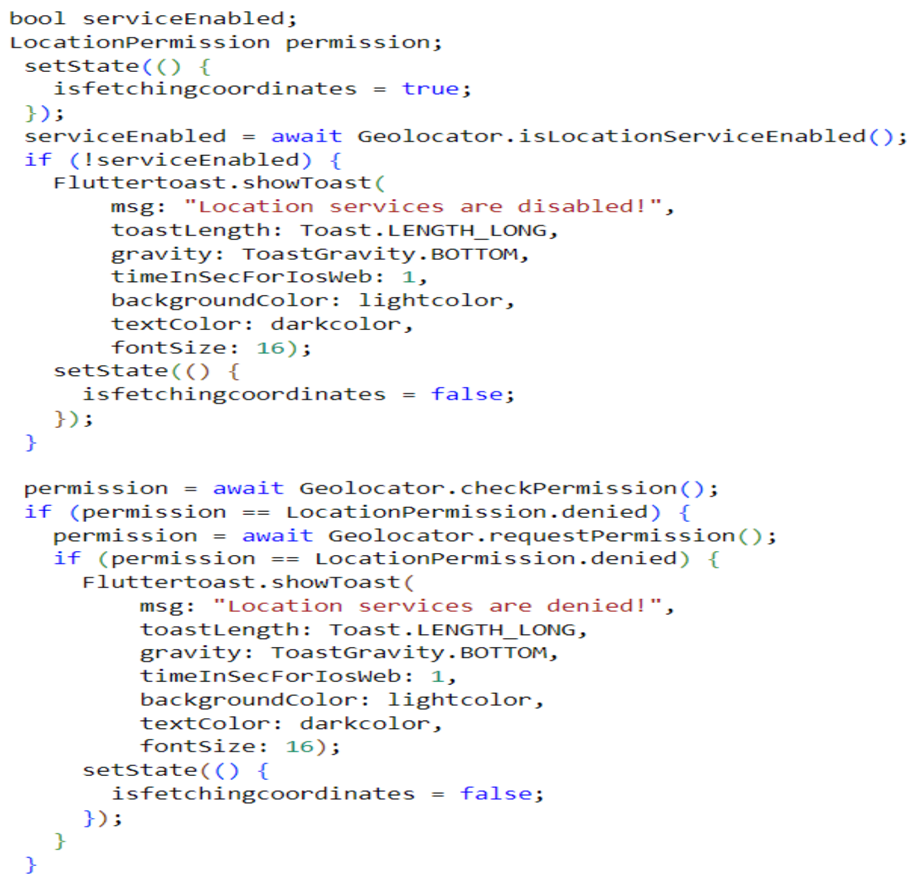
5.3.2 Authentication

```
double screenWidth = returnScreenWidth(context);
var phone = loginnumber.text.trim();
var pass = loginpassword.text.trim();
if (phone.isEmpty || pass.isEmpty) {
  Fluttertoast.showToast(
      msg: "Please do not leave any textfields blank",
      toastLength: Toast.LENGTH_LONG,
      gravity: ToastGravity.BOTTOM,
      timeInSecForIosWeb: 1,
      backgroundColor: lightcolor,
      textColor: darkcolor,
      fontSize: screenWidth * 0.037);
} else {
  setState(() {
    signinloading = true;
  });
  QuerySnapshot<Map<String, dynamic>> snapshot = await FirebaseFirestore
      .instance.collection('Users')
      .where('phoneNumber', isEqualTo: phone)
      .get();
  if (snapshot.docs.isEmpty) {
    Fluttertoast.showToast(
        msg: "User not found",
        toastLength: Toast.LENGTH_LONG,
        gravity: ToastGravity.BOTTOM,
        timeInSecForIosWeb: 1,
        backgroundColor: lightcolor,
        textColor: darkcolor,
        fontSize: screenWidth * 0.037);
    setState(() {
      signinloading = false;
    });
  } else {
    List<QueryDocumentSnapshot> docs = snapshot.docs;
    var data = docs.first.data() as Map<String, dynamic>;
    var id = docs.first.id;
    if (data['status'] == "blocked") {
      Fluttertoast.showToast(
          msg: "Your account is blocked. Please contact the administrator to get it unblocked.",
          toastLength: Toast.LENGTH_LONG,
          gravity: ToastGravity.BOTTOM,
          timeInSecForIosWeb: 1,
          backgroundColor: lightcolor,
          textColor: darkcolor,
          fontSize: screenWidth * 0.037);
      setState(() {
        signinloading = false;
      });
    } else if (data['password'] != null && data['password'] == pass) {
      var body = {...data, "id": id};
      await onGetToken(body);
      await onRemovePreference("peerpool_logged_user");
      await onSetPreference("peerpool_logged_user", json.encode(body));
      setState(() {
        signinloading = false;
      });
      Navigator.push(context, MaterialPageRoute(builder: (context) => const Dashboard()));
    } else {
      Fluttertoast.showToast(
          msg: "Wrong Password",
          toastLength: Toast.LENGTH_LONG,
          gravity: ToastGravity.BOTTOM,
          timeInSecForIosWeb: 1,
          backgroundColor: lightcolor,
          textColor: darkcolor,
          fontSize: screenWidth * 0.037);
      setState(() {
        signinloading = false;
      });
    }
  }
}
```

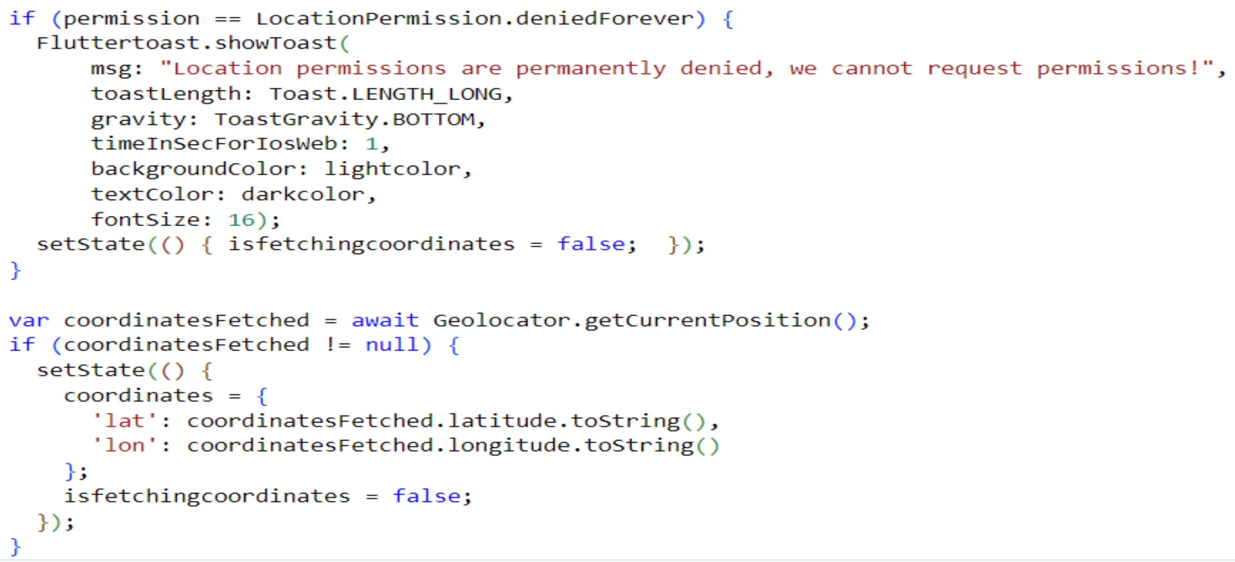
5.3.3 Create Ride

```
double screenWidth = returnScreenWidth(context);
    String startTime = timeselected;
    String destination = pdestination.text.trim();
    String numberPlate = pcarnumberplate.text.trim();
    String description = pcardescription.text.trim();
    if (startTime.isEmpty ||
        destination.isEmpty ||
        numberPlate.isEmpty ||
        description.isEmpty ||
        pcarmaxpeople.text.trim().isEmpty) {
      Fluttertoast.showToast(
          msg: "Please do not leave any textfields blank",
          toastLength: Toast.LENGTH_LONG,
          gravity: ToastGravity.BOTTOM,
          timeInSecForIosWeb: 1,
          backgroundColor: lightcolor,
          textColor: darkcolor,
          fontSize: screenWidth * 0.037);
    } else if (numberPlate.length < 4) {
      Fluttertoast.showToast(
          msg: "Your number plate is not correct",
          toastLength: Toast.LENGTH_LONG,
          gravity: ToastGravity.BOTTOM,
          timeInSecForIosWeb: 1,
          backgroundColor: lightcolor,
          textColor: darkcolor,
          fontSize: screenWidth * 0.037);
    } else {
      setState(() { addrequestloading = true;  });
      int maxpeople = int.parse(pcarmaxpeople.text.trim());
      await onGetCurrentCoordinates();
      if (coordinates.isEmpty) {
        Fluttertoast.showToast(
            msg: "Failed to get GPS coordinates. Please ensure your GPS is enabled!",
            toastLength: Toast.LENGTH_LONG,
            gravity: ToastGravity.BOTTOM,
            timeInSecForIosWeb: 1,
            backgroundColor: lightcolor,
            textColor: darkcolor,
          toastLength: Toast.LENGTH_LONG,
          gravity: ToastGravity.BOTTOM,
          timeInSecForIosWeb: 1,
          backgroundColor: lightcolor,
          textColor: darkcolor,
          fontSize: screenWidth * 0.037);
      setState(() {
        addrequestloading = false;
      });
    } else {
      try {
        var revisedDate = "$dateselected $timeselected:00.000";
        await FirebaseFirestore.instance.collection('Requests').add({
          'requesterID': user['id'],
          'requesterName': user['firstName'] + " " + user['lastName'],
          'requesterEmail': user['emailAddress'],
          'requesterPhone': user['phoneNumber'],
          'requestDate': DateTime.now().toString(),
          'requestToken': user['token'],
          'coordinates': coordinates,
          'startDate': revisedDate,
          'destination': destination,
          'destinationCoordinates': destinationcoordinates,
          'numberPlate': numberPlate,
          'description': description,
          'maximumPeople': maxpeople
        }).then((ref) async {
          setState(() { addrequestloading = false; });
          Navigator.pop(context);
        }).onError((error, stackTrace) {
          Fluttertoast.showToast(
              msg: "There was an error adding your request. Please try again later!",
              toastLength: Toast.LENGTH_LONG,
              gravity: ToastGravity.BOTTOM,
              timeInSecForIosWeb: 1,
              backgroundColor: lightcolor,
              textColor: darkcolor,
              fontSize: screenWidth * 0.037);

              setState(() {
                addrequestloading = false;
              });
            });
          } catch (e) {
            Fluttertoast.showToast(
                msg: "Encountered unknown error. Please try again later!",
                toastLength: Toast.LENGTH_LONG,
                gravity: ToastGravity.BOTTOM,
                timeInSecForIosWeb: 1,
                backgroundColor: lightcolor,
                textColor: darkcolor,
                fontSize: screenWidth * 0.037);
            setState(() {
              addrequestloading = false;
            });
          }
        }
      }
```

5.3.4 Getting Current Location



```
bool serviceEnabled;
LocationPermission permission;
 setState(() {
   isfetchingcoordinates = true;
 });
 serviceEnabled = await Geolocator.isLocationServiceEnabled();
 if (!serviceEnabled) {
   Fluttertoast.showToast(
       msg: "Location services are disabled!",
       toastLength: Toast.LENGTH_LONG,
       gravity: ToastGravity.BOTTOM,
       timeInSecForIosWeb: 1,
       backgroundColor: lightcolor,
       textColor: darkcolor,
       fontSize: 16);
   setState(() {
     isfetchingcoordinates = false;
   });
 }

 permission = await Geolocator.checkPermission();
 if (permission == LocationPermission.denied) {
   permission = await Geolocator.requestPermission();
   if (permission == LocationPermission.denied) {
     Fluttertoast.showToast(
         msg: "Location services are denied!",
         toastLength: Toast.LENGTH_LONG,
         gravity: ToastGravity.BOTTOM,
         timeInSecForIosWeb: 1,
         backgroundColor: lightcolor,
         textColor: darkcolor,
         fontSize: 16);
     setState(() {
       isfetchingcoordinates = false;
     });
   }
 }
```



```
if (permission == LocationPermission.deniedForever) {
  Fluttertoast.showToast(
      msg: "Location permissions are permanently denied, we cannot request permissions!",
      toastLength: Toast.LENGTH_LONG,
      gravity: ToastGravity.BOTTOM,
      timeInSecForIosWeb: 1,
      backgroundColor: lightcolor,
      textColor: darkcolor,
      fontSize: 16);
  setState(() { isfetchingcoordinates = false;  });
}

var coordinatesFetched = await Geolocator.getCurrentPosition();
if (coordinatesFetched != null) {
  setState(() {
    coordinates = {
      'lat': coordinatesFetched.latitude.toString(),
      'lon': coordinatesFetched.longitude.toString()
    };
    isfetchingcoordinates = false;
  });
}
```

5.3.5 Join Ride

```
 setState(() {
    askloading = true;
  });
  await onGetCurrentWaitingList();
  await onGetCurrentWaitingListArray();

  var id = request['id'];

  List waitingList = [];
  List waitingListArray = [];

  Map body = {
    'personID': user['id'],
    'personName': user['firstName'] + " " + user['lastName'],
    'personPhone': user['phoneNumber'],
    'personEmail': user['emailAddress'],
    'personToken': user['token'],
    'comment': askercomment.text.trim(),
    'coordinates': coordinates
  };

  if (currentWaitingList == null) {
    waitingList.add(body);
  } else {
    currentWaitingList.add(body);
    waitingList = currentWaitingList;
  }
  if (currentWaitingListArray == null) {
    waitingListArray.add(user['emailAddress']);
  } else {
    currentWaitingListArray.add(user['emailAddress']);
    waitingListArray = currentWaitingListArray;
  }
try {
  await FirebaseFirestore.instance.collection('Requests').doc(id).update({
    "waitingList": waitingList,
    "waitingListArray": waitingListArray
  }).then((value) async {
    await onSendNotification(
        request['requestToken'],
        "${user['firstName']} ${user['lastName']} has requested for a ride. Open the app to check the details",
        "A user has requested a ride");
    setState(() {
      askloading = false;
    });
    Navigator.push(context, MaterialPageRoute(builder: (context) => const Dashboard()));
  });
} catch (e) {
  setState(() {
    askloading = false;
  });
  Fluttertoast.showToast(
      msg: "Failed to update waiting list. Please try again later!",
      toastLength: Toast.LENGTH_LONG,
      gravity: ToastGravity.BOTTOM,
      timeInSecForIosWeb: 1,
      backgroundColor: darkcolor,
      textColor: Colors.white,
      fontSize: 13);
}
```

5.4 SYSTEM TESTING

System testing is a level of software testing where complete and integrated software is tested. Testing is performed during systems development. In order for a system to be called complete, it must be tested against it proposed objectives, technical, functional and non-functional requirements. Testing is a process that is performed to ensure that the system conforms to the specification and meets the requirements of the standard users of the system, which is a registered member organization. It is the operation of the system or application under controlled conditions and evaluation of results. Controlled conditions should they include both normal and abnormal conditions. Testing should intentionally try to make things go wrong to determine whether the system is capable of handling failures. Failure is a dynamic state that always occurs after the software is launched, test cases that can cause the software to fail have a higher probability show failure. On the other hand, it is also formulated as a technique practiced to prove a check that the system in question actually meets all the business and specialty assumptions that govern it configuration and future improvements. In addition, system testing is used in quality testing software against a number of factors such as reliability, usability, to name but a few categories of testing and the results

5.4.0 WHITE BOX

White box testing is a software testing method in which the internal structure of the item being tested is known to the tester. Code implementation and code impact are tested. White box testing deeply investigates the processing flows defined in the software source code.

5.4.1 BLACK BOX

Black box testing is a software testing method in which the internal structure of the item being tested is not known to the tester. Only the external design and structure are tested. It is source code agnostic and focuses on software behavior. This kind of testing is also known as functional testing. This test method is often used for validation. In this case, the system is seen as a black box that is pumped with inputs to create outputs. There is no requirement to understand the basic processing of the system. Now there is a need compare predicted results with actual results obtained for selection purposes. This testing focuses on the functional needs of the software and allows the software engineer to control the input sets circumstances that will fully meet the functional requirements of the program. He looks for trouble such as incorrect missing functions, interface errors, data structure or external database errors, access, performance, initialization, and termination errors. The program was tested for suite control situations that meet the needs of the user.

5.4.2 GREY BOX TESTING

Grey box testing is a technique to test the software product or application with partial knowledge of the internal workings of an application. The purpose of this testing is to search for defects due to improper code structure or improper functioning usage of an application.

5.5 Testing levels

Testing levels include:

- Unit testing
- Integration testing
- System testing
- Acceptance testing

5.5.0 Unit testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

5.5.1 Integration testing

Integration testing: is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration testing is a type of software testing in which different elements are integrated and tested together. The objective of this level of testing is to identify errors in the interaction of integrated parts. Integration testing is aided by the use of test drivers and test stubs. The imperative of integrated testing is to determine the overall performance of the system. When evaluating the entire program, there is a possibility of reoccurrence of defects, because previously all test methodologies were used to test certain separate modules. Now we would combine them all and evaluate their overall compatibility for all interfaces and the charting process as they are all interconnected. The application has been tested for different kinds of inputs and passed successfully.

5.5.2 System testing

System testing is a level of testing that validates the complete and fully integrated software product. The purpose of a system test is to evaluate the end-to-end system specifications

5.5.3 Acceptance testing

Acceptance testing is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

5.6 TESTING AND RESULTS

5.6.0 Functional Testing

In functional testing, the system is tested against the functional requirements/specifications. Functions (or features) are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application. This type of testing is not concerned with how processing occurs, but rather, with the results of processing. It simulates actual system usage but does not make any system structure assumptions. Functional testing verifies that each function of the software application operates in conformance with the requirement specification. Test cases were made on the functional requirements of the system so as to perform the functional testing of the system. The proposed system main function is to match rides between drivers and passengers and allow booking of rides that is posting, searching and sharing of route information.

Table 5: Functional testing

Functional requirements	**Result**
Allow users to register and login into the system.	Success
Allow use Google maps to view current location.	Success
Allow users to create rides	Success

Allow system to send location to users	Success
Allow users to join rides	Success

5.6.1 Non-Functional Testing

Non-functional testing focuses on testing the non-functional requirements of systems. Non-functional requirements are those that reflect the quality of the product, especially in the context of the suitability of its users. Thus, non-functional testing is a software testing technique that verifies system attributes such as system performance, usability, and robustness. This testing is done at all test levels. Used to verify non-functional aspects of the project.

Table 6 Non Functional Testing

Domain	**Test Case**	**Results**
Efficiency	System should load within 60 seconds.	Success
Usability:	The system can be easily installed and used	Success
Security	Only the owner of the machine can be allowed access	Success

5.6.2 UNIT TESTING

Individual components are checked to verify that they function properly during unit testing. When tests are performed on distinct components or mechanisms of a software system, it is referred to as software testing. Each component is tested independently of other system components and these components can be small units, modules or system functions. Unit testing was done to see if it validates the flag inputs. Individual modules are checked for good function and are determined to be satisfactory in terms of the intended performance of the module. The entire project is divided into parts and each module is tested independently of the others and their functionality.

Table 7 Unit Testing

Test Id	Scenario Condition	Input1: route detail	Input 2: date details	Input 3: car details	Expected output

1	Scenario 1. Add Pool Request	Destination Source (current coordinates)	Start Date Start Time	Car description Car Number Plate	Ride successfully added
2	Scenario 2. Add Pool Request alternative flow: unauthenticated user	N/A	N/A	N/A	Back to Login Screen
3	Scenario 3. Add pool request alternative flow: System error	Destination Source (current coordinates)	Start Date Start Time	Car description Car Number Plate	Error message: No internet connection
4	Scenario 4. Add pool request alternative flow: Location disabled	Destination	Start Date Start Time	Car description Car Number Plate	Error message: Location services are disabled
5	Scenario 4. Add pool request alternative flow: invalid entry	Destination Source (current coordinates)	N/A	Car description Car Number Plate	Error message: Please do not leave any textfields blank

5.6.3 Validation Testing

At the culmination of Black Box testing, the software is completely assembled as a package and tested as a whole unit. Validation testing ensures that the product actually meets the project requirements. It can also be defined as evidence that a product meets its intended use when deployed in an appropriate environment. The process of evaluating software during the development process or at the end of the development process to determine whether it meets specified business requirements. Verification testing is where the requirements established in the software requirements analysis are verified against the software that has been constructed. It ensures that the software meets all functional, behavioral and performance requirements. The application was tested on various inputs and passed successfully.

Table 8 Validation Testing

TCI	Objective	Description	Input	Expected Output	Actual Output	Result
1	Validating Registration	Verifying that all fields are filled	String values from user	An alert to fill all the required fields	An alert to fill all the required fields	Successful
2	Validating Login	Verifying that all fields are filled	String values from user	User not found	User not found	Successful
3	Validate Add pool request	To verify that trip details are entered successfully	Enter destination and car details	An alert to fill all the required fields	An alert to fill all the required fields	Successful
4	Confirmation of sent requests	Gives the confirmation that a rider has entered into the ride	Join ride	Notification to the driver	Notification to the driver	Successful

5.6.4 Systems Testing

System testing involves the integration of components into the system. The process focuses primarily on finding bugs that result from unexpected interactions between components and component interface problems. All the objectives of this system have been met.

Table 9 System Testing

Domain	Expected Result	Actual Result
Black Box Testing	The system should accept user input and produces the desired result.	As expected
Functional Testing	The system modules should function as expected and produce the desired output. All the functional requirements should be met.	As expected
Non-functional Testing	All the non-function requirements have been met. The system is secure and efficient.	As expected

5.7 SYSTEM EVALUATION

System evaluation is the process of evaluating the performance of an entire system to determine how it is likely to perform in the outside world. The developed solution has successfully passed various testing paradigms and sample users have adopted the system as a reliable, sustainable and useful solution that provides credible career insights and advice. All the objectives of the proposed system were met

Here is a brief evaluation of the decentralized carpool application:

1. Performance: The performance of the decentralized car sharing application is measured based on the speed and accuracy of creating rides on a driver side and the rider joining the ride. The system should be able to display a recently created ride to appear immediately on the rider's side.

2. Reliability: The reliability of the decentralized car pool application is measured by its ability to function consistently and without errors over time. The system should be able to connect drivers and riders accurately, without crashing or encountering errors.

3. Usability: The usability of the decentralized car pool application is measured by its ease of use, user interface, and user experience. The system should be user-friendly, with an intuitive interface that allows users to operate the system easily.
4. Security: The security of the decentralized car pool application is measured by its ability to protect user data and information. The system should implement appropriate security measures, such as data encryption and user authentication, and not to allow anyone to change user's information.
5. Compatibility: The compatibility of the decentralized car sharing application is measured by its ability to work seamlessly across different devices, platforms, and software versions. The system should be able to function consistently and without errors across a range of environments.

5.8 INSTALLATION, DEPLOYMENT AND MANTAINANCE

In the deployment, we used a parallel conversion, where the old and new computer systems are operated simultaneously. Some people are latecomers, meaning they will be slow to adopt new technology and prefer an already existing system, a system that cannot be fully implemented as a stand-alone.

It is still not ready for public deployment on the Play Store. The main reason is that it takes time to approve and publish an app on the Play Store. The application will be uploaded to the Google Play Store and other platforms where the user can download and install it on their mobile phones. During maintenance, the app will rely on user feedback and make some necessary updates.

5.9 CONCLUSION

This chapter gave the opportunity to carry out test cases as there was a need to determine whether the system built was correct or accurate and also consider how the proposed system was developed using different test criteria. The objectives were met and there were no problems navigating the system functions.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.0 Introduction

This study includes a unique methodology for a decentralized carpooling model that aims to create an ideal match between passengers who want carpooling services and relevant carpooling-ready drivers. The server continuously acquires information and preferences from passengers and drivers, and passengers for drivers, in terms of proximity in time and place, as well as the compatibility of characteristics and preferences between passengers, drivers and passengers on board. This article elaborates on the proposed system, which consists of three primary modules: Offer a Ride, Search for a Ride, and authenticate the user through Registration. This system uses Google Maps services and the GPS module to create user-specific services. This ultimately leads to the construction of a well-organized transportation facility. It increases transport efficiency by reducing fuel consumption and carbon emissions by reducing the number of cars on the road and reducing traffic congestion. This results in cost savings for users as they can split the costs of fuel, parking and tolls. As a result, it is an environmentally friendly application.

The primary objective of this study is to present a preliminary prototype of the proposed system. This decentralized car sharing app adheres to enterprise class app standards. It is designed to be fast, scalable, extensible and highly available. It also protects the privacy and security of user data. The program is also easy to maintain as it can be improved in various ways. This project has produced a working Android application that meets the standards outlined in this document. It's still not ready for public release on the Play Store. The main problem is that it takes time to get an app approved and published on the Google Play Store. A limitation that should be considered is the time required to build the server and Android application. This should be factored into the time allocated to each of these activities. Many aspects of the current application can be improved due to lack of time. This presents a more attractive user interface with a more attractive design. Adding additional authentication system support may also be beneficial.

6.1 Results

The decentralized car sharing application project has been successfully implemented and it has been proven to be effective in reducing the number of cars hence reducing fuel consumption. The application provides users with an interactive platform to share rides easily and conveniently.

During the testing phase, the application underwent various types of testing, including functional testing, non-functional testing, unit testing, integration testing, and validation testing. These tests helped to identify and fix several bugs and errors, ensuring that the application is of high quality and performs optimally.

The implementation and testing of the decentralized car pooling application have resulted in a robust and reliable application that can serve as a valuable resource for individuals interested in sharing rides. The application has the potential to make a positive impact on the environment as it aims to reduce global warming and drivers who want to travel on roads without facing severe congestion.

The table below checks if goals set have been successfully achieved, the ones that have been achieved are ticked to confirm.

Table 10 Results

Objectives	Fully achieved	Partially achieved	Exceeded
➢ To create a decentralized android application where people will find car owners using the same routes as they are.	●		
➢ To use google maps to get current location for a user	●		

➢ To include notifications which will alert car pooler	•		

Overall, the decentralized car sharing application project has been a success, and it is poised to be an essential tool for reducing global warming and reduce the number of cars on the road dealing with the issue of traffic jams.

6.2 Scope of Future work.

The decentralized car sharing application project has opened up many possibilities for future work and improvements. Some of the areas that could be explored include:

- Allowing user to comment to an event could improve the coordination among users. Driver rating technique if used to give a feedback about the drivers and also in application chatting between the passengers and drivers.
- Use of recommendation algorithms to recommend drivers to passenger. Use of neural network algorithms such as Convolution Neural Network (CNN) pooling system can be for transportation goods in sharing manner (Truck Pooling).
- Using ads within the pages of the application it can also be used for advertising.
- To deploy the application on Google Play Store and have another application for IOS and windows phones.
- Use blockchain rather than databases

Overall, decentralized carpool application has a lot of potential for future work and improvements. By addressing the areas mentioned above and other potential areas, the application can be made more useful and accessible to users.

6.3 Recommendations

Based on the evaluation and results of the decentralized carpool application project, the following recommendations can be made:

- Add more database to the already decentralized carpool application
- Usage of blockchain
- Use of payment gateway
- Use of group chats

6.4 Challenges

- Getting a valid visa card to access Google Maps API which needed a billing account.
- Data for internet access

REFERENCES

[1] T. Copel, Noam and Ater, “DAV White Paper,” tech. rep., 2017.

[2] F. Vogelsteller and V. Buterin, “Erc-20 token standard,” Ethereum Foundation (Stiftung Ethereum), Zug, Switzerland, 2015.

[3] FFQuest, “FFQuest - Blockchain marketplace for Car rental, Ridesharing and Parking.” Accessed: 2019-04-15.

[4] WONO, “WONO White Paper,” tech. rep., 2018.

[5} Meshkani, S. M., & Farooq, B. (2021). A Decentralized Shared CAV System Design and Application. arXiv preprint arXiv:2104.10022.

[6] Lei, A., Cruickshank, H., Cao, Y., Asuquo, P., Ogah, C. P. A., & Sun, Z. (2017). Blockchain-based dynamic key management for heterogeneous intelligent transportation systems. IEEE Internet of Things Journal, 4(6), 1832-1843.

[7] Jabbar, R., Kharbeche, M., Al-Khalifa, K., Krichen, M., & Barkaoui, K. (2020). Blockchain for the internet of vehicles: A decentralized IoT solution for vehicle communication using ethereum. Sensors, 20(14), 3928.

[8] Wang, Q., Ji, T., Guo, Y., Yu, L., Chen, X., & Li, P. (2020). TrafficChain: A blockchain-based secure and privacypreserving traffic map. IEEE Access, 8, 60598-60612.

[9] Rahardja, U., Aini, Q., & Maulana, S. (2021). Blockchain innovation: Current and future viewpoints for the travel industry. IAIC Transactions on Sustainable Digital Innovation (ITSDI), 3(1), 8-17

[10] Wang, Y., Kim, D. K., & Jeong, D. (2020). A survey of the application of blockchain in multiple fields of financial services. Journal of Information Processing Systems, 16(4), 935-958.

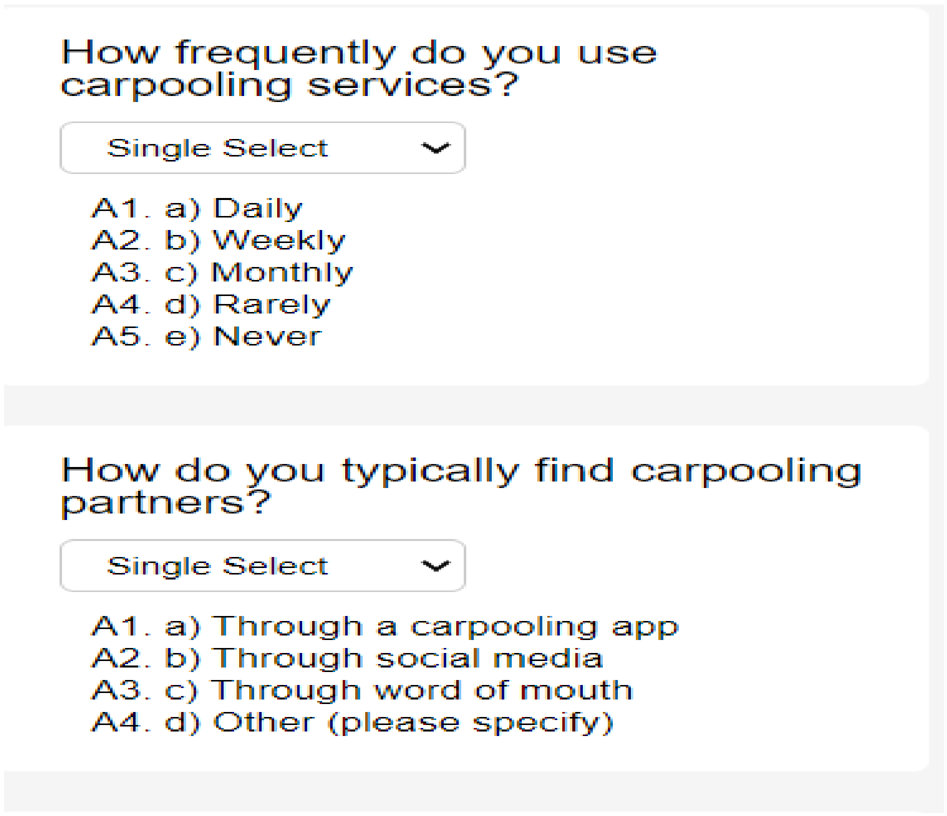
[11] Chau, S. C. K., Shen, S., & Zhou, Y. (2020). Decentralized ride-sharing and vehicle-pooling based on fair costsharing mechanisms. IEEE Transactions on Intelligent Transportation Systems

[12] Ryan M Shivers. Toward a secure and decentralized blockchain-based ride-hailing platform for autonomous vehicles. PhD thesis, Tennessee Technological University, 2019.

[13] Meng Li, Liehuang Zhu, and Xiaodong Lin. Coride: A privacy-preserving collaborative-ride hailing service using blockchain-assisted vehicular fog computing. In International Conference on Security and Privacy in Communication Systems, pages 408–422. Springer, 2019.

[14] Meng Li, Liehuang Zhu, and Xiaodong Lin. Efficient and privacypreserving carpooling using blockchain-assisted vehicular fog computing. IEEE Internet of Things Journal, 6(3):4573–4584, 2018.

APPENDIX 1: DATA COLLECTION TOOLS



How frequently do you use carpooling services?

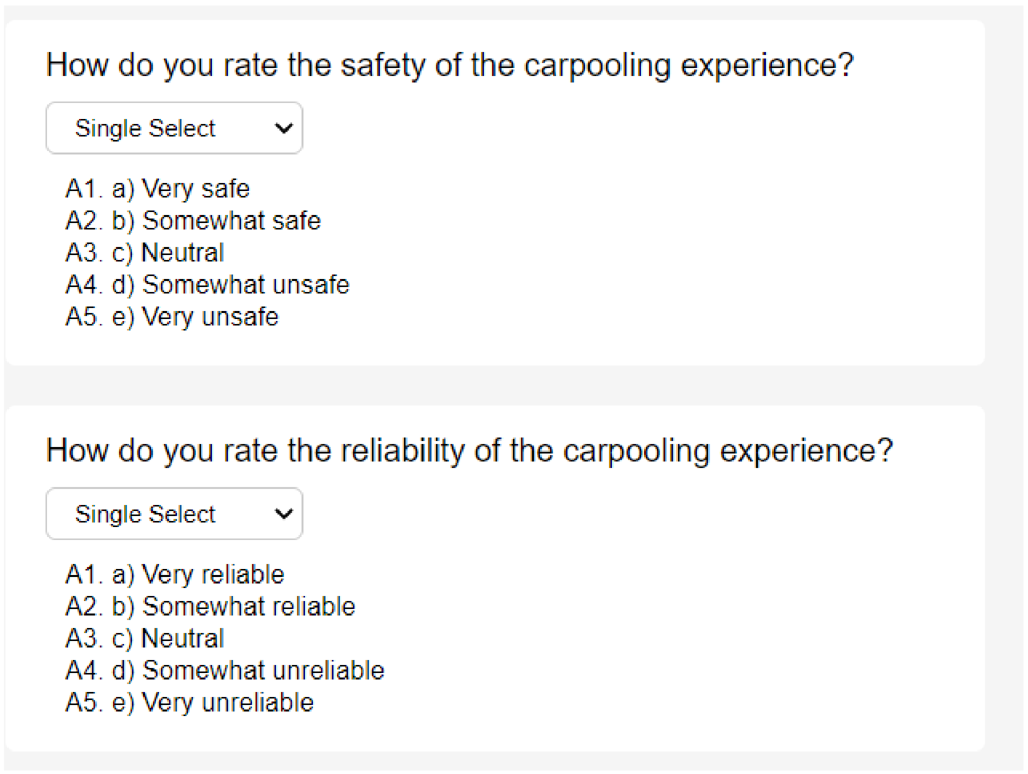
Single Select

A1. a) Daily
A2. b) Weekly
A3. c) Monthly
A4. d) Rarely
A5. e) Never

How do you typically find carpooling partners?

Single Select

A1. a) Through a carpooling app
A2. b) Through social media
A3. c) Through word of mouth
A4. d) Other (please specify)



How do you rate the safety of the carpooling experience?

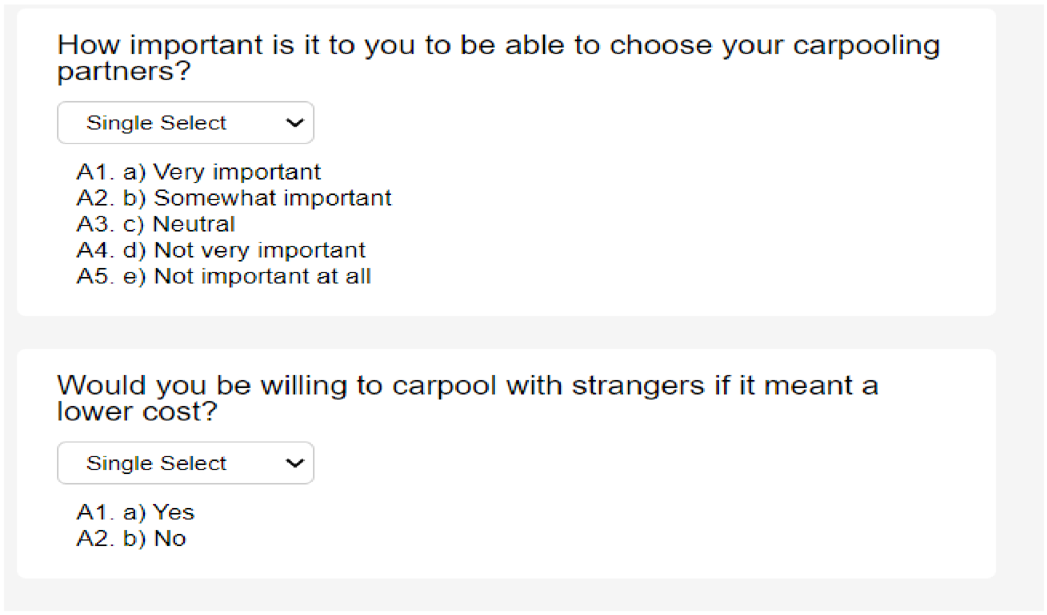
Single Select

A1. a) Very safe
A2. b) Somewhat safe
A3. c) Neutral
A4. d) Somewhat unsafe
A5. e) Very unsafe

How do you rate the reliability of the carpooling experience?

Single Select

A1. a) Very reliable
A2. b) Somewhat reliable
A3. c) Neutral
A4. d) Somewhat unreliable
A5. e) Very unreliable



How important is it to you to be able to choose your carpooling partners?

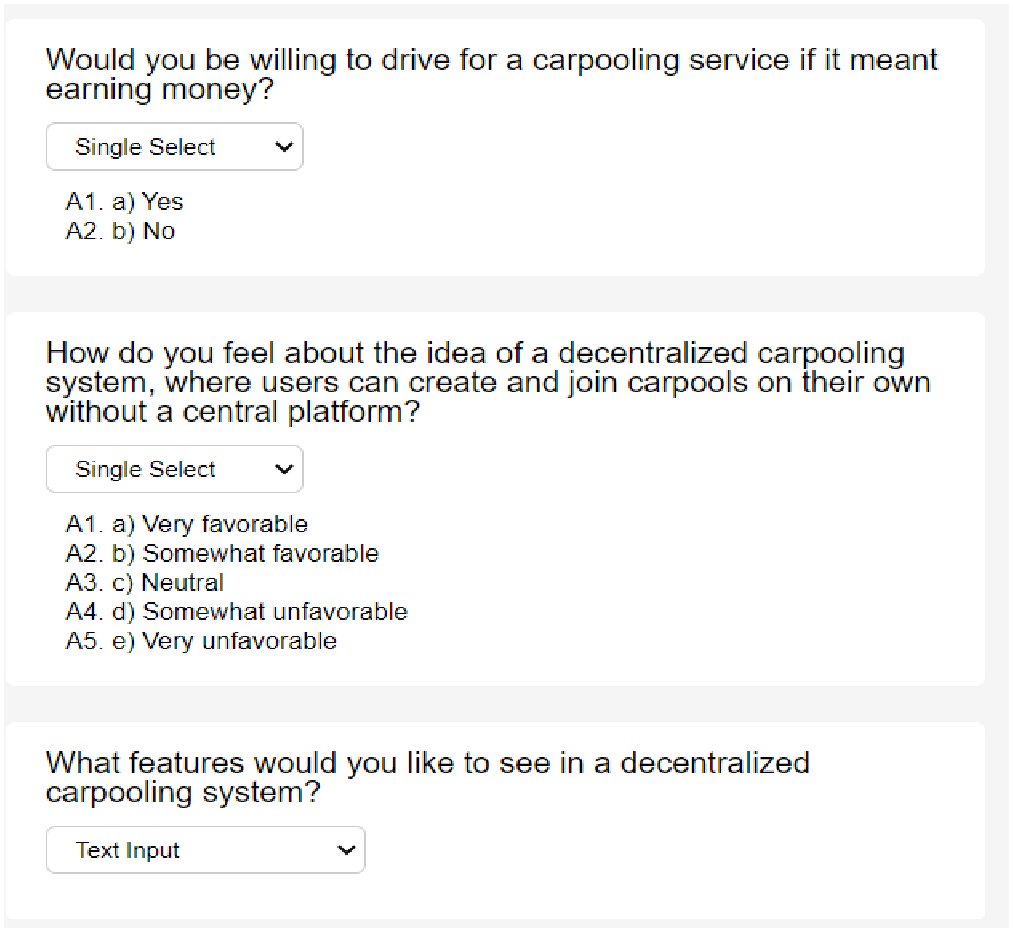
Single Select

A1. a) Very important
A2. b) Somewhat important
A3. c) Neutral
A4. d) Not very important
A5. e) Not important at all

Would you be willing to carpool with strangers if it meant a lower cost?

Single Select

A1. a) Yes
A2. b) No



Would you be willing to drive for a carpooling service if it meant earning money?

Single Select

A1. a) Yes
A2. b) No

How do you feel about the idea of a decentralized carpooling system, where users can create and join carpools on their own without a central platform?

Single Select

A1. a) Very favorable
A2. b) Somewhat favorable
A3. c) Neutral
A4. d) Somewhat unfavorable
A5. e) Very unfavorable

What features would you like to see in a decentralized carpooling system?

Text Input

APPENDIX 2: USER MANUAL

For Driver

Driver must own a car with vacancy seats to carry other passengers but not exceeding the maximum required by the law

Registration and Login

1. Open the application
2. If you don’t have account click create account link
3. Enter user details required to registered
4. Login with phone number and password used in registration
5. Securely keep authentication credentials

Offer screen

1. Click offer ride button to offer ride
2. Enter required details
3. Click offer button to save your ride
4. Navigate to rides page to see your rides you have created

My Request screen

1. Click on the view button to view a specific ride
2. Click on the delete button to delete the specific ride

For Rider

Registration and Login

1. Open the application
2. If you don’t have account click create account link
3. Enter user details required to registered
4. Login with phone number and password used in registration
5. Securely keep authentication credentials

Sent Request screen

1. Click on the ride to view available ride details
2. Click on the join ride to join a specific ride

APPENDIX 3: SOURCE CODE

```
void main() async {
  WidgetsBinding widgetsBinding = WidgetsFlutterBinding.ensureInitialized();
  FlutterNativeSplash.preserve(widgetsBinding: widgetsBinding);
  SystemChrome.setPreferredOrientations(
      [DeviceOrientation.portraitUp, DeviceOrientation.portraitDown]);
  try {
    await Firebase.initializeApp().then((value) => {
          FlutterError.onError = FirebaseCrashlytics.instance.recordFlutterError
        });
  } catch (e) {}

  runApp(MaterialApp(
    initialRoute: '/',
    routes: {
      '/': (context) => const MyHomePage(),
    },
    debugShowCheckedModeBanner: false,
  ));
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({Key key}) : super(key: key);

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  @override
  void initState() {
    onGetTrackingConfirmation();
    onStart();
    super.initState();
  }

  @override
  void dispose() {
    super.dispose();
  }
  onGetTrackingConfirmation() async {
    if (await AppTrackingTransparency.trackingAuthorizationStatus ==
        TrackingStatus.notDetermined) {
      try {
        await AppTrackingTransparency.requestTrackingAuthorization();
      } catch (e, s) {
        //failed to process tracking request
      }
    }
  }

  onStart() async {
    String userDetails = await onGetPreference('peerpool_logged_user');

    if (userDetails == null) {
      Navigator.push(
          context, MaterialPageRoute(builder: (context) => const Login()));
      FlutterNativeSplash.remove();
    } else {
      Navigator.push(
          context, MaterialPageRoute(builder: (context) => const Dashboard()));
      FlutterNativeSplash.remove();
    }
  }

  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

APENDIX 4: TECHNICAL PAPER

DECENTRALIZED CARPOOL SYSTEM

Kudakwashe Exstaff Koti[a]; Chibaya Amanda[b]

Department of Software Engineering, School of Information Sciences and Technology, Harare Institute of Technology, Harare, Zimbabwe

kudakotie@gmail.com[a]; achibaya@hit.ac.zw[b];

ABSTRACT

Existing carpooling optimization techniques based on a centralized approach serve policy makers' goals but neglect the reality of participants. Moreover, without strict enforcement, participants often ignore centralized solutions and maximize their own savings. We present a new heuristic, formulated and tested on real and simulated carpooling problem cases, that mimics the decentralized carpool self-organization process. Our findings reveal system-wide savings similar to centralized models and a potential strategy for improving carpooling use. Decentralized ridesharing is a system in which individuals can arrange rides with each other without the need for a centralized organization or platform. Instead, participants use a peer-to-peer network to connect with other riders or drivers traveling in the same direction.

This system enables more efficient use of resources, reduces traffic congestion and carbon emissions, and provides a more affordable alternative to traditional modes of transportation. The decentralized nature of the system also provides greater flexibility and autonomy for participants to arrange rides on their own terms without relying on a third-party platform or service. Overall, decentralized ridesharing has the potential to change the way we think about transportation and contribute to a more sustainable and equitable future. The system proposed by this study combines route customization, i.e. seamlessly integrated with GPS, Google Maps, Android mobile app and Firebase database.

I. INTRODUCTION

The study focused on decentralized car sharing system using mobile phones specifically on the Android platform.

The car-sharing market is constantly growing and recently it has become popular more than car ownership. This classic car sharing system is based on a centralized database server which can often lead to hacker attacks or password leaks.

Moreover, in a classic car-sharing system the owners of the car can misuse customers' data. Nowadays we have come to see from a lot of use cases, that the best solution is to use blockchain technology. The blockchain technology is decentralized, immutable, public ledger provides the customers with security that is impossible to tamper with. The aim of the proposed system is to create and implement peer-to-peer short term car-sharing application that is decentralized.

This project focused on providing an interactive way for drivers to connect with users in a decentralized way.

II. PROBLEM STATEMENT

Lately there is a problem of traffic on our roads and therefore the increasing fuel costs boost the misery of daily users of private vehicles. The number of vehicles plying on the road is rising as each day passes due to the low-cost cheap imports from Japan. This gives rise to few prime factors that square measure pollution, depletion and rising rates of fossil fuels and massive traffic congestion. The current system well known as mushika shika has been deemed illegal, with touts being suspected of rowdy behavior which has resulted in the inconvenience of travelers. Other solutions at hand include taxi management system, in this system the taxi driver, passenger and management system are important entities. These components interact with one other in existing taxi management systems. Because of the widespread acceptance and use of smart phones in the real world, we assume that each 1 passenger and taxi driver is equipped with a smart portable communication device, such as a smart phone or a computer. Each passenger contributes to the cab management centre, he/she must communicate his/her present position and desired destination.

The taxi management centre must be established, send the taxi to carry the passengers efficiently from their designated starting points to the intended destinations. Taxi drivers who are available to convey the passenger will react to the taxi management centre to express their desire to pick up the passenger. The cab control centre then moves on to the next passenger's request.

. The previously described practice of existing taxi management systems has various drawbacks. For example, passengers in a single cab cannot readily share a journey with one another to reduce costs. The transportation system is inefficient. As a result, there is no way to minimize air pollution and carbon emissions. Carpooling entails two or more passengers traveling in the same direction in a private car along a partially shared route. The idea of carpooling is that some individual passengers' journey time is sacrificed in order to receive additional benefits such as sharing driving costs, parking fees, reduced traffic congestion, or lower carbon emissions. Unfortunately providing carpooling with general systems to users brings a lot of security flaws, first of all is overall stability of service.So decentralization will add transparency and ensures data protection as it is termed an unchangeable method.

III. RELATED WORK

<u>HireGo</u>

It is built on Ethereum using smart contracts as a distributor of virtual fungible tokens (standard ERC-20) [1] called HGO. If one wants to rent a car using HireGo, he or she has to exchange ETH to HGO tokens. HireGo’s market places has three contracts. The first contract of HireGo provides HGO tokens and the ERC-721 car tokens are provided by the second contract and lastly the third contract provides rental contract. The rental contract is responsible for Vehicle token between two parties and also acts as an escrow. So what happens is when one wants to rent a car, he or she sends HGO tokens to rental smart contract which locks car token and HGO tokens for selected period. After this the contract then manages the transfer of the vehicle token from owner to us. The rental solution also comes up with big security issue. If we rent a car for a day trip and then we lose the phone with Ethereum address, the car will be stuck for the rest of time and the owner is unable to help us because he/she does not own that vehicle at the time of renting.

<u>Helbiz</u>

Its philosophy is to create and integrate transport ecosystem where users are rewarded for using this system. The project is using well-known implementation of blockchain car-sharing platform by providing ERC-20 tokens called HBZ. If the user does not have enough tokens, the application offers the user to but the HBZ tokens through the use of credit card.

DAV

Its main goal is to connect autonomous vehicles with users. The use the DAV token for communication and this DAV token also acts as an access token, Services in the DAV network can be bought with virtual tokens based on Ethereum. If a person is owning a vehicle or charging station which he or she is providing to other users in the network he or she is rewarded with DAVE tokens [2]. Autonomous vehicles automatically interact with environment and also can fulfil user's commands.

FFQuest

Its main strength is the use of their own chain called Distributed FFQ Ledger based on Ethereum Virtual Machine. All the transaction details between car companies, drivers and customers such as payment conditions, assets, availability and reservations by using their own utility token type ERC-20 called FFQ are passes by the ledger. This project uses central backend to mirror Distributed FFQ Ledger on a server and also for storing videos and images [3].

WONO

It provided a decentralized market for renting services such as cars and working man days. They used a set of smart contracts that work as a bridge between Ethereum main net and their blockchain. Their blockchain uses Proof-of-stake algorithm to ensure consensus in the network. The IPFS file system which is a decentralized system is used to store users; profile data represented as JSON. Data properties which contain confidential information are encrypted on a client's app side by a public key of the User's address and can be decrypted only by the privatekey [4].

Meshkani

With blockchain technology emerging quickly, significant efforts are being made to incorporate this technology in the transportation sector. Meshkani [5] suggested an empirical ride sharing method where individual users can share their ride with only a single user. Lei [6] and Abbas [7] have suggested decentralized data management systems that are integrated with blockchain and IoT to provide better transparency, data sharing and tracking.

IV. SOLUTION

In recent years, the problems of global warming and the energy crisis have aroused widespread public concern. One recommended solution for reducing the harmful factors leading to such problems is carpooling. This type of transportation service could make a big difference if organized on a large scale by government or big companies, particularly large corporations with many branches or sub companies. Carpooling schemes are designed to encourage commuters to share travel expenses and resources with colleagues. The proposed system overcomes the drawbacks of the existing system. It has advanced facilities to make it more user-friendly. It provides details of the owner and his/her car to maintain transparency between users of the system. It will track the location of users those who are involved in the pool through GPS Navigation system.

Overview of the proposed system

Users:
Two types of users are involved in the proposed system that is the driver and passenger.

Passengers:
Passenger is any person that doesn't own a car and wants to join a rider in a ride. The passenger posted and conditions specified (price and general agrees to all the behaviour).

Find ride and choose ride:
When a rider goes to the sent requests he or she can see the available rides from the drivers. He or she can see the driver nearest to him through the shown distance. The rider can choose the ride of his or her own writing. A notification is sent to the driver notifying him or her of the rider who has joined the ride.

Driver:
Driver is the person providing carpooling Services using his own car.

Schedule ride and choose passenger: The driver can create a new ride to be displayed when passengers search for ride. The application will prompt the information of the ride which consists of destination, origin, meeting, departure time/date, estimated arrival time and travelling preferences. The driver accepts the passenger seeking ride if he/she meets the requirements of the trip.

Profile registration
System's users have their own profiles, where they store a description and have important information. The available

information is the description provided by the user, name to be recognized, workplace or education institution the user is inserted in and vehicle information if it is the case of a driver. At the same time these profiles, depending on what info is available, give to other users some validation and trust when it comes to share a trip.

User Alerts

When important actions occur, users get a notification warning for those, for example, that their status in the lift has been changed, either because his request to join the lift has been accepted, or denied, or the lift itself has been cancelled. This is important because, smart phone users do not have time to check all their applications for updates, instead, the applications must notify the users if such updates exist or else they will just go unnoticed.

A. Solution Architecture

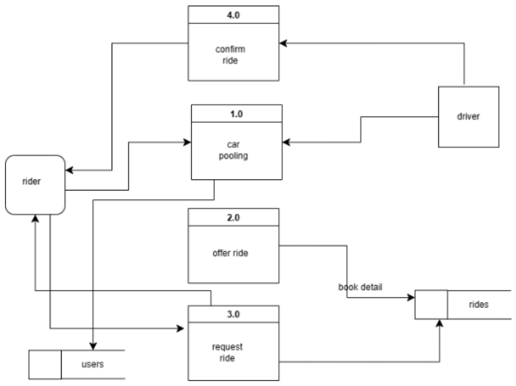


Figure 1: DFD level 1

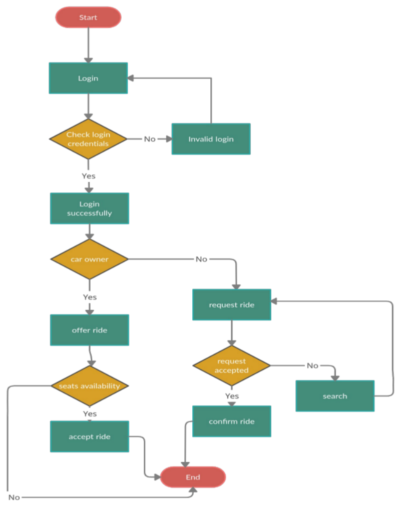


Figure 2: Activity Diagram of the proposed system

B. Coding Strategy

The coding strategy is the series of steps taken to accomplish all the objectives in a project. As the size of this project was big, this project was divided into different modules. The structure and relationships between classes was defined first before the classes where created. Some of the features that were developed using trial and error until the desired results were obtained.

C. *Experimentation and Testing*

Domain	**Expected Results**	**Actual Results**
Functional Testing	• The system modules should function as expected. • The system should be easily accessible and user friendly. • Error messages should be displayed on the system.	As expected
Integration Testing	All the integrated modules should work together flawlessly.	As expected
System Testing	All the components of the system should function properly.	As expected
Acceptance Testing	The system should meet user requirements and system objectives.	As expected

V. CONCLUSION

This study features a unique methodology for the Decentralized Rideshare model, which aims to establish ideal matches between passengers wanting rideshare services and relevant drivers ready to carpool. The server continually gets information and preferences from passengers and drivers, and passengers to drivers in terms of closeness in time and place, as well as compatibility of traits and preferences among passengers, drivers, and passengers on board. This paper elaborates on the proposed system, which is made up of three primary modules: Offer a ride, Seek a ride and user verification via Registration. This system uses Google Maps services and a GPS module to create user-specific services. It eventually leads to the construction of a well-organized transportation facility. It is a program targeted at lowering fuel usage and carbon emissions. As a result, it is an environmentally beneficial application. The primary goal of this study is to present the preliminary prototype of the suggested system.

We now realize that mobile technology is a broad and deep subject to master, that there are multiple ways to leverage it, and that app development is challenging and requires extensive planning and optimization. Decentralized carpooling is an effective way to reduce air pollution, parking problems, fuel consumption, commuting costs based on shared use of private cars or vehicles, mismanagement of user data, and also to increase the transparency of the system. In this paper, we study the carpooling problem and develop a decentralized carpooling prototype to implement ridesharing on the smartphone platform and Google Map API. Our proposed decentralized mobile ridesharing app aims to generate a simple ridesharing app that can help passengers and drivers and ensure no privacy violations. Finally, and after understanding the mobile technologies and solutions available for use, and understanding how car sharing work, we can determine the best options to use to build our app.

VI. FUTURE WORK

The decentralized car sharing application project has opened up many possibilities for future work and improvements. Some of the areas that could be explored include:

- allowing user to comment to an event could improve the coordination among users. Driver rating technique if used to give a feedback about the drivers and also in application chatting between the passengers and drivers.
- Use of recommendation algorithms to recommend drivers to passenger. Use of neural network algorithms such as Convolution Neural Network (CNN) pooling system can be for transportation goods in sharing manner (Truck Pooling).
- Using ads within the pages of the application it can also be used for advertising.
- To deploy the application on Google Play Store and have another application for IOS and windows phones.
- Use blockchain rather than databases

BIBLIOGRAPHY

Kudakwashe Exstaff Koti is a final year student studying Software Engineering at HIT

REFERENCES

[1] Nakamoto, S.: ‘Bitcoin: A peer-to-peer electronic cash system’, in Editor

(Ed.)^(Eds.): 'Book Bitcoin: A peer-topeer electronic cash system' (2008, edn.)

[2] Teigland, R., Yetis, Z., and Larsson, T.O.: 'Breaking out of the bank in Europe-exploring collective emergent institutional entrepreneurship through bitcoin', Available at SSRN 2263707, 2013

[3] Helbing, D.: 'Systemic Risks in Society and Economics', in Helbing, D. (Ed.): 'Social self-organization: Agent-based simulations and experiments to study emergent social behavior' (Springer, 2012)

[4] Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G.M., and Savage, S.: 'A fistful of bitcoins: characterizing payments among men with no names', in Editor (Ed.)^(Eds.): 'Book A fistful of bitcoins: characterizing payments among men with no names' (ACM, 2013, edn.), pp. 127-140

[5] Barber, S., Boyen, X., Shi, E., and Uzun, E.: 'Bitter to better—how to make bitcoin a better currency', in Editor (Ed.)^(Eds.): 'Book Bitter to better—how to make bitcoin a better currency' (Springer, 2012, edn.), pp. 399-414

[6] Laidler, D.: 'The definition of money: theoretical and empirical problems', Journal of Money, Credit and Banking, 1969, 1, (3), pp. 508-525

[7] Brezo, F., and Bringas, P.G.: 'Issues and risks associated with cryptocurrencies such as Bitcoin', 2012