



**BALANCING THE TRIPLE BOTTOM LINE IN LAST MILE DISTRIBUTION: A DIGITAL
MOBILE APPROACH**

by

MANGALISO PHILLIP SIPAMLA

submitted in accordance with the requirements for
the degree of

MASTER OF COMPUTING

in the subject

COMPUTING

at the

UNIVERSITY OF SOUTH AFRICA

SUPERVISOR:

PROF MARCIA MKANSI

CO-SUPERVISOR:

PROF ERNEST MNKANDLA

FEBRUARY 2023

DECLARATION

Name: Mangaliso Phillip Sipamla

Student number: 57661847

Degree: Master in Computing

Balancing the triple bottom line in last mile distribution: A digital mobile approach

I declare that the above dissertation is my own work and that all the sources that I have used or quoted have been indicated and acknowledged by means of complete references.

I further declare that I submitted the dissertation to originality checking software and that it falls within the accepted requirements for originality.

I further declare that I have not previously submitted this work, or part of it, for examination at Unisa for another qualification or at any other higher education institution.

Mr MP Sipamla

DATE

ABSTRACT

The digital age has extensively changed the way consumers make their purchases from in-store to online purchases. With this shift, companies are struggling to maintain and balance the trade-offs between profits, people and planet (3Ps). Companies are challenged to offer on-time guaranteed delivery frequently along with seasonal demand and volatility that is within good margins while considering the environment and society. To address this challenge, academics and industries have studied various distribution solutions such as crowdsourcing. However, little research is observed on a digitally enabled solution that balances the trade-offs between the 3Ps, especially the last-mile industry which has seen increased sale activities and environmental impact. Proceeding from quantitative computational research, this study aims to develop a digitally enabled mobile application that links a package to a traveller already moving in the same direction in order to positively influence the 3Ps and ultimately achieving a triple bottom line solution.

KEY TERMS: Crowdsourcing, triple bottom line, distribution models, digitally enabled solutions, last mile delivery, supply chain system, deliveries, people, planet, profit.

CONTENTS

DECLARATION	i
ABSTRACT	ii
ACKNOWLEDGEMENTS	xi
CHAPTER 1: BACKGROUND TO THE PROBLEM	12
1.1 A REVIEW OF EXISTING SOLUTIONS TOWARDS THE TRIPLE BOTTOM LINE	14
1.1.1 User's freight	14
1.1.2 Outsourcing the last mile	14
1.1.3 Drones and robotics	15
1.1.4 Rail strategies.....	15
1.1.5 Click and collect.....	16
1.1.6 Cyclist and pedestrians.....	16
1.2 BRIEF OVERVIEW OF CROWD LOGISTICS RESEARCH	17
1.2.1 Drivers of the last mile distribution demand	19
1.3 PROBLEM STATEMENT	20
1.4 RESEARCH AIM	21
1.5 RESEARCH OBJECTIVES	21
1.6 RESEARCH QUESTIONS	21
1.7 SIGNIFICANCE OF STUDY	21
1.8 THE SCOPE AND LIMITATIONS OF THE STUDY	22
CHAPTER 2: LITERATURE REVIEW	23
2.1 BACKGROUND REVIEW OF CROWDSOURCING DISTRIBUTION MODEL	23
2.2 SUPPLY CHAIN COORDINATION THEORY	25
2.2.1 Logistics synchronisation.....	26

2.2.2 Information sharing	27
2.2.3 Incentive alignment.....	27
2.2.4 Collective learning	27
2.3. INFORMATION SYSTEMS THEORY	27
2.4 TRIPLE BOTTOM LINE.....	28
2.4.1 People	28
2.4.2 Profit	29
2.4.3 Planet	30
2.5 CROWDSOURCING LOGISTICS MODEL COMPONENTS	30
2.5.1 Creation	31
2.5.2 Couriers	31
2.5.3 Senders and recipients	32
2.5.4 Offers.....	32
2.5.5 Revenue model.....	33
2.5.6 Character.....	33
2.6 CONCEPTUAL MODEL	33
2.7 SUMMARY	35
CHAPTER 3: REASERCH DESIGN AND METHODOLOGY.....	37
3.1. PHILOSOPHICAL STANCE.....	37
3.2. QUANTITATIVE RESEARCH APPROACH	37
3.3. PHASE 1 – SOFTWARE DEVELOPMENT LIFE CYCLE	39
3.3.1. Requirements Gathering and Analysis Stage.....	39
3.3.2. Design Stage	41
3.3.3. Implementation stage	41
3.3.4. Integration and testing stage	42
3.3.5. Deployment	42

3.4. LIMITATION AND RESTRICTIONS	42
3.5. ETHICS STATEMENT	42
CHAPTER 4: SYSTEM DESIGN AND IMPLEMENTATION	43
4.1. INTRODUCTION	43
4.1.1. The Waterfall Model.....	43
4.1.2. The Agile Model.....	45
4.1.3. Application of waterfall and agile model in green crowd mobile application	
46	
4.2. CHAPTER SUMMARY	69
CHAPTER 5: SYSTEM TESTING AND RESULTS ANALYSIS	71
5.1 SYSTEM TESTING	72
5.1.1. Functional testing	72
5.1.2. Compatibility testing	78
5.1.3. Performance testing.....	80
5.1.4. Confidentiality Security Testing	93
5.1.5. User experience testing	94
5.2. CHAPTER SUMMARY	95
CHAPTER 6: SUMMARY OF FINDINGS AND DISCUSSIONS	98
6.1 CONTEXTUAL BACKGROUND FINDINGS	98
6.1.1 Green crowd mobile application (Dilivari) as a solution	98
6.2 OVERALL AIM OF THE STUDY	98
6.2.1 Research objectives	99
6.2.2. Research questions	99
6.3. DISCUSSION OF THE FINDINGS	99
6.3.1 Research question 1: How can technology be used to alleviate the triple	
bottom line challenge of people, profit, and planet?.....	99

6.3.2. Research question 2: What techniques can be used to identify, motivate and manage the crowd for the solution?	101
6.3.3. Research question 3: What algorithms are best suited at achieving optimal artificial intelligence of utilising crowd logistics?	102
6.4. RESEARCH AIM ACHIEVEMENT	107
6.5. SUMMARY FINDINGS AND RECOMMENDATIONS	107
6.6. CHAPTER SUMMARY	108
REFERENCES.....	109
APPENDICES	117
APPENDIX 1: TURNITIN REPORT	117
APPENDIX 2: ETHICS CERTIFICATE	118
APPENDIX 3: EDITING CERTIFICATE.....	121

LIST OF FIGURES

Figure 2.1:	A typical crowdsourcing implementation.	24
Figure 2.2:	Crowd logistics model components	31
Figure 2.3:	Triple Bottom Line conceptual model	35
Figure 3.1:	The different kinds of research	38
Figure 3.2:	Approach for the study	39
Figure 3.3:	Use case diagram of the online delivery system	40
Figure 3.4:	High level flow-chart diagram	41
Figure 4.1:	Waterfall model steps	44
Figure 4.2:	Agile model sample diagram	46
Figure 4.3:	Required components for obtaining location	50
Figure 4.4:	Green Dilivari architecture diagram	53
Figure 4.5:	Use case of a user requesting/offering a delivery	54
Figure 4.6:	Use case of a user offering to make a delivery	55
Figure 4.7:	Use case of the administrator of green crowd (Dilivari) mobile application	56
Figure 4.8:	Green Dilivari Class Diagram	57
Figure 4.9:	Green crowd (Dilivari) mobile application ERD diagram	58
Figure 4.10:	Login flow	59
Figure 4.11:	Delivery request flow	59
Figure 4.12:	Delivery request offer flow	60

Figure 4.13:	Preference votes from developers	65
Figure 4.14:	Data storage and insertion on MongoDB	67
Figure 4.15:	Mobile Operating System Market Share Worldwide	67
Figure 5.1:	Functional testing of the application – the map functionality showing receipts of deliveries, requests and location simulation (Maps)	73
Figure 5.2:	Simulator testing	79
Figure 5.3:	Simulator testing for registering, log-in, request and deliver	79
Figure 5.4:	User report on compatibility	80
Figure 5.5:	Sample testing on UI	82
Figure 5.6:	GET testing on Microsoft Azure	84
Figure 5.7:	POST/PUT testing	85
Figure 5.8:	Testing the UI to the API and database	86
Figure 5.9:	Testing the creation of correct collections	87
Figure 5.10:	Data sent from the UI	88
Figure 5.11:	Collection of user details	88
Figure 5.12:	Collection of package types	89
Figure 5.13:	Delivery collection	89
Figure 5.14:	Validation of status changes to user updates	90
Figure 5.15:	Microsoft Azure monitoring the database activity and the application performance	92

Figure 5.16:	Microsoft Azure monitoring and measuring the throughput of the application	92
Figure 5.17:	Testing of the log in functionality	93
Figure 5.18:	Testing email-password validation	94
Figure 5.19:	User feedback from the user experience testing	95

LIST OF TABLES

Table 1.1:	Types of crowd logistics	18
Table 4.1:	Types of mobile applications	62
Table 4.2:	Android-Flutter Comparison	64
Table 5.1:	Functional test checklist	73
Table 5.2:	Testing discoveries	96
Table 6.1:	Summary of findings	103

ACKNOWLEDGEMENTS

This study would not have been feasible without the support of many people.

It is with immense gratitude that I express my sincere appreciation to both my supervisors: Profs Mkansi and Mnkandla. To Professor Marcia Mkansi, thank you for all the guidance, drive, motivation, ideas, corrections and tireless efforts to deliver greatness on all spheres, a true nurturer. May 1 Timothy 6:20 hold you into such greatness always. Professor Ernest Mnkandla, thank you for all your time, valuable feedback and guidance. I extend my gratitude to Mr Siphiso Dzingwe, his unorthodox approach, technical prowess and great support. I have learned treasured lessons from you. Mr Mzamo Mahaeng, thank you for all your encouragements, technical know-how and support; greatly appreciated Sir. To Tshgofatso SehloDIMELA and Retha Burger, I am thankful for your efforts in providing language editing assistance.

Above all, I give God the glory for bringing me this far, *ebhen hā-'ezer!!!*

CHAPTER 1: BACKGROUND TO THE PROBLEM

The digital age has drastically altered the way in which people do their shopping from the normal in-store purchases to online purchases, whether it be from web shops or mobile shops (Odongo, 2017; Nguyen, 2019; Kafle, Zou and Lin, 2017). Consumers have a selection of all types of goods within this mode of shopping. Peng and Xu (2016) present several reasons leading consumers to shop online, ranging from buying gifts for special occasions for a timely delivery at the recipient's home to having the convenience of ordering groceries and fresh food delivered at home (Wicaksono, 2018). Odongo (2017) also observed that clothing and footwear sales dominate online volumes, exhibiting growth metrics of 50% between 2013 and 2018; these numbers are continuing unabated.

According to Peng and Xu (2016), this new wave of shopping has resulted in businesses failing to meet the exponential demand for decentralised deliveries as consumers purchase online from dispersed locations. The demand for quick and effective delivery presents a challenge impacting the triple bottom line for most businesses engaged in last mile distribution. The type of challenges include the provision of containers and vehicles of different sizes, logistics around scheduling deliveries, the availability of drivers and the transportation of cargo (Buldeo Rai, 2019).

Furthermore, Kafle, Zou and Lin (2017) elaborate that the increased number of delivery trucks have a negative effect on the urban environment causing traffic congestion, increased carbon emissions, wear and tear of road infrastructure and parking restrictions and lack of parking spots during week days. In addition, goods in transit vehicles contribute to about 50% of the total vehicle emissions in urban areas while also accumulating significant parking penalties (Kafle, Zou and Lin 2017). A study conducted by the European Union supports this point by stating challenges such as transport congestion, motor vehicle accidents, noise pollution, infrastructure wear and tear, and a contribution of 23.2% greenhouse gas emissions in the year 2014 (Buldeo Rai, 2019). These challenges are bound to increase with the growing demand for deliveries.

The online shopping footprint continues to grow. Nguyen (2019) contends that customers' expectations of door-step deliveries necessitate flexible delivery solutions that will cater for frequent orders and fast deliveries (Peng and Xu, 2016). Numerous companies are experimenting with various solutions in a bid to meet customers' satisfaction and expectations as well as to gain a competitive advantage in terms of deliveries (Odongo, 2017; Nguyen, 2019; Kafle, Zou and Lin, 2017).

Several solutions have been implemented in trying to solve the problem such as outsourcing the last mile delivery to specialist companies (Serafini, Nigro, Gatta, Marcucci, 2018); however, this is often expensive and reduces competitiveness for small and medium enterprises (SMEs) (Buldeo Rai, 2019). Large companies have increased their delivery fleet to handle the rapid demand, but this too is expensive to maintain due to the added costs of labour and fuel (Brotcorne et al., 2019). The challenge is, therefore, balancing frequent deliveries by increasing the fleet, whilst also reducing the number of fleets required for a rapid response to demand orders. The increase in fleet inadvertently leads to a higher fuel consumption that contributes negatively to the environment and greenhouse gas emissions (Kafle, Zou & Lin, 2017). An increased delivery fleet adds to traffic congestion and the already limited parking space available in cities. As such, several industries and scholars have considered alternative solutions that are mostly technology based, such as the use of drones and humanoid robots (Odongo, 2017; Buldeo Rai, 2019); however, these solutions still fail to cater for all package sizes and locations as well as at times coming into conflict with regulative bodies. There is limited evidence on the possible solutions that offer a balanced distribution approach of the triple bottom line.

Therefore, this research explores a digitally enabled distribution model that addresses and implements a triple bottom line approach to the last mile distribution, using technology, information systems and algorithms. In doing so, the study begins with a literature review of existing solutions towards achieving a triple bottom line and how it can respond to the growing demand for effective on-time deliveries in order to solve current issues.

1.1 A REVIEW OF EXISTING SOLUTIONS TOWARDS THE TRIPLE BOTTOM LINE

The delivery of goods from business to consumers is a universal problem. As such, numerous studies have been conducted to tackle this problem and have offered various solutions such as crowd logistics as a response to consumers' delivery preferences while also reducing the business costs (Buldeo Rai, 2019). Crowd logistics is the use of people – “the crowd” – to undertake a task or an activity (Peng and Xu, 2016), which in this case would be making deliveries. For most SMEs, implementing crowdsourcing is challenging because it requires tracking of each individual of the crowd (Buldeo-Rai, 2019). In contrast, Odongo (2017) asserts that big businesses such as Amazon can implement an efficient delivery network. A crowd network for deliveries is time consuming and costly even for big businesses (Nguyen, 2019; Purakala, 2020; Wicaksono, 2018). Costly because significant budget allocation must be invested into personnel and delivery vehicles that must also be maintained, adding more costs to the business. The addition of more vehicles presents an environmental hazard coming with increased disadvantages instead. In this light, the concept of sustainability in assessing environmental, economic and social performance has led to scholars taking interest in reviewing several solutions as a means to reach a triple bottom line solution (Ahi and Searcy, 2015).

1.1.1 User's freight

Amazon has utilised crowdsourcing by getting people with discretionary time to collect and deliver parcels to consumers (Serafini, Nigro, Gatta, Marcucci, 2018). In this instance, a driver must register and is allocated a shift where they would make collections from a nearby store or warehouse and deliver to consumers (Chopra, 2018). Although this solution adds significant value in on-time deliveries, it increases the numbers of vehicles on the road and adversely impacts traffic congestion and the emission of greenhouse gases. According to Chopra (2018), this solution comes with expensive costs to customers and is yet to be implemented profitably. Also, this service is only available for a limited number of products.

1.1.2 Outsourcing the last mile

Numerous businesses have opted to utilising a company that specialises in distribution logistics to take care of their deliveries such as PostNet and DHL (Giret, Carrascosa,

Julian, Rebollo, Botti, 2018). While such a solution of outsourcing deliveries seems ideal, it does however extend the last mile delivery time and tends to be expensive for either the business or consumer (Buldeo Rai, 2019). Again, presenting a challenge to the guaranteed on-time delivery promises to customers.

1.1.3 Drones and robotics

Odongo (2017) explains that first-world countries have started experimenting with the use of drones for delivering packages within stipulated time frames, with big companies such as Amazon driving such innovations. Drones with their aerial presence help in reducing traffic congestion and emissions. They also become very useful in ensuring deliveries to those hard-to-reach places. A disadvantage of drones, as Odongo (2017) highlights, is that they can only load small-sized parcels at a time. Additionally, regulatory authorities tend to have restrictions on the use of drones. A resolution to these regulatory restrictions may present future opportunities.

The use of robots offers an alternative to drones. Robots fitted with cameras and GPS-enabled sensors to navigate the streets and buildings without the aid of humans could also be employed (Odongo, 2017; Buldeo Rai, 2019). Although robots are a good advancement in the technology space, they have limitations of size, speed and regulatory body boundaries. Nonetheless, a negative impact of robots is job insecurity for the workforce (Wisskirchen, Thibault, Ulrich, Annemarie, Gunda, Guillermo, Soler, von Brauchitsch, 2017).

1.1.4 Rail strategies

An observable trend shows a significant shift from conventional delivery vehicles to generic modes of transport such as trains, cargo bikes and buses with additional storage capacity being used to move deliveries (Odongo 2017). Though cargo bikes tend to be quick, especially for on-time deliveries, they can only do so within a small area failing when needed to cover great distances (Binetti, Caggiani, Camporeale, Ottomanelli, 2019). Notwithstanding, this trend presents safety issues such as package theft and goods arriving in dilapidated conditions. Moreover, Buldeo-Rai (2019) has noted a shift away from the normal delivery hours into the so-called “off-hour deliveries” where deliveries are made at night, early in the morning or late evening so as to avoid traffic congestion, reduce fuel consumption and minimise

environmentally harmful emissions. Although this method comes with great savings, it creates noise pollution at very inconvenient times (Peng and Xu, 2016; Buldeo Rai, 2019) and consumers unobliging deliveries at such times.

1.1.5 Click and collect

Retailers have presented customers with an option where they can opt for either a home delivery or package collection from a nearby warehouse. Takealot is the market leader in this space (Odongo, 2017). With this approach, consumers are still made to wait for their products to be delivered to their desired location. For the few who opt in to collect at the warehouse, this does not alleviate the challenges of increased vehicles on the road. In other spaces, retailers have introduced pickup/smart lockers. Regardless of all these options, the majority of consumers still want home deliveries (Odongo, 2017). And the challenge of triple bottom line persists.

Looking into the future, it is expected that advances in technology will intervene with sophisticated drones, hybrid airships, hovercrafts and pipelines; however, the implementation thereof is unclear given the rate of development. Buldeo Rai (2019) pre-empted big data as a potential component in providing an optimised solution for an efficient crowdsourcing solution.

While several proposed solutions address the delivery problem, achieving the triple bottom line persists in the last mile distribution rendering the available solutions inefficient and ineffective. Special holidays such as year-end holidays, Black Friday (US) (Paper 2018) and Singles Days (China) exacerbate the challenge. If big companies with established logistics systems are not immune to the challenges presented by special days, it becomes egregiously problematic for SMEs and highlights the need for an off-the-shelf logistics product (Buldeo Rai, 2019). Random clustered or dispersed geographical location purchases make deliveries costly, more especially on such days. Countries such as South Africa that have townships and rural areas with unstructured infrastructure and geographical dispersed households are adversely impacted on such days (Mkansi, 2020).

1.1.6 Cyclists and pedestrians

An understudied opportunity is the application of crowdsourcing on cyclists and pedestrians. This option tends to have great manoeuvrability, it comes at a low cost

and is environmentally friendly (Maesa, Vanelslandera, 2012). The only setback is that cyclists and pedestrians are only able to cover short distances, necessitating a linkage with trucks near customers (Kafle, Zou and Lin, 2017). This means that a truck would collect from the warehouse and drop-off goods at locations reachable by pedestrians and cyclists to enable deliveries. Perishable products are also cumbersome in that deliveries must be made within specific timeframes so that they do not spoil (Mkansi et al., 2018). Buldeo Rai (2019) proposed a solution for low volume deliveries to maintain the freshness state. Beyond that, there is still a challenge of having a shortage of drivers as this solution requires added personnel. Most importantly, the solution is a step towards people and the planet, but not much on profit, requiring a more holistic solution.

Peng and Xu (2016) assert that should a solution be successfully implemented, it would result in fewer delivery trips, modal shift, a reduction in delivery distances and increased efficiencies. Odongo (2017) adds that a successful implementation of crowdsourcing may also result in the reduction of pollutants emitted by delivery trucks. The following section offers a critical review of the existing crowd logistics solutions.

1.2 BRIEF OVERVIEW OF CROWD LOGISTICS RESEARCH

Peng and Xu (2016) describe “crowdsourcing” as assigning tasks to be accomplished by an uncertain group. In general, crowdsourcing can be implemented for either Business-to-Business (B2B) or Business-to-Consumer (B2C). The latter is the focus of this study where the “crowd” would be any group of people who would be willing to take on the task of making the last mile delivery (Chen, Mes, Schutten, 2018). There are a several national and international types of crowd logistics from passengers, storage, goods, and shipping among others. Goods crowd logistics form part of B2C and are responsible for delivering packaged goods and fast foods. While storage crowd logistics are achieved by renting storage from basements, rooms, and even houses for bread and breakfast businesses. Airbnb is an example where hosts would register their estates so that the rooms would be used to offer the hotel experience at a cheaper price (Nguyen, 2019). Warehouses can also form part of storage crowd logistics (Jeremić, Andrejić, 2019). The table below provides a summary of the different types of crowd logistics. Whilst there is a high number of crowd solutions, research reveals that none is able to achieve the triple bottom. The challenge is

exacerbated global pandemics such as Covid-19 and the rise in online shopping, which proves more favourable for working populations.

Table 1.1: Types of crowd logistics

Types	Description	Example
Passenger	Responsible for transporting people from one location to another	Uber, Taxify
Local delivery service	Utilises motor vehicles, motorcycles, bicycles or people walking	Amazon crowd logistics, DHL
Goods	Delivery of packaged goods to clients	EOM, PostNet, Ubereats
Storage	Achieved by renting storage from multiple individuals such as a basement, a garage, or rooms	Airbnb, Takealot
Shipping	Use of a platform to purchase goods from abroad and getting a person from a platform to be the carrier	Maritime Carrier Services, Morgan Cargo air and sea freight logistics

Peng and Xu (2016) observed that with the exponential growth in online shopping, the demand for deliveries exceeds the available delivery transport and personnel causing delayed deliveries and low customer satisfaction and retention. Nguyen (2019) concurs that most online consumers expect door-step deliveries. Though low delivery prices contribute positively to customer satisfaction and retention, many sources state that it is expensive in terms of cost and time (Todorovic et al., 2018; Brotcorne et al., 2019). Brotcorne et al. (2019) elaborates that these costs include, but are not limited to, fuel cost, packaging, personnel costs, acquisition and maintenance of vehicles as well as the ICT systems. Deliveries have also become a challenge as consumers are always expecting to pay only a fraction of the delivery amount and are more often happy to settle for a later delivery date provided they get a free delivery (Buldeo-Rai, 2019); hence the need for a triple bottom line green crowd logistics.

Crowdsourcing logistics faces risks of the goods being stolen, lost and/or damaged and liability (Ducarme, 2019). As there are limited regulations on the carriage of goods, it has presented a window of opportunity for insurers to conduct business on goods

(Odongo, 2017), which reiterates the need for an efficient green crowdsourcing carrier service system especially an off-the-shelf solution to assist SMEs.

1.2.1 Drivers of the last mile distribution demand

Predictions, projections and forecasts indicate that online shopping will continue to grow (Buldeo Rai, 2019; Chen, Mes and Schutten, 2018; van Cooten, 2016). With this knowledge in hand, it is evident that failure to secure an effective and efficient solution would lead to increased traffic congestion resulting in elevated greenhouse gas emissions (Odongo, 2017), causing ozone layer depletion, noise pollution, car accidents and associated carcinogenic and respiratory health problems (Wicaksono, 2018). It has then become an interesting field for scholars to investigate solutions to this problem.

The cost of distribution and delivery of goods is of great concern for business. Although the figures are a trade secret, Buldeo-Rai (2019) reveals that in 2017, online sales contributed 10.2% of global retail sales, a figure that is forecasted at 17.5% by the year 2021. A paper by Deloitte (2022) supports these projections and further gives the exponential increase of online sales due to the COVID pandemic; one such example is the business Home Depot which achieved a 86% increase during the year 2020. These figures cover both physical products and non-physical products and services such as event tickets, which require transport to deliver. These increases have financial implications for business such as the cost of labour, assets, equipment and maintenance for last mile deliveries (Buldeo Rai 2019). The distribution cost of returns, commonly known as reverse logistics, is another financial issue business has to contend with (Purakala, 2020). Returning faulty products to an online retailers' distribution centre presents a layer of challenges related to profit, people and the planet. Customers expect hassle-free forward and reverse service, at no additional costs to the order. Evidently, providing such a service comes at a cost to the service provider, with financial repercussions in profits (Buldeo Rai, 2019). Odongo (2017) accedes that most businesses acknowledge low delivery prices as a selling point to consumers.

In an attempt to balance the profit, people and planet of the last mile distribution, several studies (Frehe, Mehmman and Teuteberg, 2017) recommend crowd logistics.

However, the existing crowdsourcing model adds more cars to the road, which ultimately increases congestion and transport carbon emissions (Ducarme, 2019). Notwithstanding, we need to understand green crowdsourcing.

1.3 PROBLEM STATEMENT

Customer attraction and retention are critical in ensuring that online businesses gain a competitive advantage (Journal, 2020). Customers expect and consider product deliveries as a norm (Wicaksono, 2018) leading online retailers to invest in on-time delivery solutions as a means to gain customers and advertise their offering.

Numerous studies have been undertaken exploring different solutions from increasing delivery personnel, vehicles, robots or drones to outsourcing deliveries, but they fall short with reasons ranging from being too expensive (Chen, Mes and Schutten, 2018), causing environmental harmful or even failing to meet timely deliveries (Arslan *et al.*, 2016). As such, a more holistic and sustainable solution is required.

The need to develop a sustainable solution is necessitated by the strengthened e-commerce growth and its economic, social and environmental effect (Ducarme, 2019). The last mile is often considered the most expensive on the shipping framework posing a financial challenge in terms of costs. In addition, it requires a number of personnel to execute effectively while the freight negatively impacts the environment due to the carbon emissions (Binetti *et al.*, 2019).

Supply chain businesses are challenged with balancing the triple bottom line of profit, people and the planet especially where the last mile is concerned. In an attempt to respond to the challenge, scholars and industries considered several options including crowdsourcing (Koch *et al.*, 2013; Hultberg, 2016). However, existing crowd logistics models add more cars on the road and contribute heavily to green gas emissions, while also being expensive due to vehicle costs, maintenance, and fuel. This study, therefore, proposes a digitally enabled green crowd logistics that can reduce the overall number of vehicles on the streets, which is crucial in achieving the triple bottom line in the last mile distribution.

1.4 RESEARCH AIM

This study aims to use a combination of technologies, algorithms and artificial intelligence to develop a digitally enabled green crowd logistics model for achieving the triple bottom line (people, profit, and planet) at the last mile distribution.

1.5 RESEARCH OBJECTIVES

This study is intended to achieve the following objectives:

1. Explore how the use of different technologies can contribute to alleviate the triple bottom line challenge of people, profit, and planet.
2. Create a database to facilitate the management of crowd data.
3. Compute an algorithm that uses artificial intelligence to integrate technologies that map and identify the crowd to retailers or businesses.

1.6 RESEARCH QUESTIONS

The following research questions will be asked in order to not only address the aim, but to also achieve the stated objectives:

1. How can technology be used to alleviate the triple bottom line challenge of people, profit, and planet?
2. What techniques can be used to identify, motivate and manage the crowd for the solution?
3. What algorithms are best suited to achieving an optimal artificial intelligence of utilising crowd logistics?

1.7 SIGNIFICANCE OF STUDY

Studies by scholars and logistics companies such as DHL reiterate that advances in information and communication technologies are yielding better results as compared to conventional logistics methods. Hence, our primary objective is to blend technology and artificial intelligence for last mile deliveries to bring forth a sustainable and environmentally friendly, cost effective and positive solution for the people.

The United Nations (UN) anticipate steady growth in the number of people living in urban areas. A matter of concern for the UN, however, is that the majority of people engaged in online shopping are residents of these urban areas and a further migratory increase will inadvertently increase carbon emissions in delivery motor vehicles (Sampaio *et al.*, 2017). Moreover, this population growth comes with the expectation of timely deliveries, which echoes this study's intention to provide a sustainable solution for the environment and on-time delivery.

1.8 THE SCOPE AND LIMITATIONS OF THE STUDY

Home deliveries have been around for decades, but their significance had not garnered enough interest from scholars until the exponential explosion on online shopping which has only been around for approximately 20 years (Ducarme, 2019). The paucity of research may be a limitation in terms of providing an in-depth understanding of the key factors involved in implementing a digital last mile delivery solution. In contrast, the aforementioned may be rivalled by the abrupt studies done over the last ten years on last mile deliveries and sustainable solutions around it.

Additionally, the study does not cover the entire software development life cycle, which may be perceived as a limitation. Due to constraints in time and based on findings from the literature review, the study is only limited to development which may inadvertently limit testing to functionality without covering the impact on the profit, people and planet. Nevertheless, the study offers a digital solution as a platform for measuring impact as a stepping stone for future studies.

CHAPTER 2: LITERATURE REVIEW

The previous chapter discussed the need to balance the profit, people and planet in relation to last mile distribution and the necessity for a digital solution to address the imbalances. This was introduced as a motivation and fuel for this study. This chapter presents a comprehensive literature review of the crowdsourcing distribution model. A discussion on the two theories – coordination theory and information systems theory – that serve as a lens for this study follows thereafter. The first theory provides the lens to coordinate the triple bottom line while the information system theory is used to develop digital solution options required to redress the imbalances. Following the theoretical discussion is a review of the triple bottom line and the components of crowdsourcing. The conceptual framework paves a way to concluding the literature review section.

2.1 BACKGROUND REVIEW OF CROWDSOURCING DISTRIBUTION MODEL

Rougès and Montreuil (2014) explains crowdsourcing as a type of participative activity where individuals, or a company, proposes to a group of individuals via a flexible open call to voluntarily undertake a task. A crowdsourcing distribution model informs the building blocks for implementing a crowdsourcing solution. In recent years, crowdsourcing distribution models have become more prevalent due to the rise of mobile phones and internet-driven technology. For this reason, different forms of crowdsourcing distribution models have been developed and exist in various technology-mediated platforms (Chaves et al., 2019).

Examples of crowdsourcing applications that have emerged are Flexe, MyWays, and Checkrobin (Mladenow, Bauer and Strauss, 2015). Checkrobin is designed as a consumer-to-consumer (C2C) model linking consumers to ride share for private shipments. MyWays is an example of a delivery service model by DHL, where crowd sourcers may either specify parcels to be delivered to a destination or the parcels they are willing to collect and deliver. On the other hand, Flexe incorporates both the demand and supply of warehouse space, which allows customers to book space at a predefined location for certain period of time based on a specific requirement. All these

technology crowdsourcing platforms were designed to inspire public participation, to provide opportunities for both people and profit, but not the environment. Hence, the need for a green crowdsourcing distribution model that offers a balance of the triple bottom line.

A typical crowdsource delivery system, as depicted in Figure 1, explains how a company that was used for crowdsourcing research by Peng and Xu (2016) implemented crowdsourcing. In support of this approach Kafle, Zou and Lin (2017) proposed a similar two-step approach. The first step selected the crowdsources to make the delivery, and the second step selected the pick-up points, route and schedule.



Figure 2.1: A typical crowdsourcing implementation. Source: Peng and Xu (2016)

There are many reasons advocating for crowdsourcing as a possible solution to the issue of implementing an effective delivery system. Peng and Xu (2016) elaborate on the advantages of crowdsourcing by exploring work from other scholars. Among the advantages are the financial benefits (profit) as the crowd would replace customer

service employees similar to outsourcing and the effectiveness of crowdsourcing especially when there is limited time to carry out a task. Marcucci, Delle, Pompetti & Gatta (2018) augment the advantages by elaborating on the economic and financial aspects therein. They aver that the use of in-store customers making deliveries for online shoppers, combining shipping with community member's trips, and having occasional drivers reduces costs as compared to a professional delivery fleet (Gatta et al., 2018).

Whilst Ronghe et al. (2016), Gatta et al. (2018), and Peng and Xu (2016) provide a good understanding and platform for a typical crowdsourcing, there are, however, several challenges that present research gaps and development opportunities within security and legal liability, lack of crowd control, and carbon emissions racked during distribution. For example, Peng and Xu (2016) highlight cargo security and legal liability, and further elucidate that the minimal credit information, or the lack thereof, poses a threat to eliminating the risk. Moreover, a company has minimal-to-no control on the crowd. Therefore, Peng and Xu (2016) found that running police background checks and acquiring participant ID details assisted in the selection and recruitment of a trustworthy crowd. Additionally, Brown and Guiffrida (2014) elaborate on the well-documented issues of greenhouse gas emitted during deliveries in that the burnt carbon-based fuel was harmful to the environment and demanded focus from organisations regarding the environmental impact of their business operations.

From the challenges, this study focuses on the issue of carbon emission due to the magnitude of its impact on the environment and people as well as the sustainability of production and supply chain. This focus is an attempt to balance the triple bottom line. Using the coordination theory and information system theory lens, the study explored the different aspects in which the people, profit, and planet could be better coordinated using mobile technologies.

2.2 SUPPLY CHAIN COORDINATION THEORY

Over the years, businesses have transitioned from competing on price to selling similar goods; the focus has shifted to businesses investing more on customer retention by seeking to understand customer needs. Last mile deliveries have inspired businesses to invest in their supply chain with the intention of achieving more effective logistics

systems so as to optimise the time and cost in reaching customer deliveries (Gay and Norrman, 2016). Matheu (2011:) explains supply chain as organisations, people, technology, activities, information and resources that form a network being involved together, through upstream and downstream connections, all having processes that produce value in the form of products or services in the hands of the end consumer. The ultimate goal is meeting the needs of the consumer (Engineering, 2011), which is making on-time deliveries.

According to Simatupang, Wright and Sridharan (2002), “coordination” is defined as the act of managing interdependencies among activities done to achieve an objective. Within the context of last mile deliveries, coordination can be seen as adequately combining, adjusting and aligning information, knowledge, systems, people and funds for the achievement of making on-time deliveries. An effective coordination system is key in gaining competitive advantage by means of coordinating the rate at which products are prepared and delivered on time to meet the rate of customer consumption leading to customer satisfaction. Tracking of customer behaviour in terms of the products they purchase, when and where; tracking the rate at which products need to be delivered to stores or consumers; tracking of time frame expectations of deliveries by consumers, and the affective movement from suppliers may all be achieved by effective coordination (Simatupang, Wright and Sridharan, 2002). In addition, customer delivery timelines are met with ease, products are available to consumers in a timely manner, and reduced logistics costs as well as lower inventory costs are all achievable. The application of consideration should consider these four main aspects: logistics synchronisation, information sharing, incentive alignment and collective learning.

2.2.1 Logistics synchronisation

Logistics synchronisation entails creating value by understanding customer needs, products, inventory, suppliers and delivery. This analysis helps understand customer needs and behaviours in terms of when they make orders or purchases to achieve a balance between supply and demand.

The intention behind this synchronisation is to leverage key activities that present the most value to customers and then supplementing them with less essential activities to ensure an optimal chain performance (Holweg and Pil, 2001).

2.2.2 Information sharing

Information sharing is key for all stakeholders involved, whether they are suppliers or carriers to the customers. Gaining in-depth understanding of all stakeholder needs aids in developing patterns that may increase effectiveness and efficiency based on the information obtained, thus creating a loop of continuous improvement (Chankov, Becker and Windt, 2014).

2.2.3 Incentive alignment

Incentive alignments are key in measuring performance such that good performance may be rewarded accordingly or undertake adequate recourse for unsatisfactory performance such as instituting relevant punitive measures (Simatupang, Sandroto and Lubis, 2004).

2.2.4 Collective learning

Collective learning is termed as a continuous process for all involved as a means of understanding the components involved and determining ways to continuously improve to maintain and keep abreast of changing customer demands (Simatupang, Wright and Sridharan, 2002).

These abovementioned aspects enable one to employ the supply chain coordination theory to crowd logistics to coordinate the triple bottom line.

The following section reviews the information system theory that will aid in developing a digital solution that integrates the aspects of coordination towards the desired balance of the triple bottom line.

2.3. INFORMATION SYSTEMS THEORY

Literature on the last mile is yet to advance an established, applicable theory (Olsson, Hellström and Pålsson, 2019). Though there are many applicable theories presenting varying perspectives on last mile deliveries, the information system theory (IST) was

most appropriate. This theory comprises systems understanding and the collection of data so that it may be transformed from data without substance to models and useful forms of computer programs. System quality and information quality are impactful on both the use and user satisfaction (Halawi & McCarthy, 2006). Lerner (2004) strengthens the case for the applicability of the information systems theory by explaining it as building a bridge between information technology and the general systems theory's formalism. Combining data and information technologies with models such as computer algorithms assists in transforming data into information and developing complex systems.

In crowd logistics, intense understanding of the crowd, customers, routes, traffic, days, pricing, and different technologies is necessary. Lastly, the crowdsourced logistics must be understood as a system to optimise efficiency. While this theory is very applicable, an exploration of other theories presents a window of opportunity as there is still a lot of ground to be covered on contributions to the growing field.

2.4 TRIPLE BOTTOM LINE

The drive for organisations to strive for sustainability is frequently based along three pillars: economic, environmental, and social development (Brown and Guiffrida, 2014). These pillars form the “triple bottom line”. Any scenario involving online shopping and deliveries, the focus is always on three main components: a buyer or customer, the seller or company responsible for delivering the purchased goods, and the environment impacted by the process of delivering to the customer. This sub-section of the study is aimed at understanding the triple bottom line – the social, environmental and financial aspect of last mile deliveries.

2.4.1 People

Crowdsourcing is a word derived from two words, “crowd” and “outsourcing” (Frehe, Mehmman and Teuteberg, 2017). A “crowd” is several people, and “outsourcing” is the exercise of securing a third party out of the employ to perform the duties, tasks or functions that were performed by internal employees.

In last mile logistics, people are a very important component because it is everyday people who undertake the role of carriers while earning from it. Because of advances

in technology and globalisation, people are now remotely connected via the Internet (Koch *et al.*, 2013), making it possible to easily implement crowdsource logistics for last mile deliveries.

Regarding the social aspect, people are the driving force as the initiators, executors and owners of the entire system around last mile deliveries. A person would initiate the process by going online via a mobile phone or computer connected on the Internet to start shopping online. After an electronic payment and the delivery date stipulated, the shop or company would package the purchased goods for shipment to the buyer's stipulated address (Buldeo Rai, 2019). The exponential growth of online shopping over the past few years has seen retailers fail to meet the demand for on-time deliveries (Devari, 2016), hence a strong case for a solution to obtain customer satisfaction and competitive advantage from other sellers.

2.4.2 Profit

The financial aspect is important for both customers and businesses. Studies place emphasis on the drive for profitability as the e-commerce surge continues to grow (Scherr *et al.*, 2018). Furthermore, financial structures of different business models highlight the costs related to fleet involved in last mile deliveries (repairs, service, fines, parking costs, fuel among others) and the social costs as having a significant impact on profits (Brotcorne *et al.*, 2019). Because of these costs, customers are negatively impacted as they have to absorb the impact on the delivery prices.

On a study done by Wicaksono (2018) on the use of crowdsourcing using bicycles for making deliveries, it was observed that there were significant savings on delivery costs, speed, timeframes, performance rating by customers and CO₂ emission savings. The only setbacks from this mode of deliveries were size limitations on the package weight, distances covered could only be to a certain range, and the number of personnel employed would increase. Though bicycles take into consideration carbon emissions, they were unfavourable for retailers due to size limitations (Germany, 2012). Hence the observed gap of a crowdsourcing digital solution that could achieve the benefits highlighted with no limitation of size as observed by Wicaksono (2018).

2.4.3 Planet

The cost of the motor-vehicles dedicated to making deliveries is expensive on the environment due to greenhouse gases emissions such as nitric oxide, carbon monoxide and carbon dioxide (Gatta et al., 2018). The impact of these gases is well documented with a number of negative effects on the environment such as global warming, rises on sea levels and the negative impact on agriculture and sea life (Latake and Pawar, 2015). This is a cry for change, innovation or enhancements on the current state and ways of conducting business because e-commerce is showing no signs of slowing down. This study seeks to address the research gap. Below is a discussion of the existing crowdsourcing logistics model.

2.5 CROWDSOURCING LOGISTICS MODEL COMPONENTS

Crowd logistics has transformed last mile deliveries (Nguyen, 2019) by significantly improving effectiveness in terms of costs (profit) and promptness yet failing to balance the environmental impact. Figure 2.2 is an example of crowd logistics showing the five main components: creation, offers, stakeholders, characters, and revenue model. This study responds to the notable mission component of the environment within the crowdsourcing logistics model. Prior to the re-conceptualisation of the model, the different components are discussed.

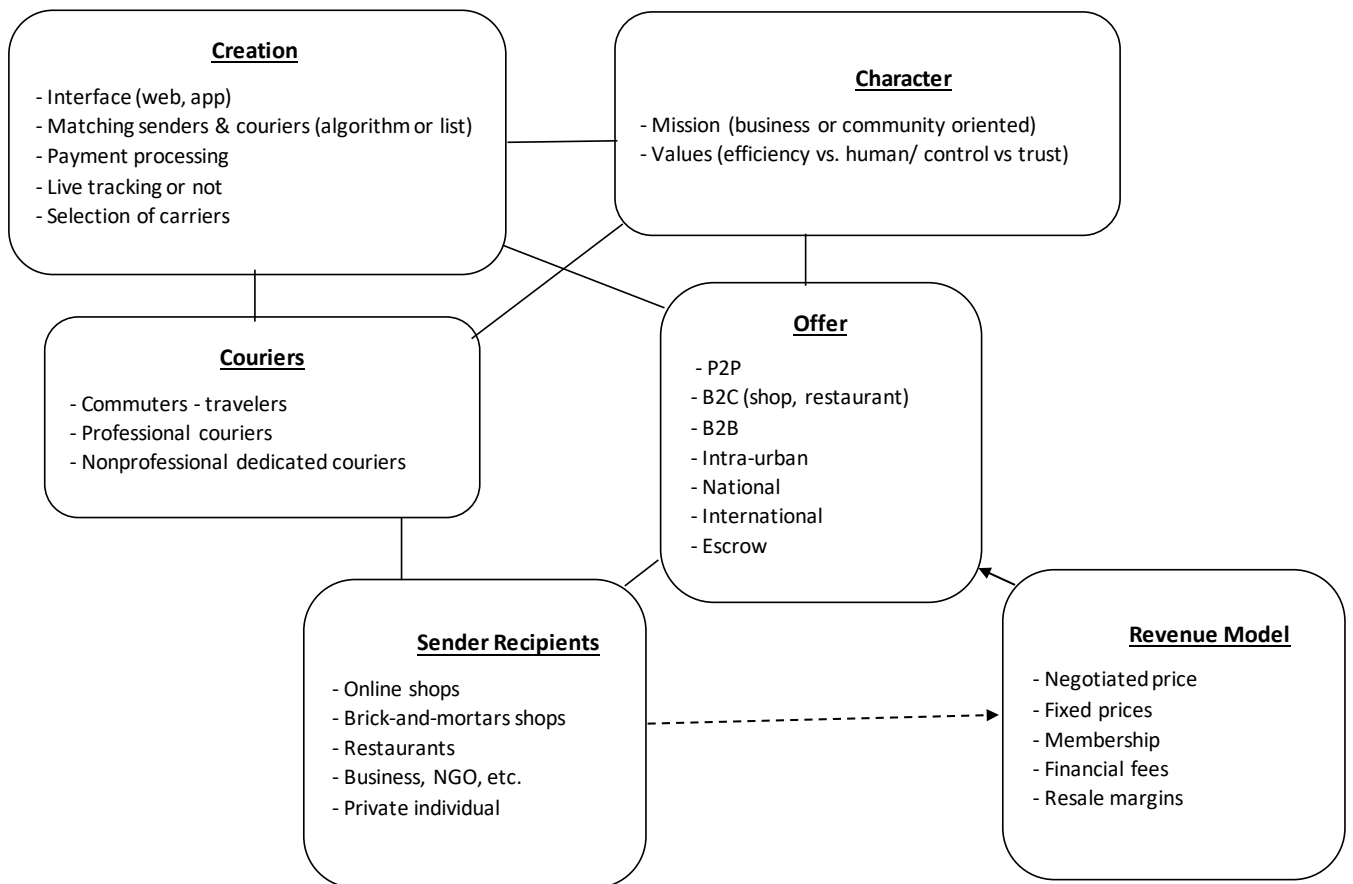


Figure 2.2: Crowd logistics model components. Source: (Rougès and Montreuil, 2014)

2.5.1 Creation

The process of “creation” is the starting point of the logistics problem. A user or customer (people) would specify a delivery that needs to be made via a website or a mobile application. After a courier (section 2.3.2) has been identified and matched to the user using algorithms, the internal logic of the application calculates the distance, based on the route to determine the price and the estimated delivery time. The price and estimated delivery time would vary based on the time of the day – whether peak or not, a public holiday or a special day, or festive season or the weekend. A courier company would continuously contact the user until the package is delivered. Package tracking is also a feature of the proposed application.

2.5.2 Couriers

The use of couriers is another component within crowd logistics. It comes in many different forms as discussed in section 1. Dedicated professional couriers offer business-to-business (B2B) transactions for making deliveries on behalf of retail

businesses to clients/customers expecting deliveries. This service comes at a high cost and becomes unaffordable especially for small shops. Another issue is that the volume of packages to be delivered varies (Maes and Vanelslander, 2012); hence the need for a flexible solution. The use of dedicated couriers increases the negative environmental impact of carbon emissions because more cars are being used solely for deliveries as the demand increases. Furthermore, couriers have to account for fleet maintenance costs. The proposed crowd logistic solution offers such convenience of re-using willing customers travelling in the same direction.

Crowdsourcing couriers include cyclists, dedicated drivers, and cargo bikes for courier companies such as DHL. The weakness faced by current couriers is the additional cars dedicated solely to deliveries, which ultimately, increase carbon emissions. An observed gap is the exclusion of traffic already on the road as couriers. Making use of the space on the crowd which is already on the move, would decrease retail store deliveries and their dedicated carriers, and reduce environmental greenhouse gas emissions thus positively impacting the next aspect on the triple bottom line – cutting costs to increase profits.

2.5.3 Senders and recipients

The sender's component involves businesses offering services or selling products that the customers purchase online. Senders range from clothing stores and restaurant to large e-commerce companies such as Amazon and Takealot selling all types of goods. Recipients would be customers who purchased from senders, clients receiving packages from businesses offering a service, or any recipient of a package.

2.5.4 Offers

The offers component in the application presents to the customer the opportunity to select crowdsourced delivery as an option when checking out purchased goods. For this to be effective, it requires a partnership between the application and business or multiple businesses to increase the chances of customers buying and requiring crowdsourced delivery for the last mile delivery. The package would then be matched to a person travelling in the same direction as the intended delivery, or willing to make the delivery, with the details already available such as the address (Rougès and

Montreuil, 2014). This same concept applies whether one is buying a gift for someone else or another company is sending a package to a client or third party.

2.5.5 Revenue model

The revenue model is an algorithm of receiving a reasonable price for the task at hand. This is a process of computing the delivery price based on fixed variables such as package size. The heavier the package, the more expensive. Unfixed variables such as the distance to be covered, the day of the week and time of the day, special occasions such as black Friday (Paper, 2018) and festive season are also computed into the delivery price that would be reasonable to the crowd courier, the purchasing customer and the shop. An advantage to the aspect of profits is cost savings. However, using the crowd is insufficient; it is quite crucial that the crowd remains motivated to garner these positive outputs. Chaves *et al.*, (2019) touch on the performance of the crowd, stating it as a relationship between the task at hand and payment based on performance.

2.5.6 Character

Rougès and Montreuil (2014) describe character as being a defining factor for the drive of the company, whether business or community orientated. The former entails profit making, efficiency and effectiveness at attaining customer satisfaction and experience. The latter relates to building customer relations, maintaining a relationship of trust and creating unique and special customer experiences.

2.6 CONCEPTUAL MODEL

A conceptual model is merely the representation of a system using concepts that would generate understanding of a system. Figure 2.3 illustrates a conceptual model that incorporates all components derived from Rouges and Montreuil (2014) and includes an additional component for the purposes of this study. This added component, the planet, addresses carbon emissions, noise and traffic reduction.

Furthermore, this study amends the couriers' component from that presented by Rouges and Montreuli to include people who are already on the move to reduce costs and impact sustainability and environmental preservation, the last aspect of the triple bottom line. On recruiting the crowd, a popular technique is stated preference (SP),

which is key in understanding perceptions, acceptance and reactions related to satisfactions and acceptance of innovative services (Gatta *et al.*, 2018). This is important for this study because it would help build momentum for the crowd. The proposed crowd logistics offers a change in a positive direction and acts as a stepping stone for future studies to measure improvements as compared to traditional modes of making deliveries. True “people, planet and profit” oriented logistics contribute not only to gaining competitiveness amongst businesses and customer satisfaction but in meeting environmental policy targets as well.

The different components of the conceptual model were discussed in the previous section except for planet. In relation to the planet, the issue of CO₂ emissions is of concern with last mile deliveries as the main contributors being compared to other segments of the supply chain (Awwad, Shekhar and Iyer, 2018). The negative impact of carbon emission is felt globally, causing concerns regarding the environment; thus strengthening the case for innovation with respect to making the last mile segment sustainable. Other negative impacts include noise (Gatta *et al.*, 2018) and traffic congestion (Chen, Mes and Schutten, 2018). The rapid increase in customer demand translates to more motor vehicles moving around making deliveries, while affecting civilians who need silence to sleep and study, and general silence for peace of mind. Additionally, this causes frustration in terms of time delays for those travelling on the roads. This is a growing concern as more people are now moving into or living in urban areas (‘World Urbanization Prospects : The 2018 Revision’, 2018).

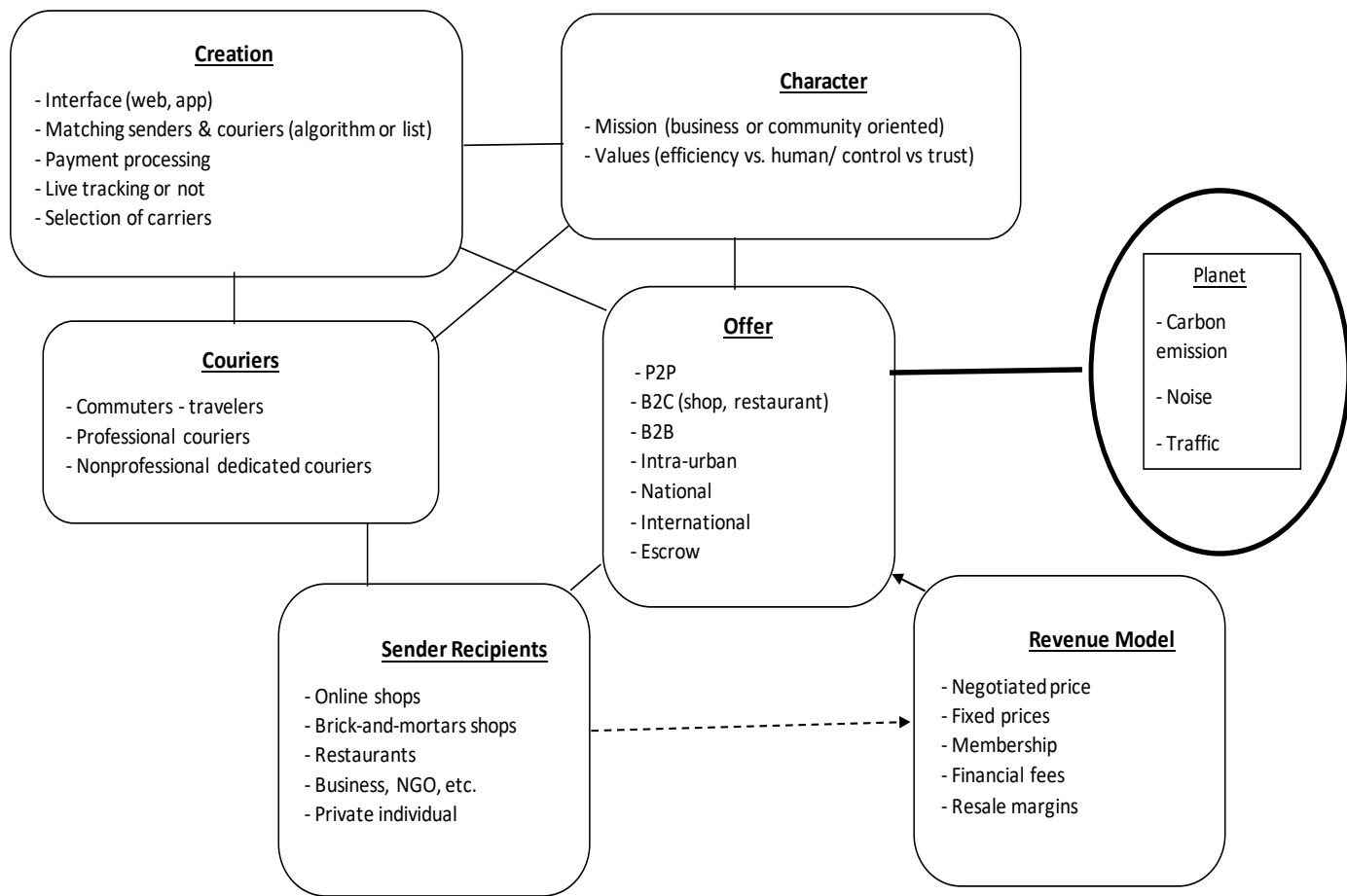


Figure 2.3: Triple Bottom Line conceptual model

2.7 SUMMARY

Many companies have been successful because of the concept of outsourcing. The use of technology adds a platform in which scattered and unrelated individuals, the crowd, can be organised to offer the service of making deliveries, also termed as crowdsourcing (Sampaio et al., 2017). With most people in urban settlement owning smartphones, technology makes it effortless to reach masses, which is key for crowd logistics by enabling access to all kinds of people travelling in all directions. This is very important as one online shopper may want deliveries made to multiple different locations.

This study presented a literature review that points to a research gap and the lack of a crowdsourcing model that incorporates environmental benefits. The study argues that environmental benefits can be anticipated from the use of crowd logistics as it

enables better usage of transportation capacity by creating efficient sustainable coordination of existing transport offered by the crowd, by reducing traffic congestion and greenhouse gas emissions. The latter supports Sampoio et al's (2017) argument for an environmentally oriented crowd logistics model. The following section discusses the research design and methodology considered best for the study's research objectives.

CHAPTER 3: REASERCH DESIGN AND METHODOLOGY

In an attempt to obtain a sustainable solution and a credible study, several research approaches were considered but ultimately the quantitative research approach was decided upon as the most suitable for this study. This approach involved the usage and analyses of numerical information using statistical techniques. The main drive behind this approach was to ensure that questions of what, who, where, when, how, how many, and how much were posed and answered (Petzer, 2016).

3.1. PHILOSOPHICAL STANCE

From literature, there are different philosophical stances and each of them has different ideas and beliefs. Examples are: Idealism, which has the belief that reality is ultimately spiritual or mental in nature, that physical phenomena and material objects are products of the mind (Guyer, Paul and Rolf-Peter Horstmann, 2021); Materialism, which has the belief that everything in the universe, including human thought and consciousness are explainable based on natural laws and physical matter (Carswell, 2023); Existentialism, is the belief that individual experience and free will are paramount, that individuals must create their purpose and meaning in life (Iram, 2013). Pragmatism, the stance taken on this study, which has the belief that the truth or value of an idea is determined by its practical usefulness or success in achieving its intended goals (Maarouf, 2019). This stance was chosen as the study aims to implement a solution that will be used by people, which can either be a success or failure based on its usefulness.

3.2. QUANTITATIVE RESEARCH APPROACH

The use of a quantitative research approach was explained as the use of quantitative measures and experimental methods to examine a hypothetical generalisation (Maxwell, 1992). Moreover, quantitative research was defined as an approach that explains a phenomenon by collecting and analysing numerical data, using mathematical methods (KamolsonSu, 2007). Furthermore, the quantitative approach can be used in understanding defined “variables” and “populations” based on “results” from a sample.

While there are a number of different research methods, as depicted in Figure 4a, the quantitative approach was chosen as it offers the ability to understand emotion, motivation, behaviour, and cognition. Quantifying these into numbers would enable understanding the crowd, delivery patterns, distances, purchases, travels as well as pricing among other involved variables. Additionally, quantitative data collection is very much structured as compared to its counterpart, the qualitative approach. Furthermore, as a data that is numeric and statistically evaluated (Gilliland, 2017), quantitative data becomes inputs on algorithms that were used on this study to determine the general trends across the population in relation to last mile deliveries and consumer behaviour. The quantitative data can then be used when designing a solution, such that the design can take in the data as part of the variables. Moreover, the data is considered when doing the design and the implementation (Figure 3.2).

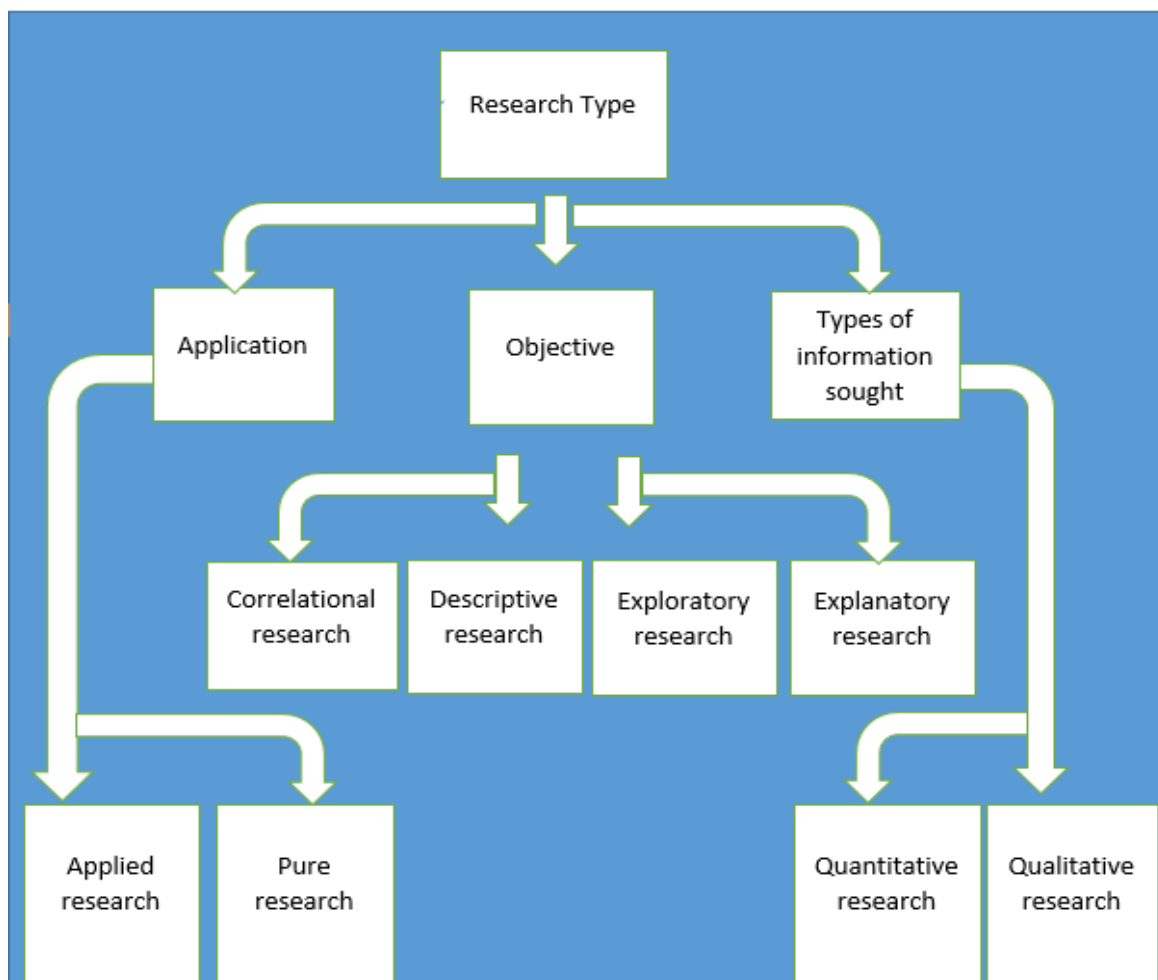


Figure 3.1: The different kinds of research

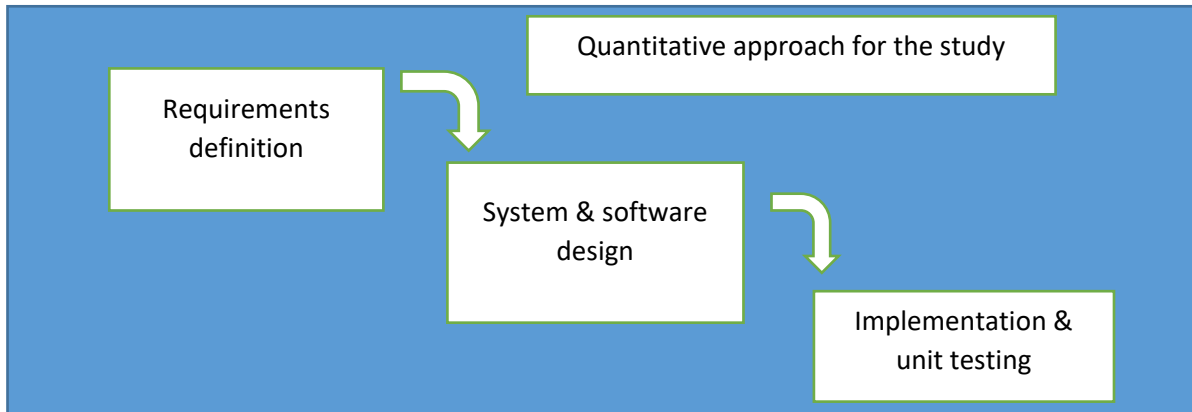


Figure 3.2: Approach for the study

3.3. PHASE 1 – SOFTWARE DEVELOPMENT LIFE CYCLE

A software development life cycle (SDLC) is a complete process of implementing an idea or a software solution from its inception where requirements are defined through to fielding and maintenance (Ruparelia, 2010). This process helps ensure the quality of the software, that the software is delivered on time as per agreed timelines and at the correct agreed-upon budget.

The SDLC process provides a framework that enables key software development activities. Such activities include planning and project tracking, and process visibility to all stakeholders that help identify risks and threats while allowing for the ability to mitigate such (Managers, Guideline and Sdlc, 2008).

3.3.1. Requirements Gathering and Analysis Stage

The section of requirements gathering is one where the project or software solution is broken down into segments so that the maximum understanding may be understood. These segments would be user requirements, system requirements and functional requirements (Thorogood, 2009).

Gathering requirements is a very important part of the SDLC; it helps in ensuring that the problem and the solution are thoroughly understood to ensure quality and satisfactory software. All this is crucial because when done incorrectly, it could

generate bad user experience. Additionally, misunderstanding the requirements can be costly to fix in terms of time and money (Subhan and Bhatti, 2015).

3.3.1.1. Use cases

Use cases are a type of gathering requirements; they are used as a means of depicting the different scenarios, user interactions and the behaviours of the system and its users (Lindholm and Lindholm, 2012). The user diagram in Figure 3.3, illustrates an overview of an online delivery system where the actors are already-registered users of the system making a request for a delivery to be made, a new user and an already registered user offering to make a delivery that would be on the same direction to where the user is travelling to. The system would decide on the shortest route with minimal traffic, calculate the expected delivery, expected payments and the user administration process.

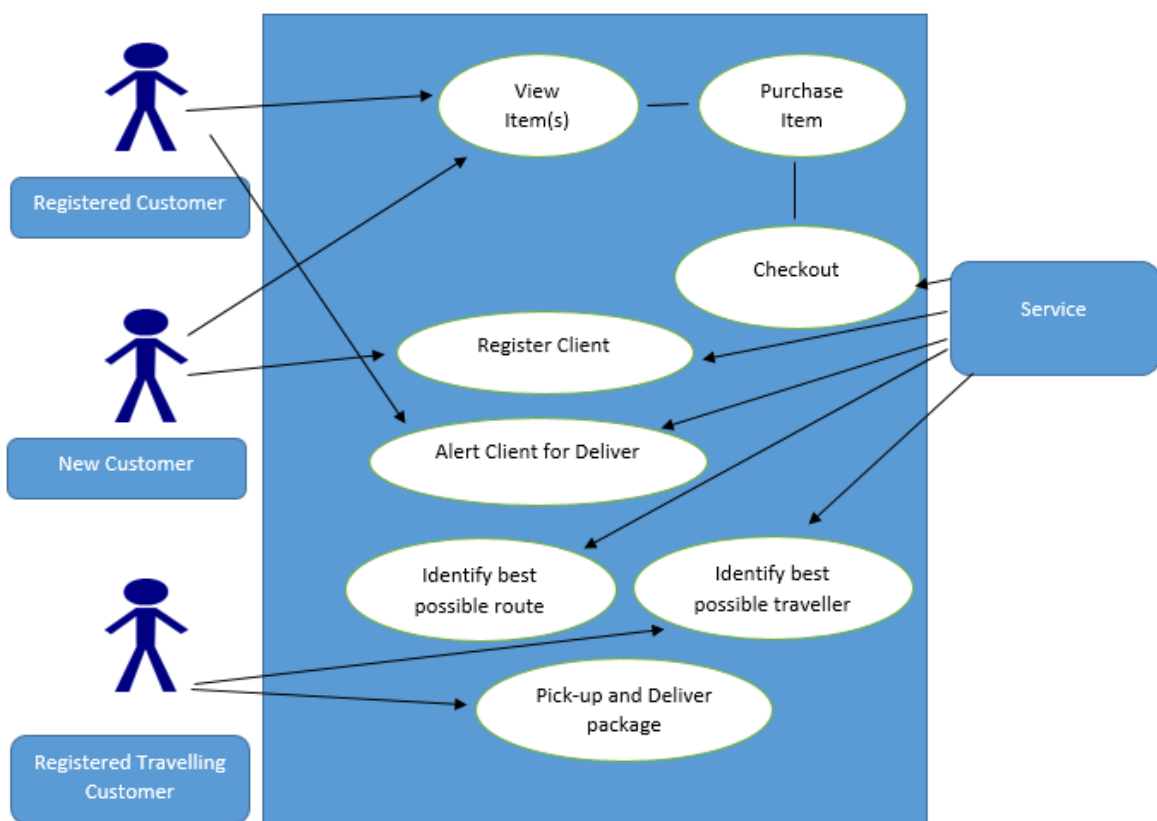


Figure 3.3: Use case diagram of the online delivery system

3.3.1.2. Flowchart

A flow chart helps by enabling the visualisation of activities in a system (Prasad and Strand, 1993). This being key on the process flow in terms of identifying the value of each activity and identifying possible bottleneck activities to improve development time and offer product quality. Figure 3.4 gives a high-level view of a crowdsourcing system.

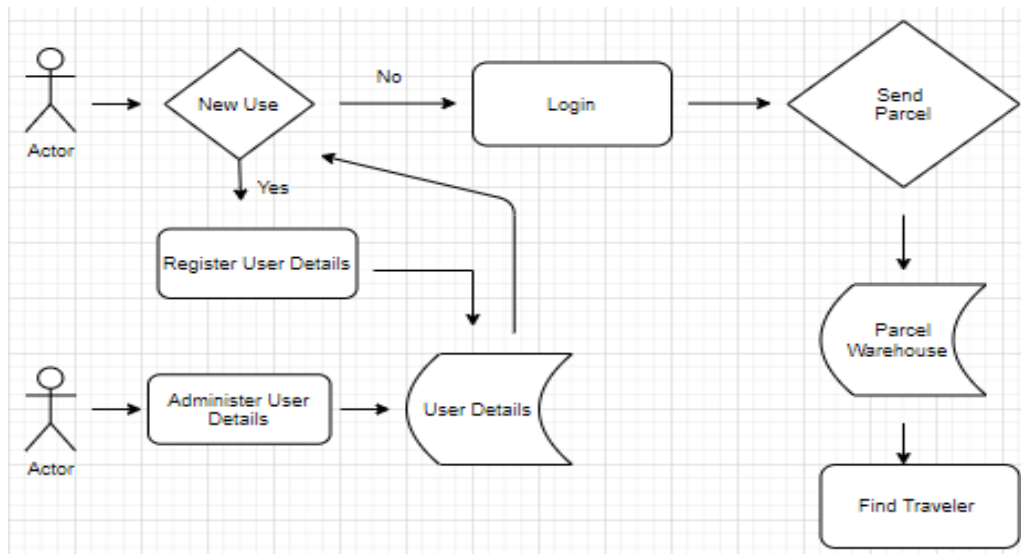


Figure 3.4: High level flow-chart diagram

3.3.2. Design Stage

The design stage entails translating the requirements into a model that derives a solution. Following an analysis, the requirements are transformed into a design document that would have the necessary inputs for the development phase (Maryland, 2009). The information obtained during requirement gathering would yield a logical design that would ensure data entry into the design information system is correct. The design stage for this study would ensure that all the requirements are understood and that the designed solution could be understood fully for the implementation phase to commence.

3.3.3. Implementation stage

In SDLC, the implementation stage is a process of creating a solution as per the design to fulfil predetermined requirements. This process of development creates testable

artefacts in terms of functionality (Priya, 2008). The functional requirements and system behaviour are developed to be tested and signed-off should they meet the agreed-upon requirements.

3.3.4. Integration and testing stage

Integration involves piecing together the different artefacts, enabling the testing of the system and security as well as user acceptance to affirm that the system meets the requirements according to the designed solution. This provides assurance to all stakeholders that the developed system is per specification.

3.3.5. Deployment

The deployment phase entails rolling out the developed software to the public or publishing it in the production space. It comes after the system or software has been tested and approved that all test cases pass.

3.4. LIMITATION AND RESTRICTIONS

The study will not cover the entire software development life cycle. Due to time constraints, the study is only limited to development, which is based on findings from the literature review. This, in turn, also limits testing to functionality and does not address the impact to profit, planet and people. Nevertheless, the study offers a digital solution as a platform for measuring impact which can be leveraged for future studies.

3.5. ETHICS STATEMENT

During the execution of this study, ethical conduct was adhered to, by respecting the rights of others who are directly or indirectly involved or affected by this study. Participation in the study was premised on informed consent, and confidentiality was guaranteed during the different stages of the research process.

CHAPTER 4: SYSTEM DESIGN AND IMPLEMENTATION

4.1. INTRODUCTION

This chapter discusses a hybrid approach for two software development models (SDMs), the waterfall development model and agile development approach, as depicted in Figures 4.1 and 4.2 used for the system design and implementation of the green crowd (Dilivari) mobile application. The agile model follows an adaptive approach through the development cycle while waterfall is a plan-driven approach (Gate, 2017). Evidence from previous studies indicates that a mobile application should stay ahead of trends and demands to compete with many others being developed daily in the same space (Vijiyalakshmi, 2016). An application's performance and aesthetic visuals are a minimum requirement for the development of mobile applications. The scope of the discussion in this chapter includes system design, implementation, testing, and deployment. Before covering the scope of discussion, the two SDMs are explained below:

4.1.1. The Waterfall Model

Waterfall was first published by Winston Royce in 1970 (Mccormick, 2012) as a software development approach with many drawbacks and limitations. However, this approach has been successfully implemented by many software development companies. The waterfall model follows a sequential flow, where each process would be dependent on the completion of the previous process – a literal depiction of a waterfall, as the water flows from one level to the other (Figure 4.1). The Waterfall framework consists of five building blocks: requirements gathering and analysis, design, implementation, integration and testing, and deployment and maintenance. Each building block is elucidated below:

1. Requirements gathering and analysis – a compilation of requirements and detailed application information necessary to provide an understanding of what is to be designed (Senarath, 2021). The analysis accomplishes the task of understanding the hardware and software requirements for the database necessary to complete the task.

2. Design – this phase is fully dependant on the previous step and it is crucial for specifying the hardware required. It also gives a visual representation of the application to be developed (Dubey, Jain, 2015).
3. Implementation – pursuant to the design phase, creates building blocks as units of the application (Dubey, Jain, 2015), which are then tested.
4. Integration and testing – units generated from the previous phase are integrated/combined and tested as a complete software application.
5. Deployment and maintenance – the application is made available to end users (Samra, Li and Soh, 2020). The maintenance phase fixes system glitches and issues picked up during deployment and introduces enhancement.

Figure 4.1 illustrates the phases of the waterfall model with their flow of dependency.

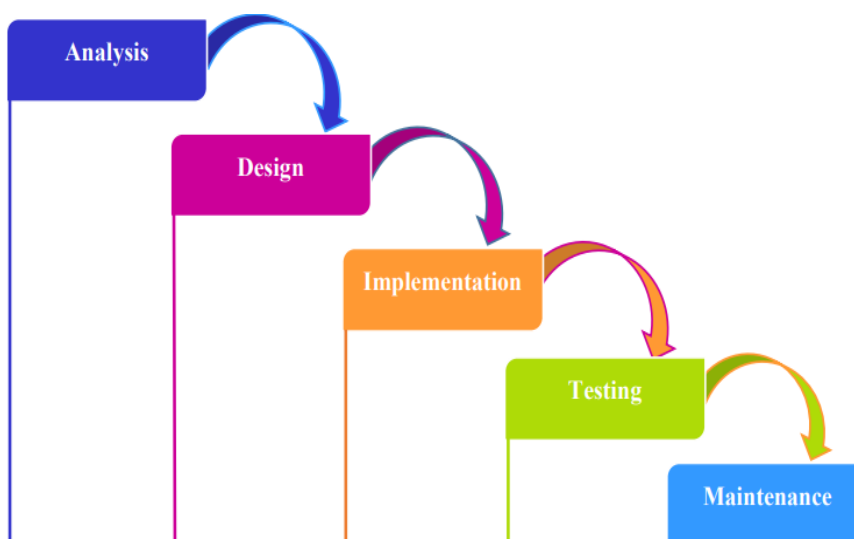


Figure 4.1: Waterfall model steps

An overview of the waterfall model shows that it is advantageous compared to other software development models in terms of producing quality because of the predefined structural flow (Gate, 2017). Furthermore, the same structure gives visibility to all stakeholders of the progress of the project as it moves along different steps of the waterfall model (Gorrod, 2004). Additionally, project tracking and monitoring is manageable and visible as the project moves from phase-to-phase. Mixed evidence illustrates that the waterfall model has shortcomings that must be considered before undertaking to use the model. Highlighted drawbacks and limitations are that it is costly

to revert to a surpassed phase, and changes in one area impact the next phases due to their dependencies (Dubey, Jain, 2015). Furthermore, the requirements must be clear for the next phase to proceed.

The disadvantages, however, have a significant impact on mobile development where the turnaround time is expected to be short and the changes quick and seamless (K. Flora, V. Chande and Wang, 2014). This is the basis for incorporating a more agile approach to compensate for these shortfalls. The agile model is explained in the next section.

4.1.2. The Agile Model

The agile model is a well-studied topic in literature (P. Abrahamson, Outi Salo and Jussi Ronkainen, 2002) and is said to be grounded on iterative development. The software application is broken down into small features that are then prioritised, developed and tested in an iterative flow. The iteration is a communication feedback mechanism that has quick decision making and action.

The approach requires considerable training and practice to master (Al-Saqqa, Sawalha and Abdelnabi, 2020). Using the model for large and complex projects has its challenges and requires an organisational change where developers work closely with staff. In contrast, the model has advantages that make it best suited for other projects because it is flexible and adaptable to requirements and situations, the time to deploy is short, has lower costs, and it drives customer and user satisfaction. The model's flexibility, adaptability and shortened time to a minimum deliverable output contributed to it being incorporated into the green crowd (Dilivari) mobile application together with the waterfall model. Figure 4.2 illustrates a typical agile iteration flow in developing software.

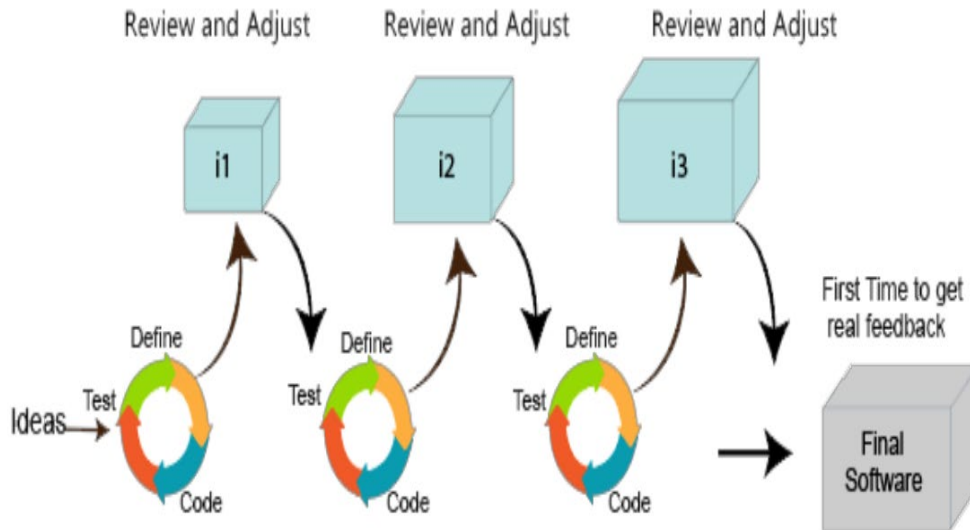


Figure 4.2: Agile model sample diagram

The sections that follow discuss in detail the elements of the waterfall and agile models used for conducting this study and developing the green crowd (Dilivari) mobile application.

4.1.3. Application of waterfall and agile model in green crowd mobile application

4.1.3.1. *Requirements gathering*

The start of any software development project is understanding the needs of the software to be produced. This provides the elements to understanding the design and tools necessary to successfully deliver the software. The next sections delve into the details of requirements gathering.

System requirements

System requirements detail how the system should function. From literature, Akanmu et al (2013) acknowledge that to avoid creating software that does not attract the use of users one has to produce software that meets the needs of the users and the needs of the system being designed. To attain user satisfaction, user requirements must be gathered, analysed, documented and translated into functional and non-functional requirements to effectively cater for the design. Well-documented requirements help eliminate time spent on rework.

Functional requirements

Functional requirements documents the functions that were expected to be provided by the application (Bahill and Madni, 2017) such as identifying functional requirement support prioritising requirements, keys for setting precedence to meeting delivery targets and setting a framework for testing (Dabbagh, Lee and Parizi, 2016). For this study, the functional requirements were fully grounded on Mkansi (2013) because the project evolves from a supply and distribution of last mile deliveries by Mkansi (2013). Drawing inspirations, Mkansi (2013) revealed insights necessary to balance the triple bottom line and as the key project leader, the following were established by her and validated with key stakeholders. For this project, the functional requirements were:

1. Create user account

The application needed a functionality to allow a new user to sign up for an account.

2. Login into the application

The application had the capability to allow an already-registered user to login using their credentials (email-password combination).

3. Request delivery to be made

The application catered for the user to add a package with the necessary details to enable delivery.

4. View available delivery requests

The application offered the user a view of packages listed to be delivered.

5. View own delivery requests

The application made it possible for the user to see their own requests.

6. View receipts

The application allowed the user to view all completed requests.

7. Pick-up delivery request

The application allowed a registered user the functionality to offer to make a delivery.

8. The application should auto complete addresses

The application needs the have functionality to pick up the address.

9. The application calculated the distance from pickup to destination and approximate the delivery time.

10. Sign delivery on receipt

The application offered the destination user to acknowledge receipt of the package.

User requirements

User requirements document the user's expectations of the application (Vijiyalakshmi, 2016). The user requirement objective was to answer the "What" question; the requirements must explain what is expected not how it will be delivered (Officer, 2016). This section expands from the work done by Mkansi (2013). The green crowd (Dilivari) mobile application user requirements were:

1. A user must be able to create a profile and sign into the application.
2. A user must be able to request collection and delivery of a parcel to a pre-defined location at a reasonable price based on the size and the distance to the destination.
3. A user must be able to view a list of delivery requests and make an offer to deliver a package enroute to their destination.

The requirements should be accurate, clear, complete, verifiable, consistent, understandable and concise. Technical solutions or elements (e.g. data architecture, application architecture, system architecture, technical infrastructure, etc.) should be avoided. Such solutions were proposed by IT staff during system design. Acceptance criteria defined the boundaries for the functional requirements, and they were written in a clear and concise manner. If the agile software development method is used, only high-level requirements should be produced in the system analysis and design stage. Requirements should be written in layman's terms called "User Stories". More information on agile is available in a document called the "Practice Guide for Agile Software Development" (Officer, 2016).

Environmental requirements

The current environmental requirements expand on a previous study (Mkansi, 2013) presented at a Norwegian conference. Environmental requirements were needed for

the green crowd (Dilivari) mobile application to function as intended and used successfully (Buys, 2009). Environmental requirements were:

1. A smartphone powered by either an Android or IOS operating system
2. An Internet connection
3. An active email account
4. The user must have access to download the application from any of the online app stores such as Android, Apple IOS, Windows store or market place.

Actors and roles

Actors and roles depict the interaction between the user and the application and their subsequent interaction within the application components. This is further explained under the UML diagrams. The Unified Modelling Language (UML) was used to model software solutions, structure and behaviour using class diagrams, use case diagram, etc.

1. Administrator
2. Requesting user
3. Receiving user
4. Application

Limitations

Limitations are constraints, challenges or blockers impacting the application design or implementation (Hwang *et al.*, 2020). There are a number of such challenges discussed hereunder. One key challenge, among others, was the 'location'; applications that use the knowledge of user's location are known as Location-based Services (LBS). LBS is a type of contextual service that helps in determining the geographical location of a user and then provides information based on this location to the user in order to enhance overall experience. The service of maps and location is rendered by Google maps and the API service they provide. Such service providers are considered as a Geographical Information Service (GIS) providers and assist with application functions that require map information (Jha and Chourasia, 2011). The figure below illustrates the necessary requirements for accurately obtaining a location.

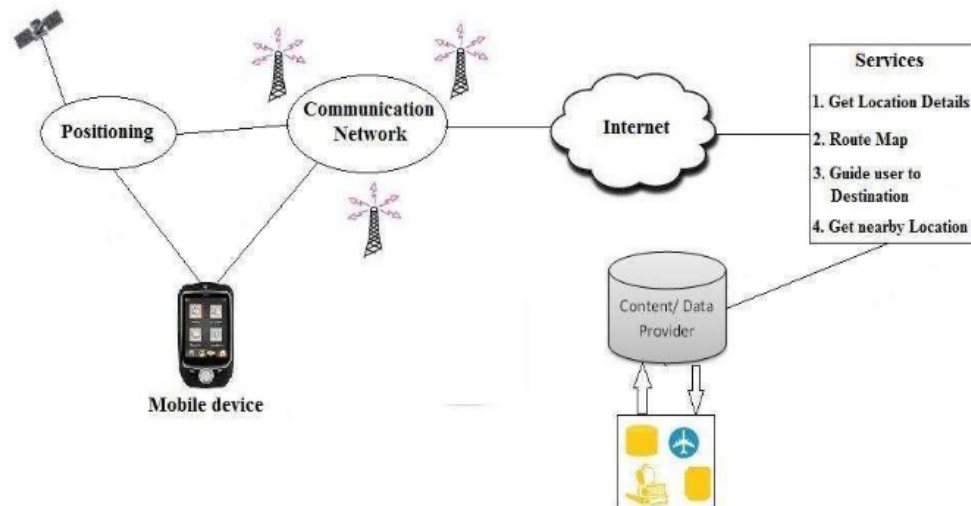


Figure 4.3: Required components for obtaining location (Jha and Chourasia, 2011)

The Google Map API is a service from Google that enables an application to gain access to functions such as locations on Maps (Net, 2020), identifying different routes between to location points and estimating travel time by different modes of travelling such as a train, walking, a car or cycling. The service generates an API key that uniquely identifies the application making a call to Google. The API key is also used for billing; high traffic calls from the application result on Google requiring payment for the calls (Google, 2022). The location was then listed as a challenge because it was dependent on several factors to report accurate results: a user must be in an area with good connectivity and a reception, have a data connection or Wi-Fi and the location must already be listed on Google Maps.

Location was not only a concern for accurately determining the user's pinpoint address, but it was also a security concern – a second constraint to the study. A disadvantage of location services is that they pose a security threat if in the wrong hands. For instance, Street View images can spot belongings momentarily exposed through windows or open doors. For example, Vandeviver (2014) found that burglars have used Street View and satellite images to seek affluent neighbourhoods or homes to burglarise, or to identify parked cars that may contain expensive possessions. Google collects all kinds of data on both Android and IOS devices (Molinary, 2018), from Google browsing behaviour, YouTube, Gmail activities, location history, online purchases, and many more. Security on user data is a limitation to be explored within the application such as authorisation and authentication. Currently, there is an email

password combination authentication that will be expanded to two-factor authentication with the use of a one-time-pin (OTP) to ensure that hackers cannot gain access to sensitive information. Attention to detail was necessary in the development phase on session-handling and login functionality (Practices and Security, 2014). Furthermore, the storage of the data from the application was kept at Microsoft data centres, relegating the responsibility of data security to Microsoft. Keeping the data on Microsoft data centres also elevated the responsibility of dealing with data security. The next limitation on the study related to data integrity, which was a different responsibility.

Data must be correctly captured and reflect all updates throughout the cycle of a delivery request. There are circumstances whereby the application may require Internet connectivity, where wireless communication is hindered such as disturbances due to Eskom's load-shedding, a remote area like rural settlements, or a shielded building. Wireless networks do not provide 100% coverage and probably will never get there. Even if wireless communication was available, it can be unstable. Mobility (2013) reports that in the first half of the year 2013, issues associated with making phone calls, sending messages and data-related activities averaged 11 problems per 100 calls (PP100) for full-service carriers. Mobility (2013), furthermore, explains that the developer must consider many challenges to synchronising data between the server and multiple mobile devices to maintain data integrity even when offline (Mobility and Platform, 2013). Due to time constraints such concerns around data integrity have been relegated and earmarked for the next version on the application.

In extension to data integrity, data should be complete, correct, accurate and verifiable. For the application, due to time constraints and with the plan of expanding the study, validating user inputs was omitted for future improvements of the application. Validation on locations was also left to the user's discretion for the first version of the application but may be improved upon in the next version such that a request is within viable routes.

4.1.1.1. System Design

System design is a planning step that entails converting the requirements into structure, components, modules, interfaces, data and methodology used in the

implementation. This step produces artefacts such as the application architecture, design diagrams, use case diagrams and flow chart diagrams which are key for the implementation step.

Application Architecture

Application architecture is built to reflect application requirements. Whitten (2000) explains that application architecture specifies all technologies needed to implement an information system. The application architecture can be depicted as a multi-layered diagram that would include the user interface, user communication layer, information retrieval layer and the database layer. Whenever a complex system is designed, architecture is needed to provide a blueprint of the system that defines the structure of the system to be designed. There are three main principles of implementing an architecture, separating concerns, minimising interdependencies, and isolating fundamental components. Using these established principles helps in modularising the application into smaller functions that are easy to modify autonomously (Costa, Diehl and Snelders, 2019). The same principles were applied to the Dilivari application by breaking the system into three components: the user interface (UI) which is the screens of the mobile application, the data model layer that defined the data structure, constraints and API calls to the final component, and the database containing the data. Figure 4.4 below gives a visual of architecture of the Green Dilivari application. The diagram follows a pre-stated multi-layered diagram that would include the user interface to show on the mobile device's screen, and the user communication layer would be harboured by the code running on the iOS or Android operating system responding to the instructions as the user makes commands on the application. On the information retrieval layer, the Flutter code communicated with MongoDB by making API calls. Finally, MongoDB was the database containing all user information.

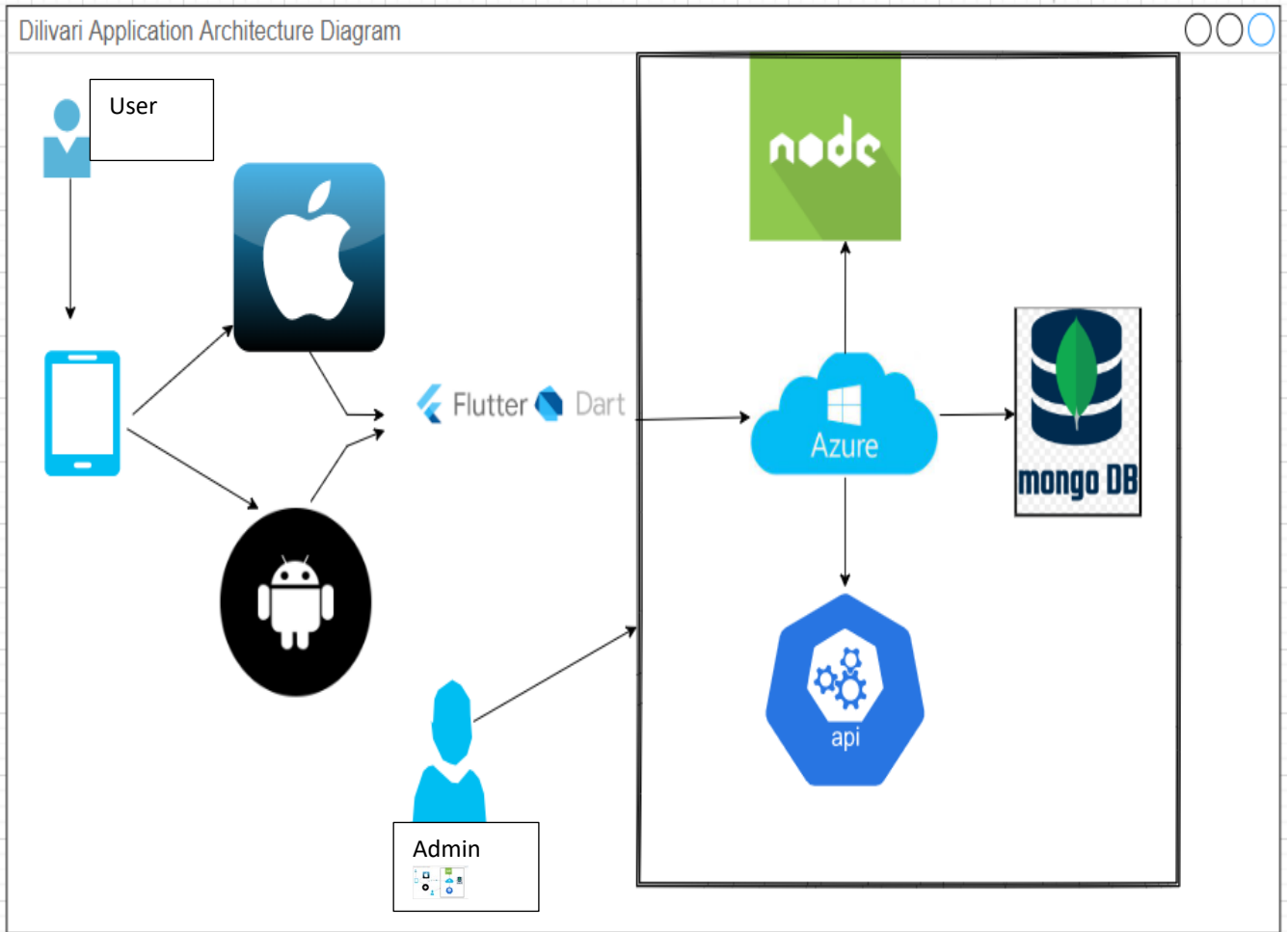


Figure 4.4: Green Dilivari architecture diagram

Use case diagram

Use case diagrams were used to capture system requirements that led internal influences and external influences, covering mainly design requirements (Mule and Waykar, 2015). The use case diagram was used as a tool to give a visual depiction of the possible interactions between the user and the Green Dilivari application. Figures 4.5, 4.6 and 4.7 depict the interaction. There were two flows for the Green Dilivari application. The first flow was a user signing up, then logging in with an email-password combination. From there on, the user added a delivery request by creating a new delivery, adding the delivery details such as the delivery type, a description, weight and a pick-up and destination addresses.

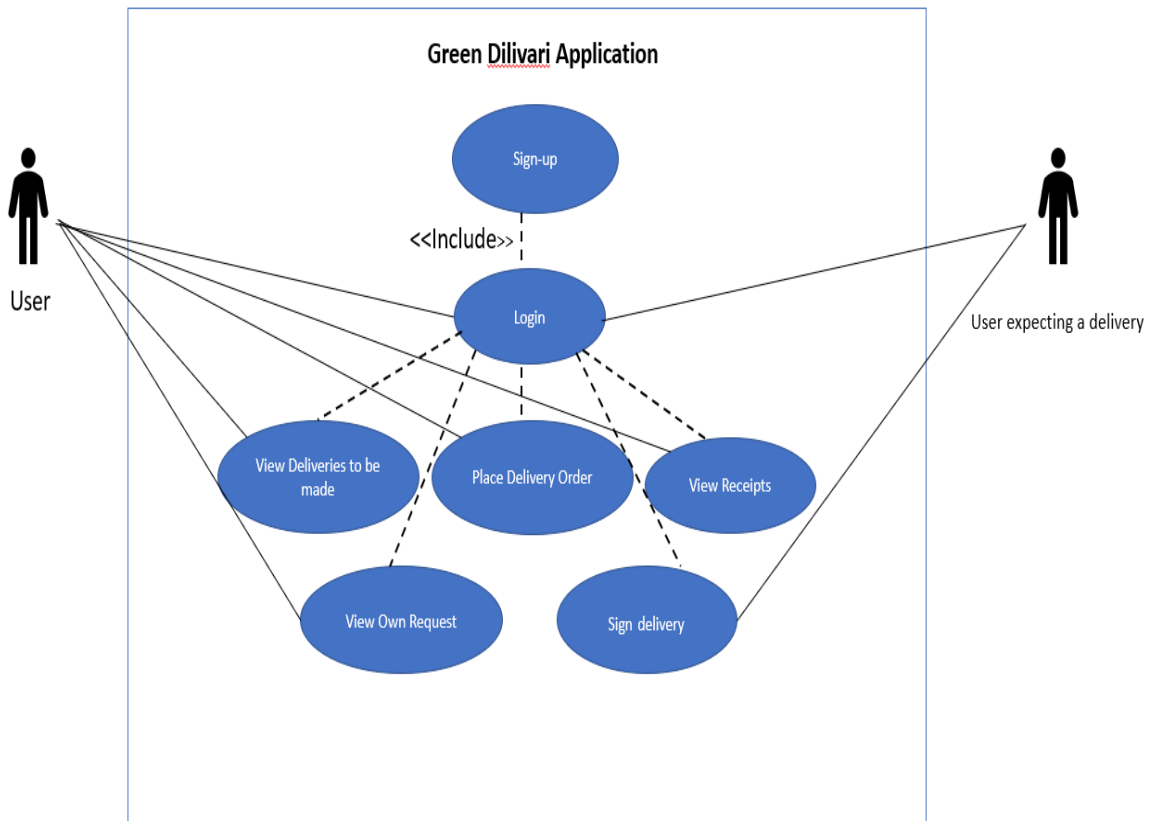


Figure 4.5: Use case of a user requesting/offering a delivery

The second flow was when an existing user of the application made an offer to pick up a delivery request and embark on the journey to make the delivery.

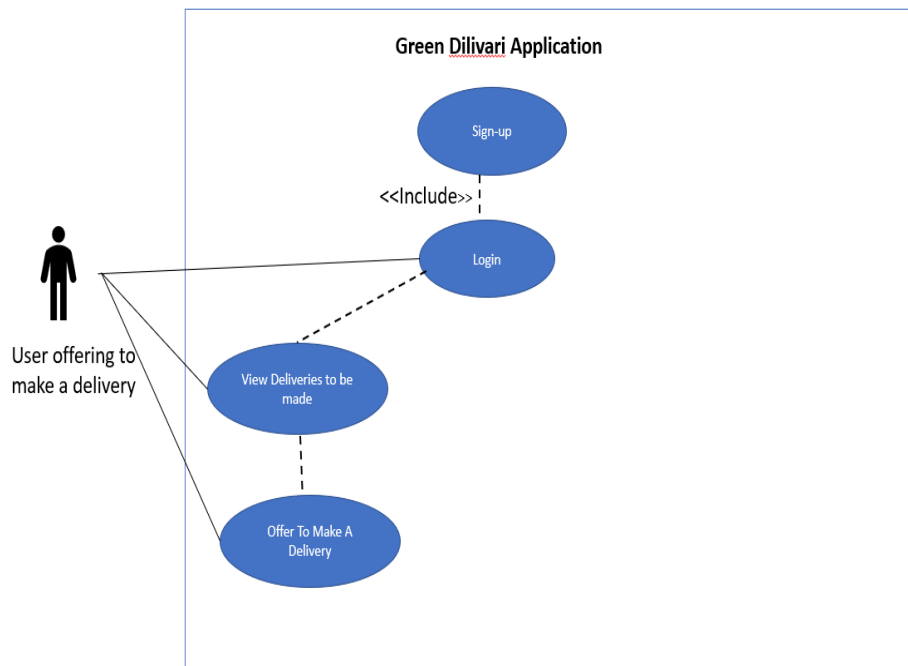


Figure 4.6: Use case of a user offering to make a delivery

The below use case diagram of the Green Dilivari application illustrates the back end meant for enhancing the user experience and creating a seamless user journey. The actor on this use case diagram was the application administrator responsible for all users' data and the back-end infrastructure, that is the database and API endpoints. The administrator was also responsible for adding new features, making enhancements and updates on the application.

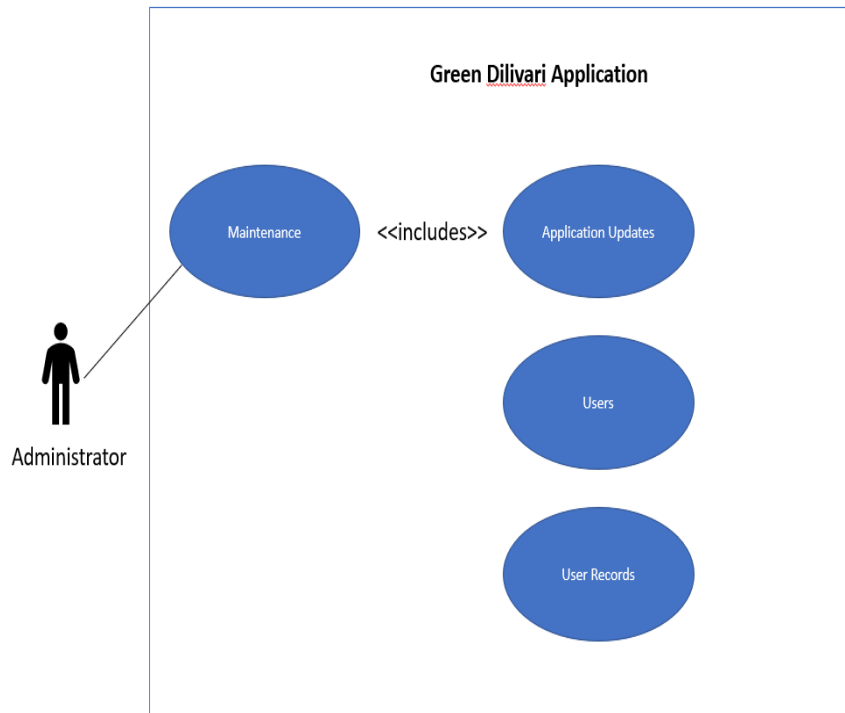


Figure 4.7: Use case of the administrator of green crowd (Dilivari) mobile application

Class diagram

Class diagram is another form of modelling in software development. A class diagram was utilised to visualise the granular levels of an application by explaining the different components of the application. Thus, class diagrams are at the level of describing the application classes with their attributes and operations (Bahill and Madni, 2017). This form of modelling is more focused on giving a visual depiction of a diagram of the classes of a system; however, it comes with the disadvantage of being difficult to maintain as the design expands over time. This type of modelling, a functional requirement, was used to give the developers and the testers a clear map to work with and to develop test plans to validate the development as per specification (Imcs *et al.*, 2020). Figure 4.8 below shows the classes of the green crowd (Dilivari) mobile application. Each of the classes had attributes and methods that would be performing functions that were documented as the application requirements. The class diagram was very useful at mapping out the layout of the code and the start process of development.

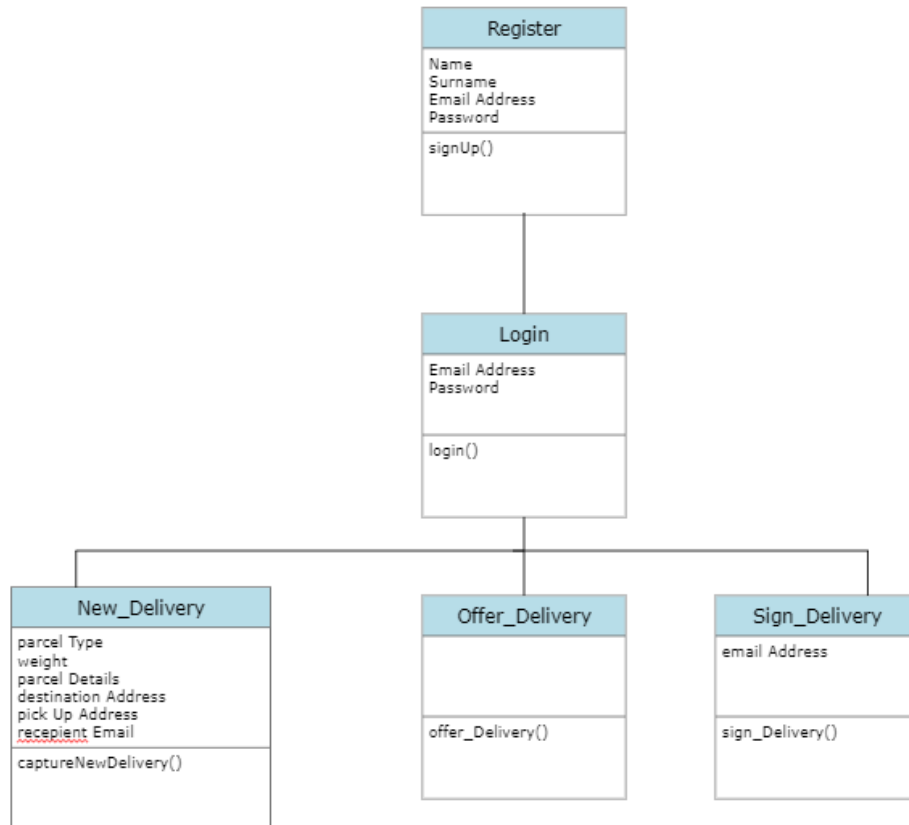


Figure 4.8: Green Dilivari Class Diagram

Entity-relationship diagram

Entity-relationship diagram is a graphical representation used in the design phase showing the relationships between individuals, things, concepts, or events in an information system (Learning and Reserved, 2010). This kind of modelling was specifically used for modelling the data layout for the database structures for storing information. In the strategy phase, the business requirements were addressed using a strategy document as a guide to the next phase, using a strategy entity relationship diagram (ERD) to break down the basic business unit categories and process flow diagrams to outline the basic flow of the high-level one entity to outline-child processes. The ERD is advantageous at achieving data modelling goals and develop a conceptual model used to build the physical database structure (Samra, Li and Soh, 2020). With a drawback of being very abstract, the ERD in this study was a key enabler of identifying relationships among entities. This modelling tool gave a high-level picture of the relationship between the user and the Green crowd (Dilivari) mobile application as two entities that interact with two main functions, requesting and offering a delivery.

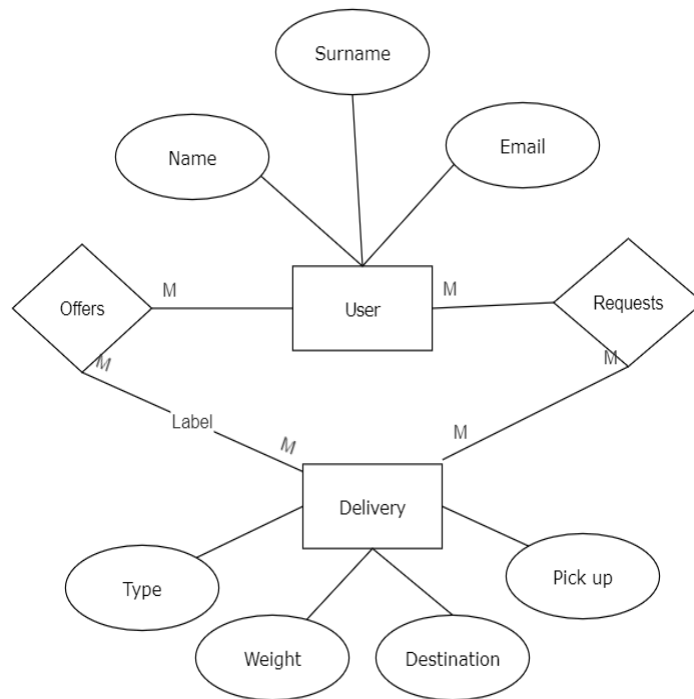


Figure 4.9: Green crowd (Dilivari) mobile application ERD diagram

Flowchart diagram

A flowchart is a diagram of the steps in a process and their sequence. These flowcharts show the main blocks of activity or the main system components in a process and are furthermore used to confirm requirements, and identify and eliminate duplicate information (Gorrod, 2004). These charts are especially useful in the early stages of a project and help prioritise improvement work. These flowcharts make it easy to visualise the application flows, spot complexity, and excessive steps. Evidence from literature illustrates that flowcharts have the benefit of outlaying a step-by-step display of functions performed by an application (McCormick, 2012). In this study, the flowchart was used to convert the requirements into pieces, layers and flows used to map out the architecture of the application.

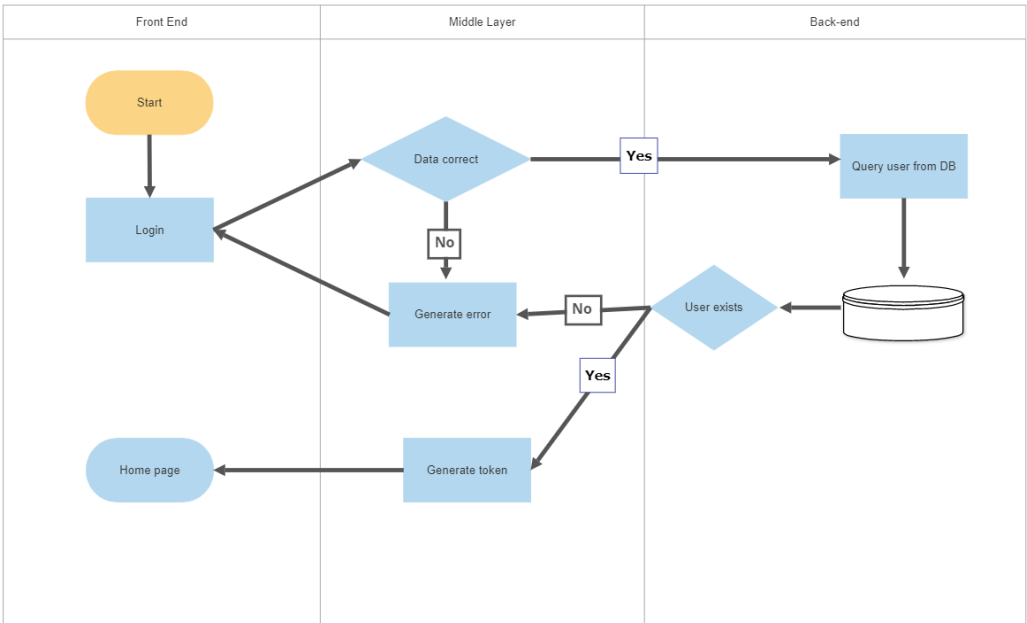


Figure 4.10: Login flow

Figure 4.10 shows the flow of a registered user executing a login request on the application. The data was validated such that an email address has an '@' character with a password minimum length as prerequisites to be met before an API call is made to query if the user exists or not on the database.

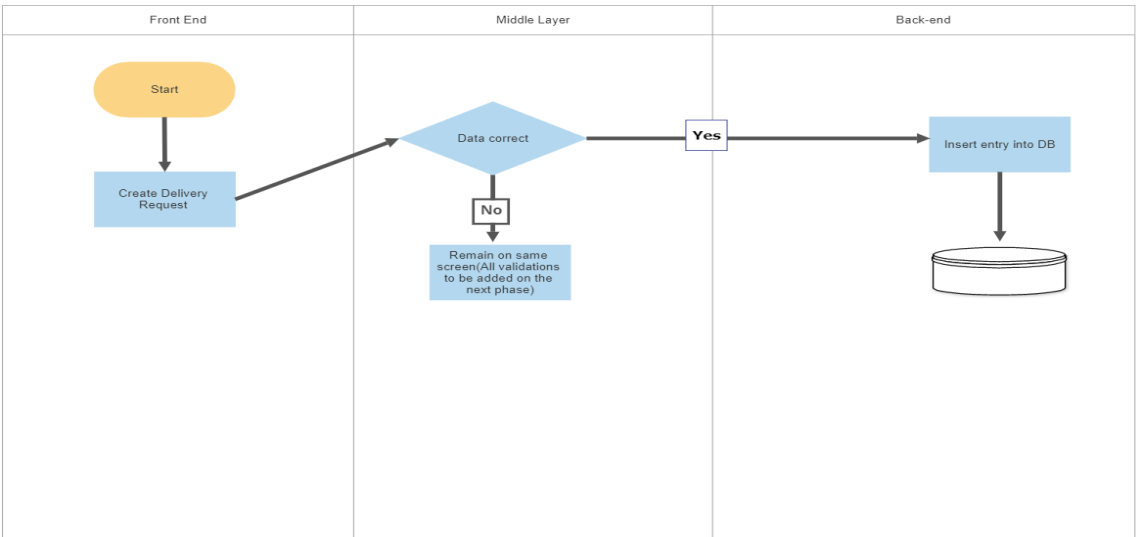


Figure 4.11: Delivery request flow

Figure 4.11 illustrates the flow of making a delivery request. Due to time constraints, not all validation of the data and all the error messages were included on the application as this is the first phase. Figure 4.11 shows how a delivery request is made

and then added as an entry on the database that can be viewed by other users of the application.

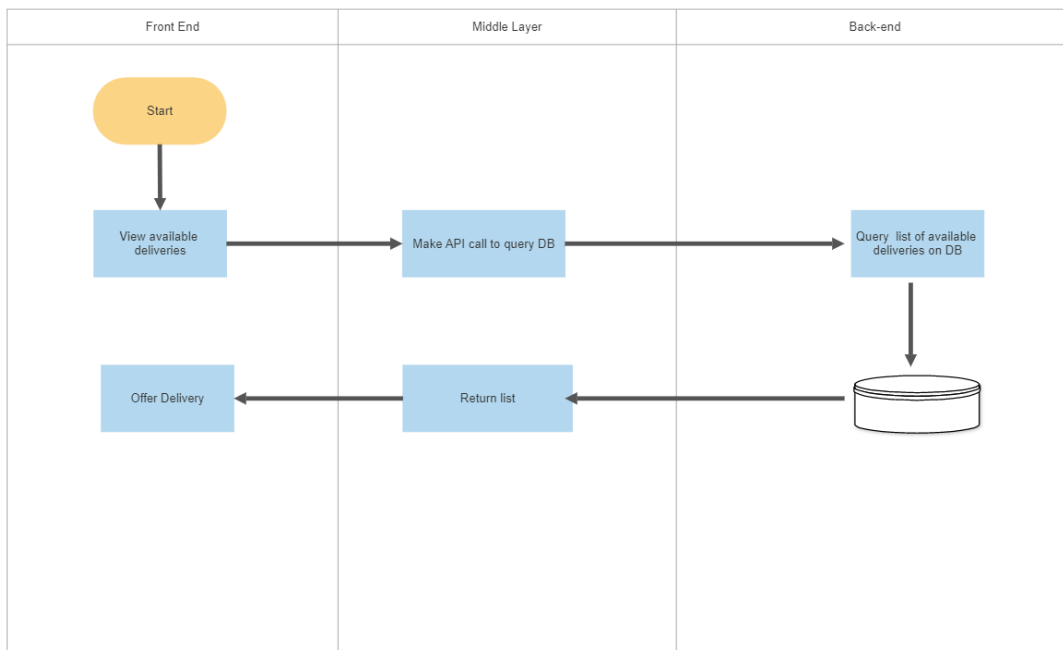


Figure 4.12: Delivery request offer flow

Figure 4.12 is a flow of the user offering to make a delivery. A user had access to view a list of available deliveries to be made and makes an offer to deliver. The three flows are the core flows of the application. Flow such as monitoring the offered delivery, viewing one’s delivery request, and checking receipts follow the same flow of the user accessing what is visible at the front end and an API call to the database to retrieve the data.

With all the modelling and the mapping of the requirements into the design of the application, the next phase was the development phase – the process of implementing the designed solution to meet the requirements.

4.1.1.2. Implementation

A well-implemented application impacts positively on user satisfaction and motivation to continue using the application (Samra, Li and Soh, 2020). The below sub-sections discuss the building blocks of the application, the tools used and the building process of the application.

Flutter

For the implementation of the screens and navigations, Flutter was the tool used for development. Flutter is a software development kit used for creating hybrid applications, that is mobile applications that can be installed on both Android and iOS operating systems while having one code base. Flutter is best for creating high-performance platform-independent applications (Ameen and Mohammed, 2022). Furthermore, with limited previous studies, Flutter proved favourable in development speed and budget reduction.

There were many kinds of applications from which to draw comparison. However, comparison was drawn between native and hybrid applications because they had fewer disadvantages to the other types shown in Table 4.1. A summary of previous literature suggest that native apps have a better memory consumption (Wasilewski, 2021), CPU utilisation and a smaller package bundling (Table 4.2). In contrast to power consumption, both were on par. Results once more suggest that Flutter applications were better than native application in terms of being smoother, having better performance and easier learning curves, and taking less time to develop. Drawing attention to the ease to learn and less development time, and the previously mentioned point about Flutter having a single code base for both applications using Android and iOS operating systems, contributed in changing the direction of developing the Dilivari application from using Android at the onset of the study to creating a native application that used Flutter. Drawing inspiration from the above and reviews from developers, the study incorporated Agile as a complimentary method to waterfall in implementing the application using Flutter. Further comparisons between Flutter and Native applications are captured below (Hussain *et al.*, 2022).

Table 4.1 Types of mobile applications

Type of Mobile Application	Advantages	Disadvantages	Example
Hybrid Application	<p>Developing a Hybrid App is cheaper than developing a Native App. It can be built for cross-platforms, i.e., reduced cost for App development.</p> <p>Maintenance is simple, as there are not many versions to be maintained.</p> <p>It can take advantage of a few features available in the device.</p> <p>It can be found in the App Store, which makes the distribution easy.</p> <p>It has a browser embedded within the app only.</p>	<p>Graphics are less accustomed with the operating system as compared to Native Apps.</p> <p>Hybrid Apps are slower than Native Apps.</p>	Instagram
Mobile Web Applications	<p>Easy access.</p> <p>Easy Development: Developing responsive design and restructuring the content to be properly displayed on a smaller screen/hardware will make any desktop website mobile friendly.</p> <p>Easy update: Just update in one location and all the users automatically have access to the latest version of the site.</p> <p>No installation required, as compared to native or hybrid app.</p>	<p>Mobile websites cannot use some of the features like access to the file system and local resources.</p> <p>Many existing websites do not support offline capabilities.</p> <p>Users will not have the app's icon on their home screen as a constant reminder.</p> <p>The website needs to be opened in a web browser only.</p> <p>While native and hybrid apps appear on the App</p>	Amazon.com

		Store and Google Play, web apps will not. So, redistribution is not that sensible.	
Progressive Web Apps	<p>Size of apps can be much smaller than if the same app is built as a Native app. For example, an app that is 10MB as a Native app could be as small as 500KB as a PWA</p> <p>PWA's update automatically when you use them.</p> <p>No need to install the app. Just use the web page to access. However, you can install if you like.</p> <p>You can share a PWA with the use of its URL which is different from other app types.</p>	<p>PWA's are not much used on social media in recent times as social media companies develop their own in-app browser.</p> <p>PWA plugins are not Facebook and Google friendly. Web login is needed to be done separately.</p> <p>Not compatible with the latest hardware such as fingerprint scanners</p> <p>Not supported in some default browsers.</p> <p>Limited Android compatibility.</p> <p>A fair amount of traffic could be missed due to Play Store not able to be directed to the app.</p>	Twitter
Native Applications	<p>Native Apps live on the device and are accessed through icons on the device home screen.</p> <p>They can take full advantage of all the device features — they can use the camera, the GPS, the accelerometer, the compass, the list of contacts, and so on. They can also incorporate gestures (either standard operating system gestures or new, and app-defined gestures).</p> <p>Native apps can use the device's notification</p>	<p>High cost for building the app: Native apps developed for one platform will not run on another platform. An App built for Android will not run on iOS. We need to build a different App altogether for iOS. Because of this reason, we need to maintain multiple versions of the App.</p> <p>Even though you might publish native Apps, you will want to keep the mobile website well maintained, as mobile brings more traffic. As a result, maintenance is higher.</p>	WhatsApp

	<p>system and can work offline.</p> <p>Publishers can make use of push-notifications, alerting users every time a new piece of content is published or when their attention is required.</p> <p>Native Apps maintain UI design of each operating system; thus, they offer the best user experience. For example, a Native App can have a left-aligned header in Android and a centre aligned header in iOS.</p> <p>Redistribution is easy, as it is found in app store.</p>		
--	---	--	--

Table 4.2: Android-Flutter Comparison

	Android App	Flutter App
Bundle Size KB	2990	18,244
CPU Usage (Avg)	11%	20%
CPU Usage (Max)	47%	36%
Memory Usage (Avg)	132 MB	418 MB
Memory Usage (Max)	150 MB	504 MB
Power Usage (Min)	16%	16%
Power Usage (Max)	66%	66%
FPS Render	58 Frames per seconds	66 Frames per seconds

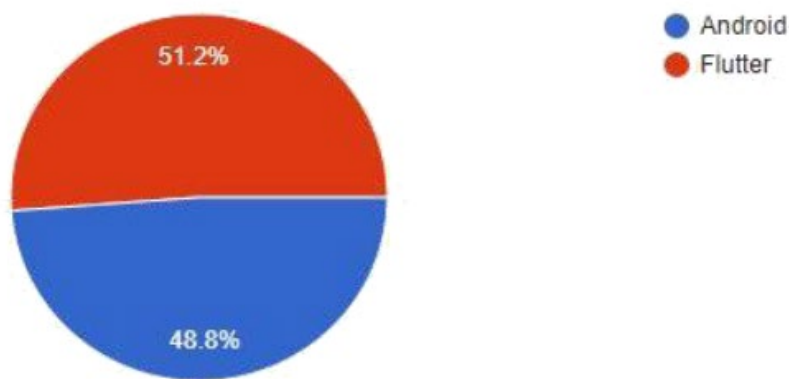


Figure 4.13: Preference votes from developers

Conclusion drawn from the comparisons were that Android is superior on a number of aspects with a key factor for consideration being that Flutter was only launched in the year 2018 while Android had been around for well over 10 years (Hussain *et al.*, 2022).

Integrated Development Environment (IDE)

Bergström (2018) defines an IDE as "... a software application that offers programmers a place to write and edit their code, compile it into functional software and debug it to find unexpected behaviours". IDEs are developed to help make the life of developers easier such that the developer would be assisted in terms of code language syntax, best practice of structuring the code and the ability to step through the code line-by-line when there is an issue (Idan Arb and Al-Majdi, 2020). All this enables developers to focus on problem solving rather than the tools necessary to solve the problem. There are several IDEs available to developers from sold, and customisable off-the-shelf to open-source IDEs. There is no real method of selecting an IDE but there are several factors to be considered before selecting an IDE. A few factors are the cost of purchase, the licensing, the developer-user experience, the languages supported by the IDE and the ability to integrate other tools onto the IDE. With all these factors, Visual Studio Code was selected for use on development.

Visual Studio Code is a Microsoft product, an IDE that is used to develop Android and IOS applications (Olsson, 2020). Visual Studio Code was used to write the code, and to compile and test it. It was also used in the process of packaging the application for deployment to iOS and Android platforms. A summary from previous literature suggests that Visual Studio Code is disadvantaged because it heavily consumes

memory compared to IDEs such as Android Studio and IntelliJ. Furthermore, it does not have an intuitive user interface to configure optimal functionality; in contrast being an open source IDE is advantageous and it is fast, powerful, customisable and could be extended with plugins for developing, simulation and testing (Net, 2020). Another IDE that was used was Xcode, which is an IDE for MacOS. However, it is only limited to Apple devices (Fojtik, 2020), but great for testing and has an instinctive interface for first-time users. Xcode was mainly used for testing the application for IOS deployment and packaging the deployment bundle.

Database

Applications are developed and deployed daily by individuals, groups, organisations and companies. Across different kinds of application one common feature is their need of a database to retrieve data whether remotely or locally. Caicedo (2014) explores the complexity of data retrieval by mobile application from remote databases compared to queries done on local databases. There are different types of databases with no absolute best database solution but there are factors that are worth considering when selecting a database, such as scalability, flexibility, performance, query language, security, data replication, licensing and availability (Sahatqija *et al.*, 2018). Based on these attributes, MongoDB was selected for the green crowd (Dilivari) mobile application. It is an open-source database with no schema; it does not have rows nor columns but stores data structurally using name value pairs stored as documents in collections. MongoDB is faster compared to SQL databases; this type of a database does not require the creation of structure for the data to be inserted, but simply needs a JSON object to do an insert. The advantages of such a database include, amongst others, faster query responses (“MongoDB,” no date), and cost efficiency because it is open source and does not require a subscription or licensing. MongoDB was used for the back-end of the application to store application data. Below is a snippet of how the data was inserted and stored on MongoDB.

```

db.places.insert(
{
  loc : { type: "Point", coordinates: [ -73.97, 40.77 ] },
  name: "Central Park",
  category : "Parks"
})

{
  "_id" : ObjectId("5790c5d9cae25b3d38c3c7af"),
  "key" : "value2",
  "key2" : "val21",
  "key3" : "val31"
}

```

Figure 4.14: Data storage and insertion on MongoDB

REST API

In the book *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces*, Mark Masse explains REST APIs (Representational State Transfer (REST)) as a technical description of how the World Wide Web (www) works. Specifically, REST informs us how the Web achieves its great scale. If the Web can be said to have an “operating system”, its architectural style is REST. A REST Application Programming Interface (API) is a type of web server that enables a client, either user-operated or automated, to access resources that model a system’s data and functions.

REST APIs have spread rapidly and become dominant across the web, penetrating modern software development practices particularly on the mobile web where they allow for convenient data loading (Rodríguez et al., 2016). REST APIs are used for securely sharing information between two systems on the internet, sharing resources during authentication, and maintaining security and control. REST was key for the application in transferring data from the application screens to the database and back. In addition, it is easily integrated and a great technology to use on mobile applications (“REST API Developer Guide,” 2022). The main drawback of REST APIs is that they are unstructured which can be mitigated by more documentation.

Node.js

Servers work in various ways to serve clients with different data capabilities. Servers harbour data and make them accessible to clients over internal networks or over the

Internet. The main function is to respond to user or client requests to get the correct data. While MongoDB is the ideal open source NoSQL database system that is a scalable and high-performance database, Node.js is mostly used as its counterpart because Node.js is designed to make MongoDB simple to create quick and scalable network applications (Mithun Satheesh, Bruno Joseph D'mello, 2015). Saheesh and D'mello (2015) continue to explain that applications can then be created fast and easily using these two technologies, which can be uploaded to a cloud with minimum effort. Node.js is primarily used for server-side programming and back-end API services. It was intended for real-time and push-based architectures. Node.js was used for making calls to insert and retrieve data from MongoDB.

MacBook

The MacBook is a computer produced by Apple Inc. The Mac notebook is a brand of computers that use the macOS operating system (Wikipedia, 2022). It was needed in the deployment of the green crowd (Dilivari) mobile application for the use of Apple's Xcode. Xcode was previously discussed in this section as an Integrated Development Environment (IDE) for building applications that run on Apple devices.

Android phone/ iOS phone

There are numerous mobile application operating systems such as Symbian, Blackberry, Windows and Harmony. When developing an application, the developer must consider all these operating systems so that the application would successfully download, install and run on various devices. With all the available mobile application operating systems, two are leading the pack: Android and iOS (Jain and Sharma, 2013). With strong evidence suggesting that Android and iOS are leading in the mobile technology space, efforts were put in place to ensure successful deployment on mobile phones using the two operating systems. For testing two phones using the two different operating systems were used.

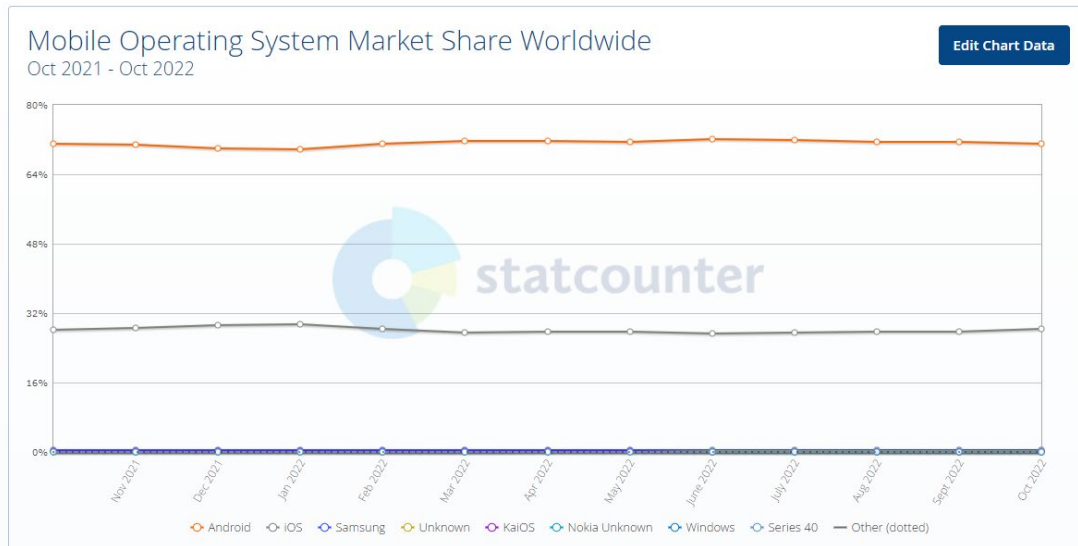
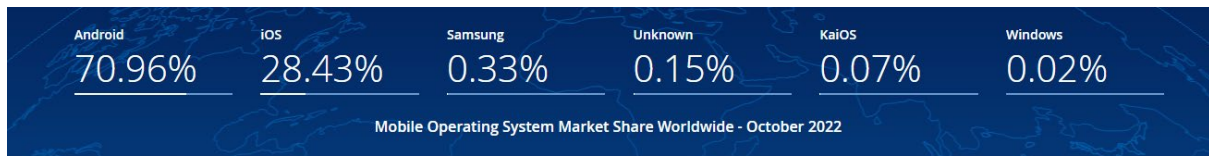


Figure 4.15: Mobile Operating System Market Share Worldwide (Source: <https://gs.statcounter.com/os-market-share/mobile/worldwide>)

Azure

Azure is a Microsoft cloud computing service by Microsoft for the purpose of application management using Microsoft data centres. It supports different technologies, programming languages, tools, frameworks and also offers software as a service (SaaS), platform as a service (Paas) and infrastructure as a service (IaaS) ([Microsoft Azure, 2020](#)). Azure is a front-end and back-end adhesive. The main function for Azure is hosting the MongoDB for storing the green crowd (Dilivari) mobile application data. There are many other cloud computing service providers. Azure was selected in terms of affordability and full security control as compared to other well-established cloud service platforms such as Amazon Web Services ([AWS](#)).

4.2. CHAPTER SUMMARY

This chapter presented a hybrid approach to two software development models, the waterfall and agile development model. Both models were used for the system design and implementation of the green crowd (Dilivari) mobile application. The agile model was used to support waterfall through the development cycle and decision making

using certain agile tools for development and the structuring of the architecture. There are other unmentioned tools such as Android Studio and Firebase that were used on the project but as the discussions, research and iterations went, they were discarded. Evidence from this study supports the notion that it is possible to employ different models in software development based on the need, and that at the time a specific model may be inadequate. Additionally, for future studies more time can be invested in comparing available technologies to get better performance, cost efficiency, security, reliability and other characteristics that can be identified and measured in selecting the best technologies for the necessary software to be developed.

Drawing inspiration from the importance of performance and pleasant visuals on mobile applications, more significance was placed on tools that detail the requirements and ensure an aesthetic visually appealing application to enable system design, implementation and testing. Testing and maintenance as the next phase of the development lifecycle are covered in the next Chapter 5.

CHAPTER 5: SYSTEM TESTING AND RESULTS ANALYSIS

Testing forms part of the software development life cycle (SDLC). It is defined as the process of running and evaluating software with the intention of finding bugs (Lawanna, 2012). Generally, the focus is on evaluating a characteristic or capability of a software or system and validating that the specification requirements have been met. Testing is done to ensure that software sent to users is in line with requirements to avoid pushing to production software that will not serve its intended purpose or software with bugs. Lewis (2008) explains testing as being part of every phase of the software life cycle, serving different purposes but falling into a spectrum of four: unit testing, integration testing, system testing and acceptance testing. Unit testing is done mainly by developers as it is at a low level of testing mainly focusing on testing each unit as a piece of a puzzle. Units, modules or components are tested separately (Albarka Umar, 2020). Integration testing is performed when components are combined into a larger unit (Ahmed, 2009). For system testing, the objective is end-to-end quality of the complete system; the functional requirements are mapped for system testing. Non-functional requirements like reliability, security, and maintainability, are also verified (Lawanna, 2012). Acceptance testing is done by the users after the software is completed and handed over. The objective is to give confidence that the software is working than finding bugs (Lewis, 2008). For this study, a 'lab-based' approach was used limiting the testing to test automation, location simulation, connectivity related testing, integration testing, performance testing, security testing and engaging with peers mainly for the user experience. Additionally, in software development peer testing assists in providing valuable benefits such as improving quality and increasing collaboration (Clark, 2004). These are factors that contributes in adding peers to the testing phase for quality measures. Because system testing has the main objective of evaluating whether the complete system meets its specification by observing the system behaviour which includes, though not limited to, verification of functionality, security testing, performance, usability and accessibility testing (Slack, 2011). Based on the objectives covered by systems it was chosen as the testing approach to be undertaken for this study.

5.1 SYSTEM TESTING

The statistical significance for the use of mobile applications nowadays is evident in different sectors across the globe. Mobile applications are used for all kinds of activities such as booking flights, paying electricity bills, and email allowing one to work while on the move. Based on the task at hand or the urgency for the mobile application, user expectations differ (Vijiyalakshmi, 2016). A user can expect a fast response from the Uber application to book a ride, another user can expect ease of use when trying to make an urgent work report submission. As mobile phones continue to evolve, so do the applications and the uses. While developing a successful mobile application not only entails an aesthetically pleasing application, the users' acceptance and experience of the application determines the success of the application (Akanmu *et al.*, 2013). Important aspects that are a measure of the application's success must be considered before it is sent to the public; these aspects include test automation, location simulation, engaging with third parties, understanding physical characteristics, user experience, connectivity-related testing, dealing with fragmentation, end-to-end integration testing, performance and security (Knott, 2015). According to Knott's (2015) book on testing, these success factors can be grouped into five segments, discussed below.

5.1.1. Functional testing

The objective of functional testing on mobile applications is to validate the actual functionality of the application (Ali and Arif, 2020). The testing is done by drawing up use case models that are documented and walked through each step at a time to compare its actual results with the expected results. To achieve even better results, this form of testing can be coupled using exploratory testing, where the tester is given more freedom to explore beyond the defined use cases. Following this approach and adding an unconventional agile touch on testing, the functionality was tested in fragmented units within end-to-end integration testing. An example was testing the map functionality as shown in images below (Figure 5.1). The map functionality was a fragment on its own yet part of the big picture by creating a delivery request as shown in the second image showing 'Requests'. Other fragments that were tested included creating a user, logging in, capturing request details and putting in an offer to make a delivery.

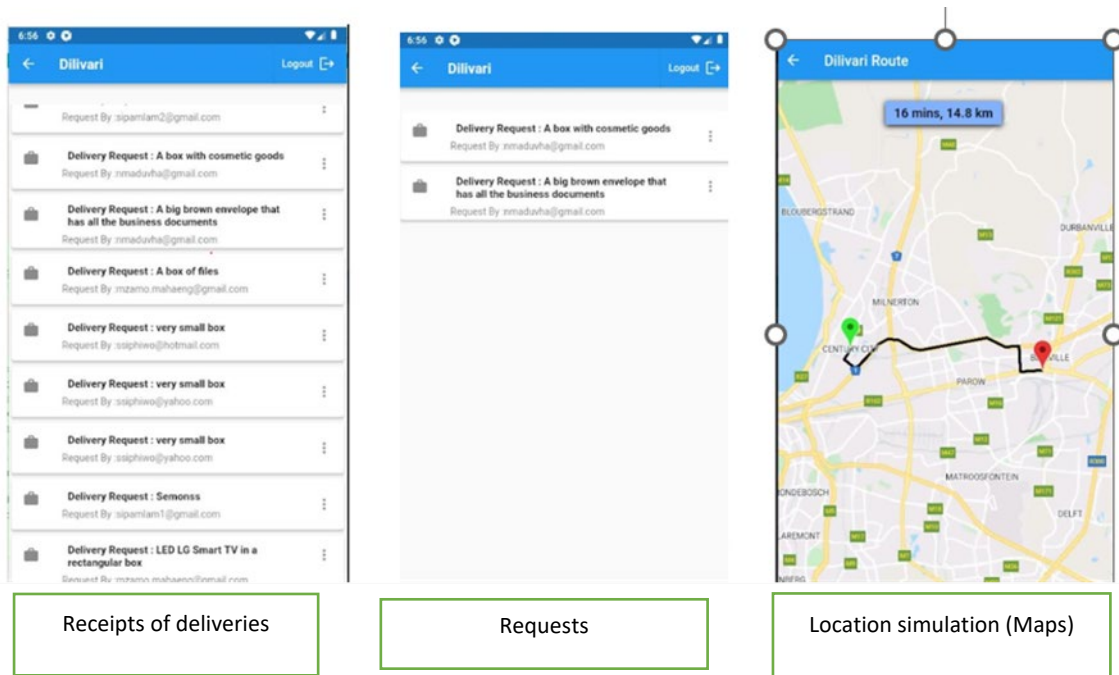


Figure 5.1: Functional testing of the application – the map functionality showing receipts of deliveries, requests and location simulation (Maps)

In compiling the test results, a checklist of the testing functions was created, as shown in Table 5.1. Based on the success of the test, the function would be marked as done; when failed it would then be sent back to the development phase to be fixed or enhanced.

Table 5.1: Functional test checklist

1.8.	1.8.2	1.8.3	Test Account	1.8.4	Steps Summary	1.8.5	Expected	1.8.6	Result
1.8.	1.8.8	1.8.9	<u>vincem@wwe.com</u> <u>password</u>	1.8.10	Login	1.8.11	Navigated to home screen Welcome Vince McMahon	1.8.12	Pass
1.8.	1.8.14	1.8.15		1.8.16	Tap on ' New Dilivari ' Populate : Package: Package Destination: Pickup address: Recipient: hhh@wwe.com Confirmation:	1.8.17	-The new delivery will appear under Requests. -The delivery requestor will never see their delivery request under 'Available Possible Deliveries' -All the deliveries will be seen under 'Receipts'	1.8.18	Pass
1.8.	1.8.20	1.8.21		1.8.22	Logout	1.8.23	Navigated to the login screen.	1.8.24	Pass
1.8.	1.8.26	1.8.27	<u>hhh@wwe.com</u> <u>password</u>	1.8.28	Login	1.8.29	Navigated to home screen Welcome Paul Le Vesque	1.8.30	Pass

1.8.	1.8.32	1.8.33	1.8.34 Tap on 'Assigned'	1.8.35 The delivery request from delivery requestor is displayed.	1.8.36 Pass
1.8.	1.8.38	1.8.39	1.8.40 Tap on the home button. Tap on 'Available Deliveries'	1.8.41 The user will see ALL deliveries request that were <> created by the logged in user. .No confirmed deliveries will be visible here. Any delivery request can be taken by the user.	1.8.42 Pass
1.8.	1.8.44	1.8.45	1.8.46 Tap on 'Assigned'	1.8.47 View of the delivery request assigned in step 2.	1.8.48 Pass
1.8.	1.8.50	1.8.51	1.8.52 Tap on the delivery request	1.8.53 The map will draw a line for the source, destination, travel time and km for the delivery request	1.8.54 Pass
1.8.	1.8.56	1.8.57	1.8.58 Tap on 'Home' >> 'Available Deliveries '	All possible available deliveries are displayed. Tap on a delivery request	1.8.59 Pass
1.8.	1.8.61	1.8.62	1.8.63 In the Available Possible Deliveries screen. Tap on a delivery request {the delivery request that was assigned to the user previously}.	1.8.64 The offer Delivery screen is displayed. The following displayed: Parcel Box, Pickup address, destination address and offer delivery button is enabled.	1.8.65 Pass

1.8.	1.8.67	1.8.68	1.8.69	Tap on offer Dilivari button	1.8.70	The Delivery Transport Registration popup is displayed.	1.8.71	Pass	
1.8.	1.8.73	1.8.74	1.8.75	Enter the Registration of the vehicle offering the Delivery	1.8.76	Registration is populated as per input	1.8.77	Pass	
1.8.	1.8.79	1.8.80	1.8.81	Tap Submit	1.8.82	The offer Delivery pop-up disappears. Use returns to available possible dilivery. The delivery in the afore steps is no longer in the available in the possible delivery screen.	1.8.83	Pass	
1.8.	1.8.85	1.8.86	1.8.87	Logoff	1.8.88	Navigated to the email password screen	1.8.89	Pass	
1.8.	1.8.91	1.8.92	vincem@wwe.com password	1.8.93	Login	1.8.94	Navigated to home screen Welcome Vince McMahon	1.8.95	Pass
1.8.	1.8.97	1.8.98	1.8.99	Tap on 'requests'	1.8.100	The Offer Dilivery pop-up is displayed. The accept button is displayed.	1.8.101	Pass	
1.8.	1.8.103	1.8.104	1.8.105	Tap on the accept button in the Delivery offer popup.	1.8.106	The accept popup screen disappears.	1.8.107	Pass	

1.8.	1.8.109	1.8.110	1.8.111 Logoff	1.8.112 Navigated to the email password screen.	1.8.113 Pass
1.8.	1.8.115	1.8.116 <u>hhh@wwe.com</u> <u>password</u>	1.8.117 Login	1.8.118 Navigated to home screen Welcome Paul Le Vesque	1.8.119 Pass
1.8.	1.8.121	1.8.122	1.8.123 Tap on 'Assigned'	1.8.124 View of the delivery request assigned in step 2. The confirm delivery Button.	1.8.125 Pass
1.8.	1.8.127	1.8.128	1.8.129 Tap the sign button	1.8.130 The only place to see delivered. Even the confirmed deliveries.	1.8.131 Pass

1.8.	1.8.133	1.8.134	1.8.135 Tap on 'Receipts'	1.8.136 The delivery will be in this screen. The delivery is <> under available deliveries. For the delivery assignee, the delivery is still populated under 'Assigned' [enhancement]. For the delivery requestor, the delivery is still populated under 'Requests' [enhancement].	1.8.137 Pass
------	---------	---------	---------------------------	--	--------------

5.1.2. Compatibility testing

Mobile device compatibility testing is done to overcome the challenge of testing hundreds of different devices to ascertain compatibility for each of them (Ali and Arif, 2020). Such testing is done by listing a few sub-sets of devices that can be physically tested, alternatively automation is put in place to automate the testing simulating the testing which would be done by a tester per device and automatically doing testing on multiple devices at once. For this study, a majority of compatibility testing was covered by users using different cellular models. Additionally, software development kits have a simulator functionality that allowed for testing on different models as shown in Figure 5.2 with a yellow arrow showing testing on an iPhone 12 model.



Figure 5.2: Simulator testing

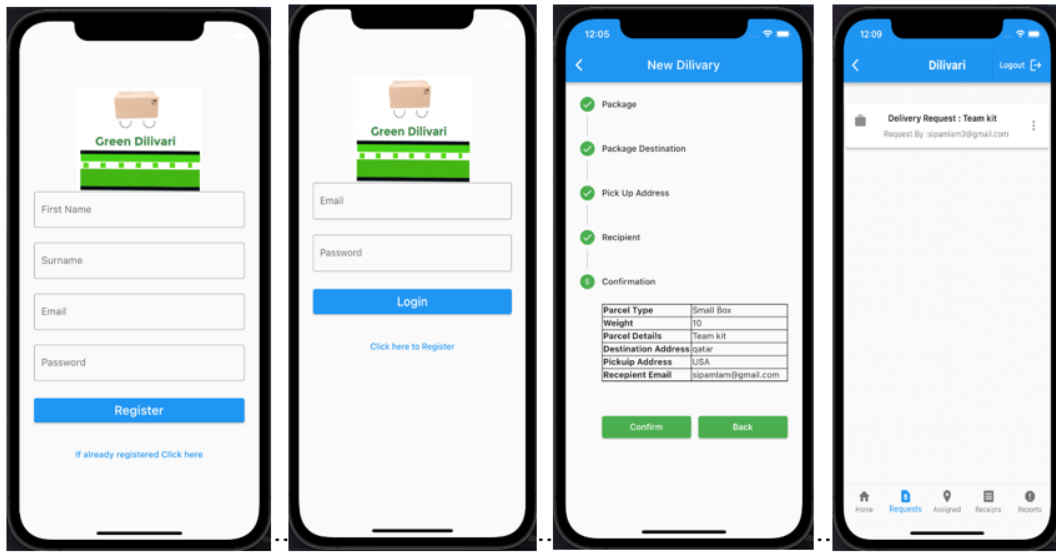


Figure 5.3: Simulator testing for registering, log-in, request and deliver

Further compatibility results were obtained from users with their different mobile devices. From the user responses gathered, the application is compatible to mobile phones running an Android operating system with a version 11 or higher. Figure 5.4 is extracted from a user report upon testing for compatibility.

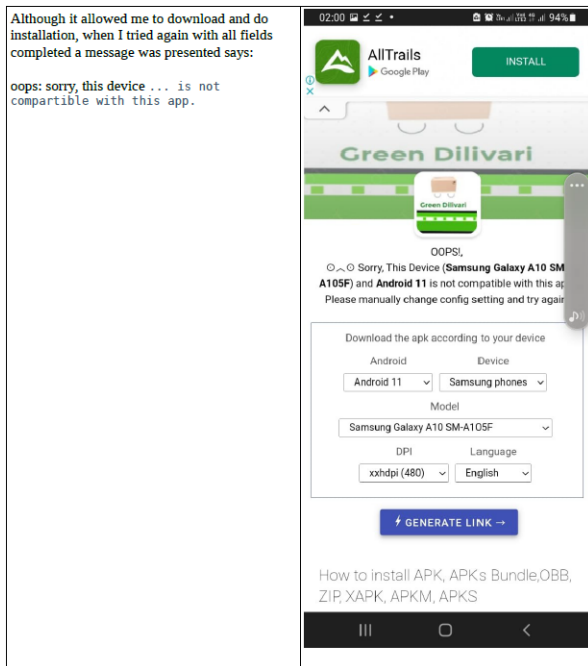


Figure 5.4: User report on compatibility

In conclusion, not all mobile phones using different operating systems and having multiple versions of each operating system were tested; a sub-set of devices was tested during the compatibility testing.

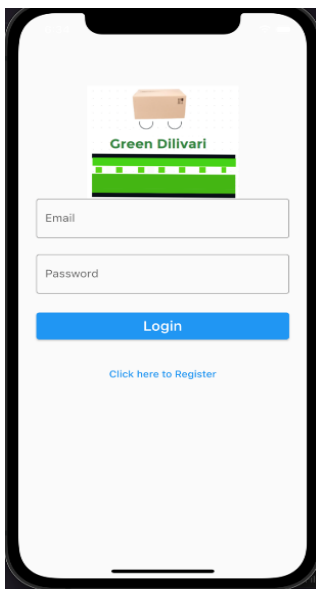
5.1.3. Performance testing

Performance testing is used for validating an application's architecture. It evaluates all layers of the application from the front-end, the middle layer to the back-end of the technology stack (Knott, 2015). This kind of testing also measures the application's performance across different mobile phone models covering whether old or new. Conventionally, this kind of testing functionality and usability are overlooked. As mentioned in the previous chapter, the Dilivari application architecture was defined as having the following components: a user interface (UI) that is the application screens of the mobile application; and a data model layer that defines the data structure, constraints and API calls to the final component, the database. On testing performance, testing was conducted in units and then integrated testing was done to validate the seamless integration of the architecture components. Details of the performance testing are captured below for each component performed.

5.1.3.1. User interface testing

Graphical user interface testing is done to verify that the application renders the anticipated output in response to the sequence of actions performed on a device. The user interface was carried out in the development phase as unit tests during the addition of each button, a new screen, a navigation button, and any action that could be done by the user. The entry point for further testing of the middle layer through to the back end was still the UI; however, the point of interest would be the back-end or the capturing of the user inputs on the application.

The screens below are a sample of the testing done on the UI:



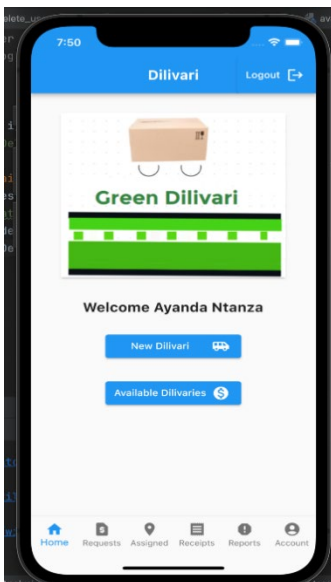
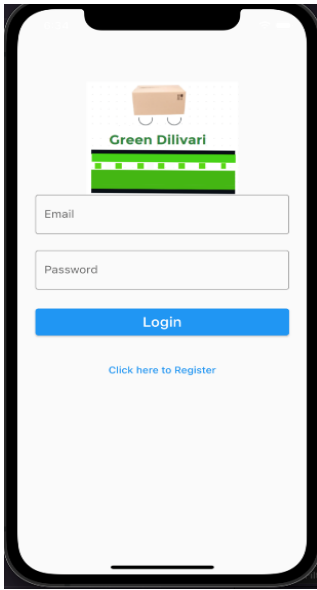
Sign-up Screen

- Testing was done to check that the user was able to write, delete and edit in the text field as well as toggle between text fields.
- Testing was done on the 'Register' button to check whether it was clickable and able to navigate to the correct screen once user inputs were added.
- Below the register is a clickable link that was supposed to land on the login; it was also tested.



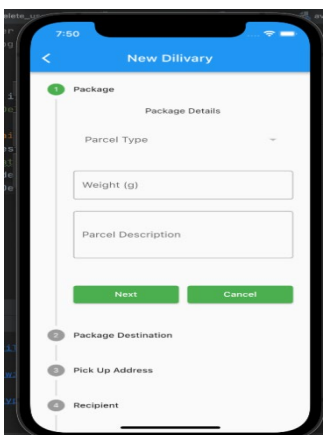
Login Screen

- Testing was done to check that the user was able to add inputs on the email and password text boxes, and able to navigate between fields.
- Testing was also done to check that the 'Login' button is clickable and led to the home page if the email-password combination was correct.
- Testing was also performed to verify that the 'Click here to Register' button was clickable and navigated to the 'Register' page.



Home Screen

- Testing was done to verify that the correct user's name ID displayed upon login.
- Testing was done to check that the two buttons to create new delivery requests and the button to check already available deliveries were clickable and navigated to the correct screen.
- Testing on the footer buttons confirmed that they were all clickable and navigated to appropriate screens.
- Testing verified the 'Logout' navigation button at the top terminated the sessions and navigated back to the login screen.



New Delivery Screen

- Testing checked that the flow worked and the navigation button helped move to the next and previous steps.
- Testing verified that upon entering an address a dropdown list of suggestions appeared as an auto-complete to the address.
- Testing also vetted that the 'Confirm' button submitted the delivery request.

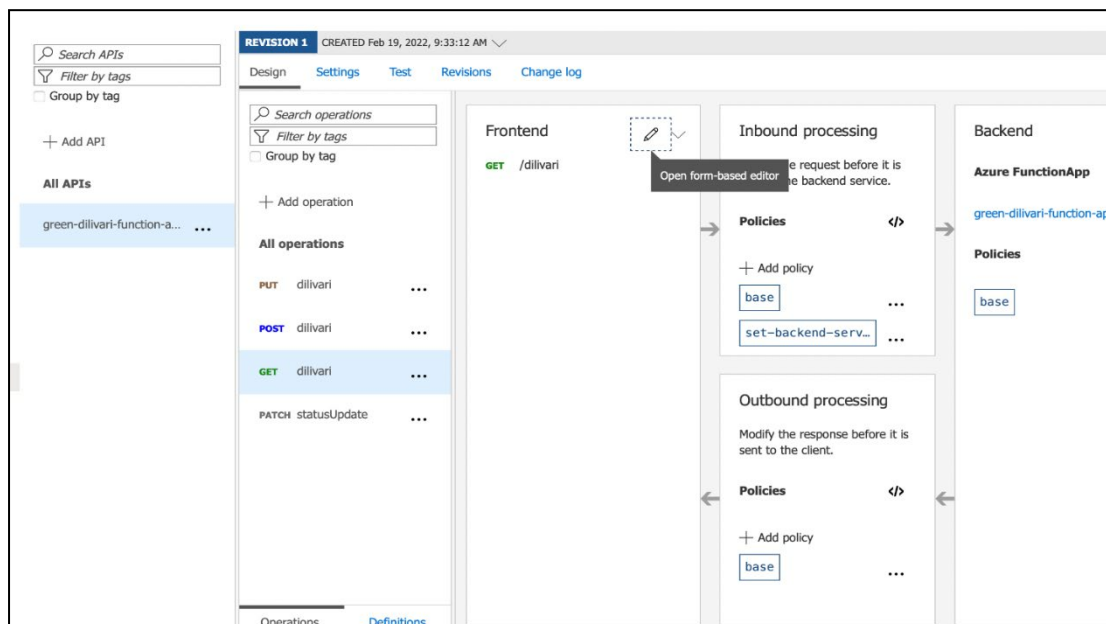
Figure 5.5: Sample testing on UI

A similar approach of testing was completed for the rest of the screens verifying that the sequence of actions performed by the user yielded the expected response from the UI.

5.1.3.2. Middle layer testing

The middle layer is a link between the UI and the back end – the database. This layer exposed the REST API that front end used to interact with the database. This layer accepted HTTP requests from the UI and utilised CRUD (create, read, update, delete) operations connecting to the database to create data and perform data manipulations, to query and get results, while also processing with those results. The middle layer was spread across the code and Microsoft Azure. The REST API endpoint and operations were housed in a Microsoft Azure function app that has the functionality to enable API management. The starting point was to test the REST API as a separate component by testing the API operations. Following Microsoft Azure the testing was conducted for the below operations:

GET – This operation was used for making queries and retrieving data.



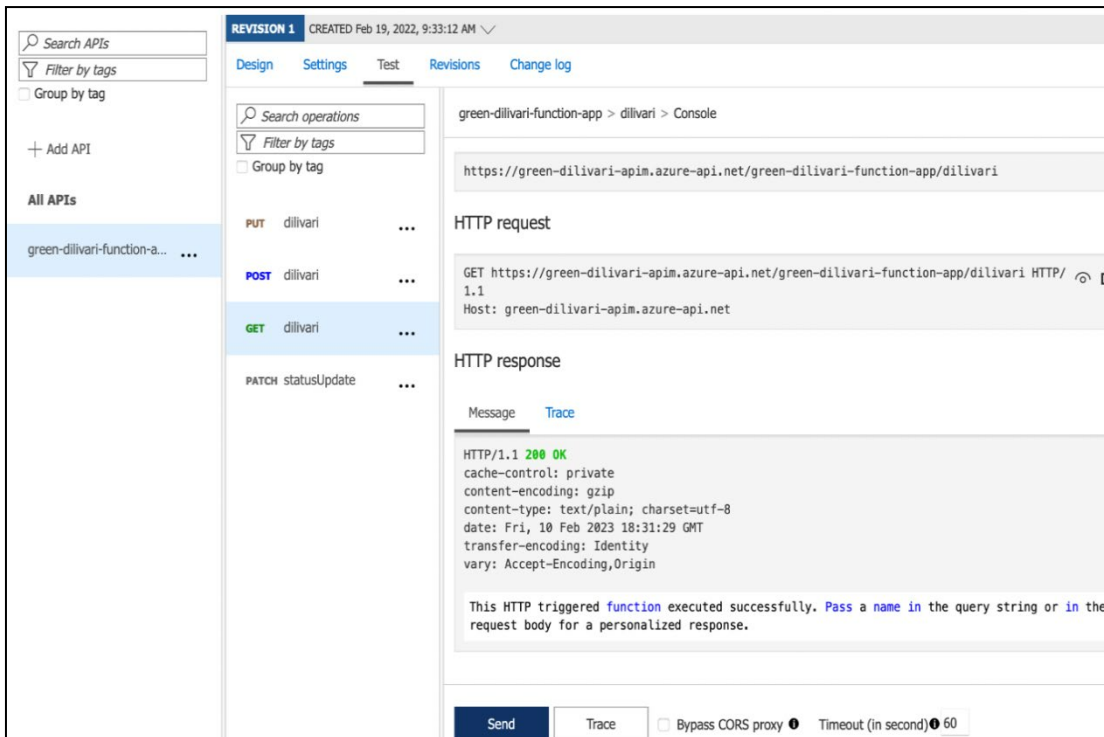


Figure 5.6: GET testing on Microsoft Azure

POST/PUT – The post and put were tested interchangeably; the PUT was meant for creating a new resource or updating an existing one, while the POST was meant to create a resource.

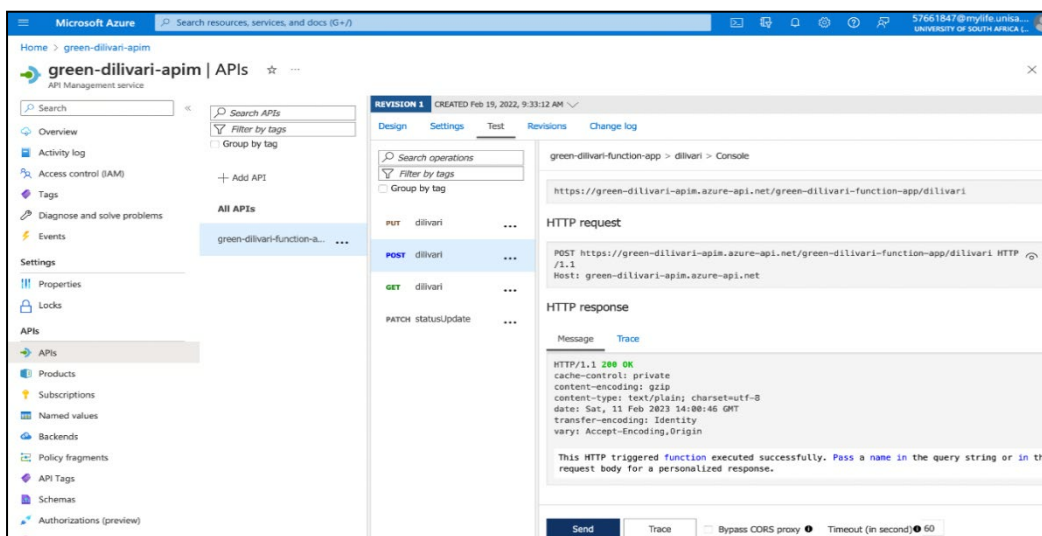
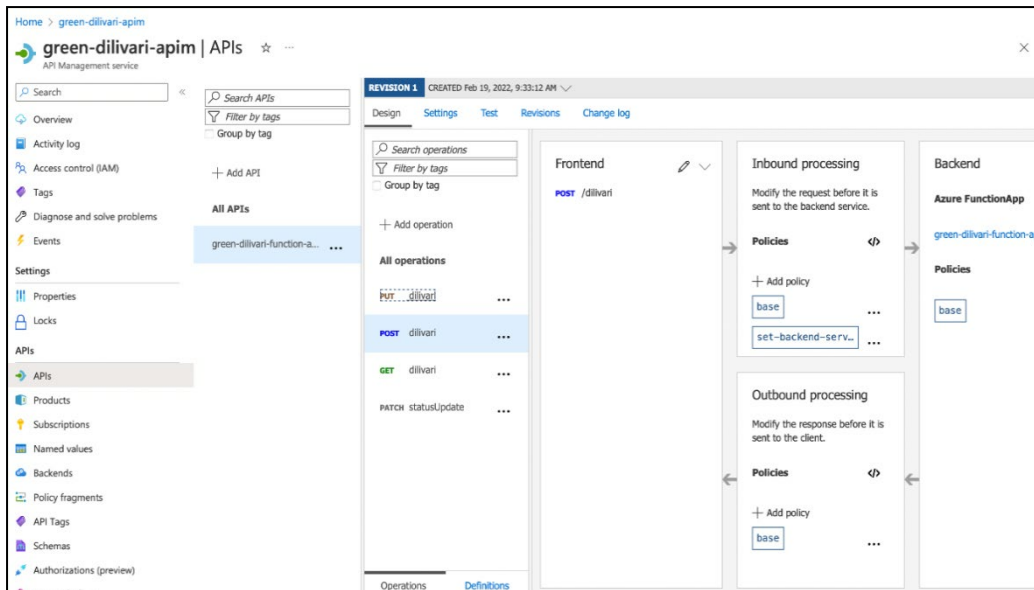


Figure 5.7: POST/PUT testing

Further testing was done via UI, where user accounts were created with API calls to insert data. Additionally, delivery requests were created and verified from the UI to the API and database.

```
1 {
2   "_id" : ObjectId("6218dace305c7c0034af6842"),
3   "name" : "wri",
4   "surname" : "eyu",
5   "email" : "witty@s.com",
6   "password" : "$2a$08$uwsR4rmBQ/GVwfQganjAq0i0JqIiv13Wn9gXQ4EHU5c7Cr
7 }
```

```
1 {
2   "_id" : ObjectId("622a706885285900383dd4de"),
3   "parcelType" : "Small Box",
4   "weight" : "14",
5   "parcelDetails" : "Beans",
6   "destinationAddress" : "worces",
7   "destinationLatitude" : -33.65389,
8   "destinationLongitude" : 19.4616074,
9   "pickupAddress" : "Paarl, South Africa",
10  "pickupLatitude" : -33.7342304,
11  "pickupLongitude" : 18.9621091,
12  "recepientEmail" : "sipamlam@gmail.com ",
13  "capturedby" : "sipamlam2@gmail.com ",
14  "deliveryId" : "4cd3102c-7c04-42d5-b92e-3b873eb00ff5",
15  "status" : "Offered",
16  "registrationNumber" : "CAA12356"
17 }
```

The screenshot on the left is an API JSON object for a user registration. This object was viewed from Microsoft Azure as part of testing to verify that the create user request from the UI went through successfully. The second screenshot was an API object of a delivery details.

Figure 5.8: Testing the UI to the API and database

5.1.3.3 Back-end testing

In order for an application to perform optimally, the database is a key component that must be fully functional. Should the database have issues, it could lead to data inaccuracy, incorrect storage of data and loss of data. As such, database testing was very important.

The process of testing a database normally involves checking schemas, tables and triggers. But MongoDB, as discussed in Chapter 4, stores data as documents stored in collections which is different to the MySQL database which is table based. Because of this reason a different testing approach had to be used. The following was done as part of testing:

Checking that the correct collections were created.

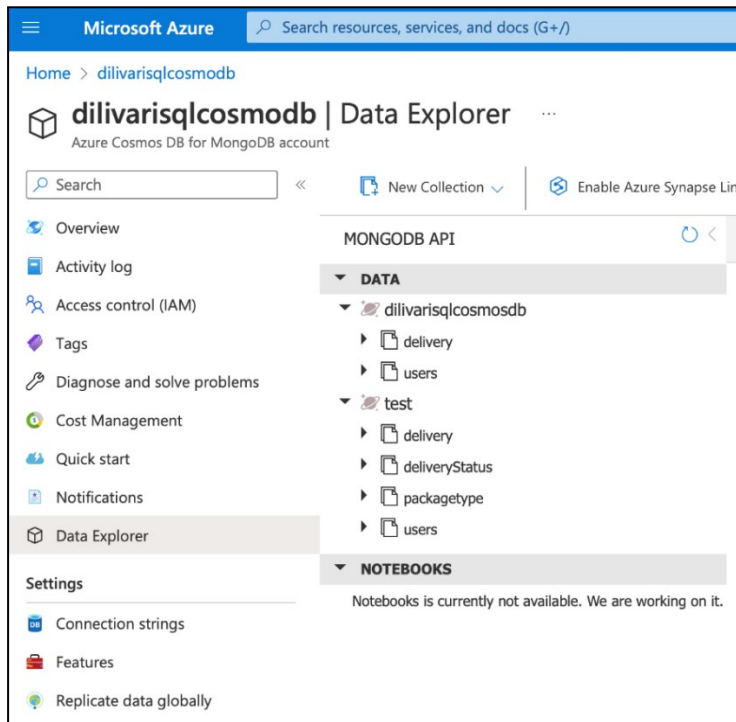


Figure 5.9: Testing the creation of correct collections

Because the collections represent tables, testing was conducted to validate that the collections were properly created and tested by sending requests from the UI. From the signup stage, a user profile was created and stored under the “users” collection. When a user creates a new delivery request, the data is sent to the delivery collection. Validation and checking to confirm that the collections were created and exist in the database level, was the outcome.

Validating that the data was stored in the correct format

Several users were registered from the UI and manual validation was performed to verify that the data sent was captured accurately on the database.

Below is an example of the data sent from the UI as the user was created:

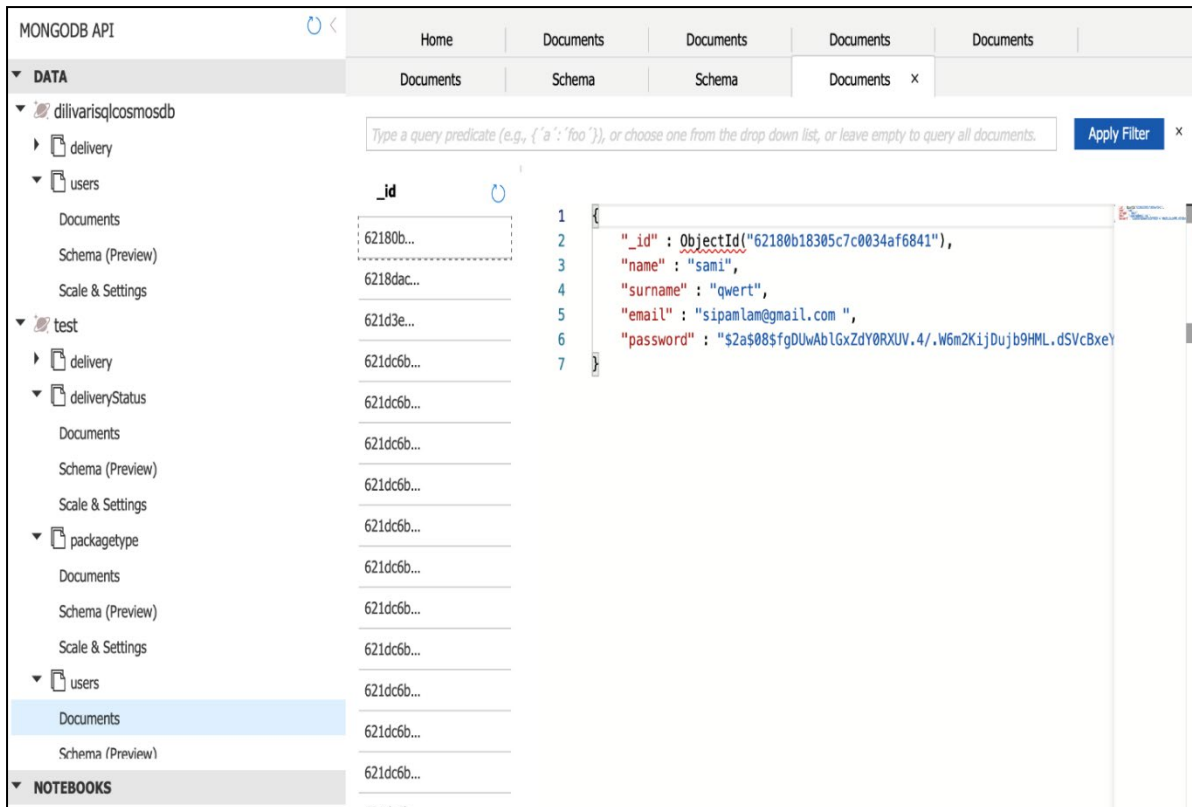


Figure 5.10: Data sent from the UI

Below is a collection of the user details:

```

{"_id": "621d3e88305c7c0034af685a", "name": "mina", "surname": "nave", "email": "mina@nave.com", "password": "$2a$08$a42.5Q6m07D7LXCq/wZo.DoEnFanblpylidsj6wz6rhhxkBV5jNHs."},
{"_id": "621dc6bb85285900383dd447", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$/7Q3DiIxYKsR3SQ.OsuZ/OcLEw0/fnl8Yre49h0M3orLLuoq10Na"},
{"_id": "621dc6bb85285900383dd448", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$0MUKI619TSELW0nDach17egJZ/BaPge/pjrdKuSz24GNF9XSDJv2"},
{"_id": "621dc6bb85285900383dd449", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$bnby8Igs01tu27ExYyY6H.BxPZ00p/bG6MHnqLlNRA50780uSQWwMC"},
{"_id": "621dc6bb85285900383dd44a", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$1CgPT8HwpcWTvSLNKDh4W.3a9rzkRrLr5CdzpyrnAGmYlNnxVl7Qi"},
{"_id": "621dc6bc85285900383dd44b", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$0Yc7UfWcVlyJl7GY9fPzLebAgJhhlykIarI7yxsWDqWPBqDP83Du"},
{"_id": "621dc6bc85285900383dd44c", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$IAS07gKUKUD0HaHhGjehPu2t1h1MOq1RgrqMhcmf/h/g8z.2CYJ02"},
{"_id": "621dc6bc85285900383dd44d", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$zjuVdLtp0dYXDia5N4HONPt0.Q5jzp21RuhvW/lvQu8RS37RZw."},
{"_id": "621dc6bc85285900383dd44e", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$/legbea4brT33WyzizOYF.DEqFoAdiRtUEHUI/Rho4PK9.2N7o5fg"},
{"_id": "621dc6bc85285900383dd44f", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$0M1leLSyNRaxAb80/QzP3epCTw/r7WxUFkzr8Fhanzkt5EGGuVe"},
{"_id": "621dc6bc85285900383dd450", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$EzVzqVw01YHGoqxIPN.H/eHPNjz1/ZA2q01DqFXmDLWadvh5QsE/i"},
{"_id": "621dc6bc85285900383dd451", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$8PwrtLsJmDd03XQIh7A.G66//xrJAXE2.iI5oLQadumerhIpa6"},
{"_id": "621dc6bc85285900383dd452", "name": "Mzamo", "surname": "Mahaeng", "email": "mzamo.mahaeng@gmail.com", "password": "$2a$08$bpqIPUgLuawMqakagSVKMeJRIqcm9ppmg7/Mm2wZD0vPeLgtcQzq"},
{"_id": "621fd6a185285900383dd459", "name": "man", "surname": "lee", "email": "sipamlam@gmail.com", "password": "$2a$08$kpUz81BsGJz/0kojnz2xeYbTybZrN4vQA7BOkKT2IuaYevrsRBS"},
{"_id": "62231a3585285900383dd480", "name": "Pitso", "surname": "Mosimane", "email": "pmosimane@hotmail.com", "password": "$2a$08$zICKGX/4SMFzeW0jq50FKeh3E8PWQZEv2p3wLUGlG8fodrISZ.0Vvy"},
{"_id": "62233a8e85285900383dd487", "name": "lingo", "surname": "legend", "email": "sipamlam2@gmail.com", "password": "$2a$08$HPwaAhmmuCO/ueKlyMPLyurJ80V9hXDok13LixJZCVFyxtPj2B4Q6"},
{"_id": "62262ec785285900383dd4db", "name": "Nelson", "surname": "Maduvha", "email": "nmaduvha@gmail.com", "password": "$2a$08$8C5oPsvvqRwzVB9BZBu5kNk.Ymu9QWNFeZH9gYcEpsQEB20mQe"},
{"_id": "622853c385285900383dd4e4", "name": "Siphiwo", "surname": "Dzingwe", "email": "ssiphiwo@gmail.com", "password": "$2a$08$1CFAInzCz5CTToGU934QD.ebrNeCz3WEG/cJzc/rtIfQ89qRmUe"},
{"_id": "623a6fb985285900383dd507", "name": "Siphiwo", "surname": "Dzingwe", "email": "ssiphiwo@here.com", "password": "$2a$08$hoXQ.2bLFRMTJ/D3gqkvO.6Q6xKjbnRGK5CMeByGK3IzbdP21rIy"},
{"_id": "625f166e421a2400379dbe16", "name": "Mangaliso", "surname": "Sipamla", "email": "sipamlam3@gmail.com", "password": "$2a$08$Yc6YNUjsYKtulfeELBUDh.hPgSib1Q7Pjd8b1G7Lo3SMKGYSTXMEW"},
{"_id": "626a5ac6c3db0200347090cd", "name": "sipamlam3@gmail.com", "surname": "sipamlam3@gmail.com", "email": "sipamlam3@gmail.com", "password": "$2a$08$IscUSgs7Mmp/UNdTW1f.ge0cHTYKxT6Ybs7.OXafu0W0.lx89TrJy"}, {"_id": "6272022761d2a50039f059cc", "name": "Vince

```

Figure 5.11: Collection of user details

Below is an example of the collection of package types:

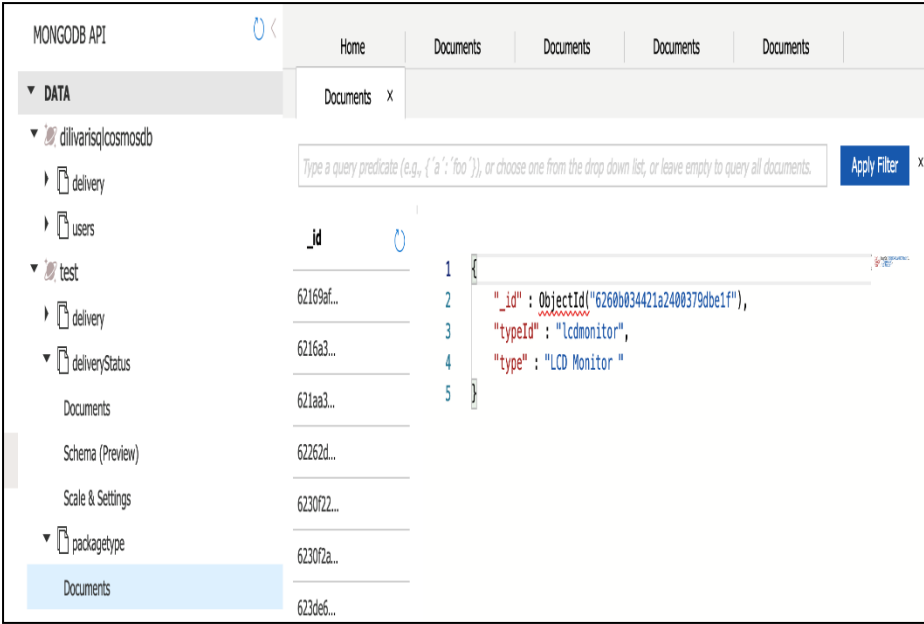


Figure 5.12: Collection of package types

Below is a sample of the 'Delivery' collection:

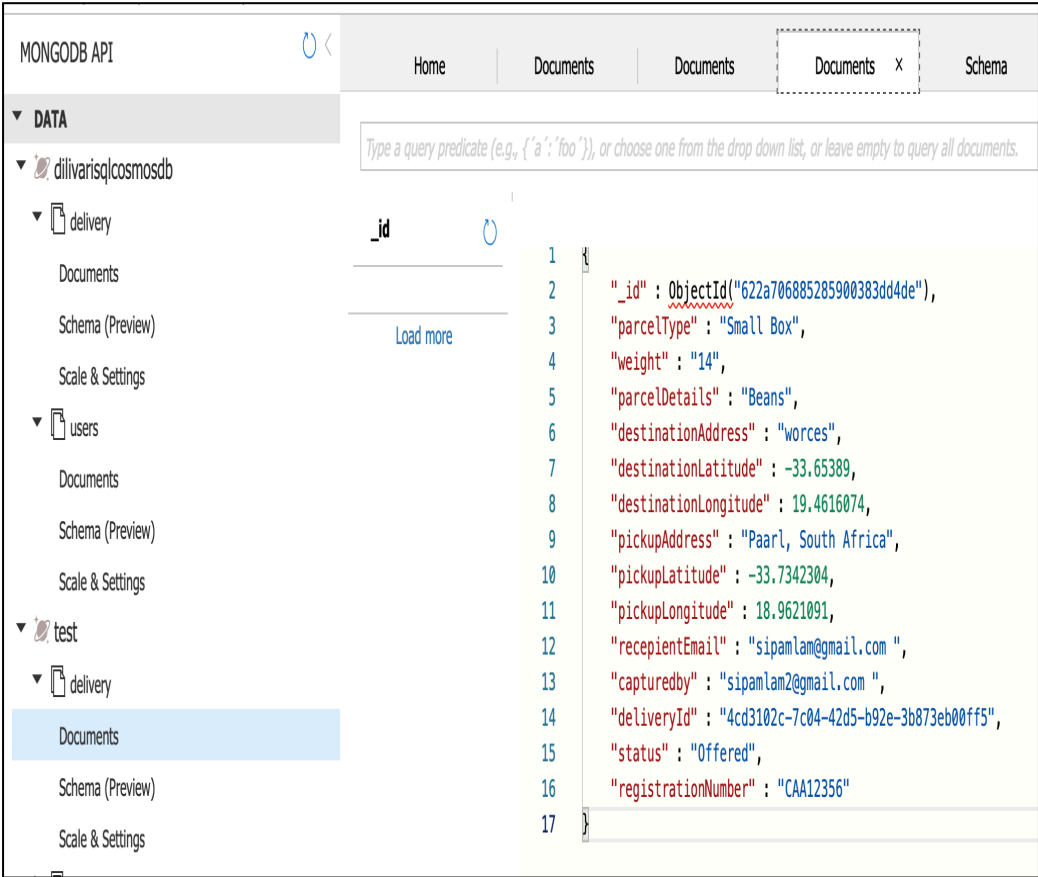


Figure 5.13: Delivery collection

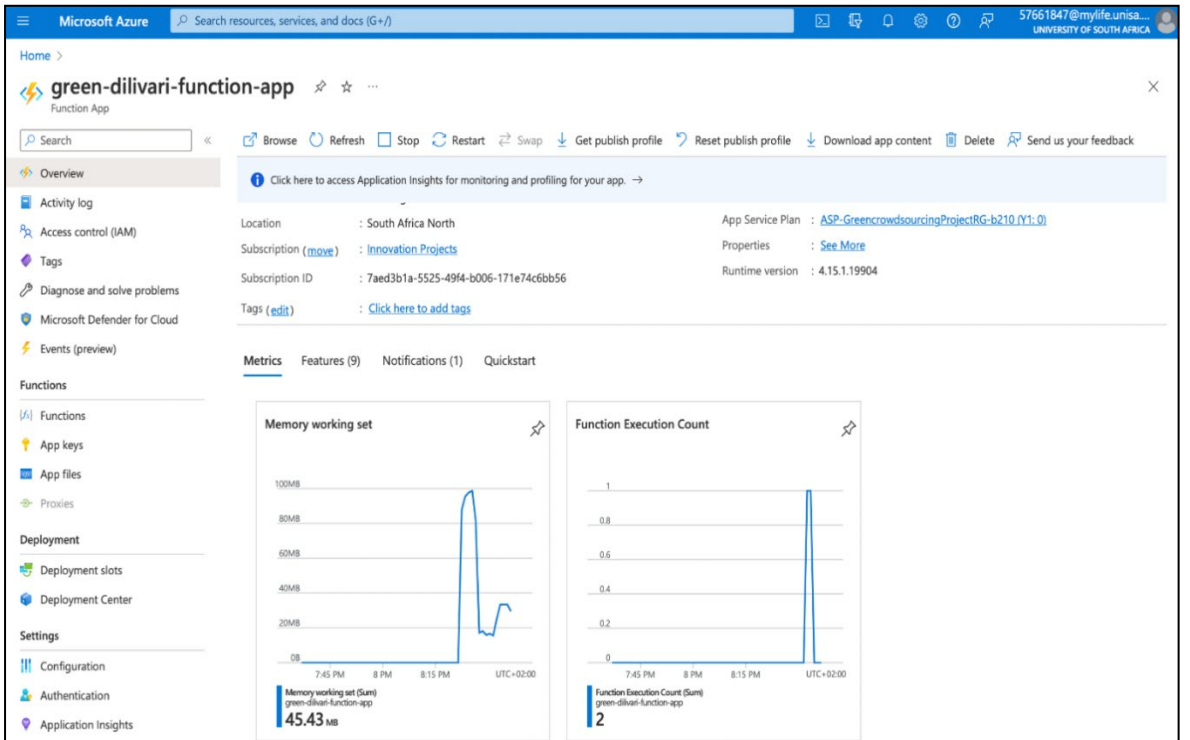
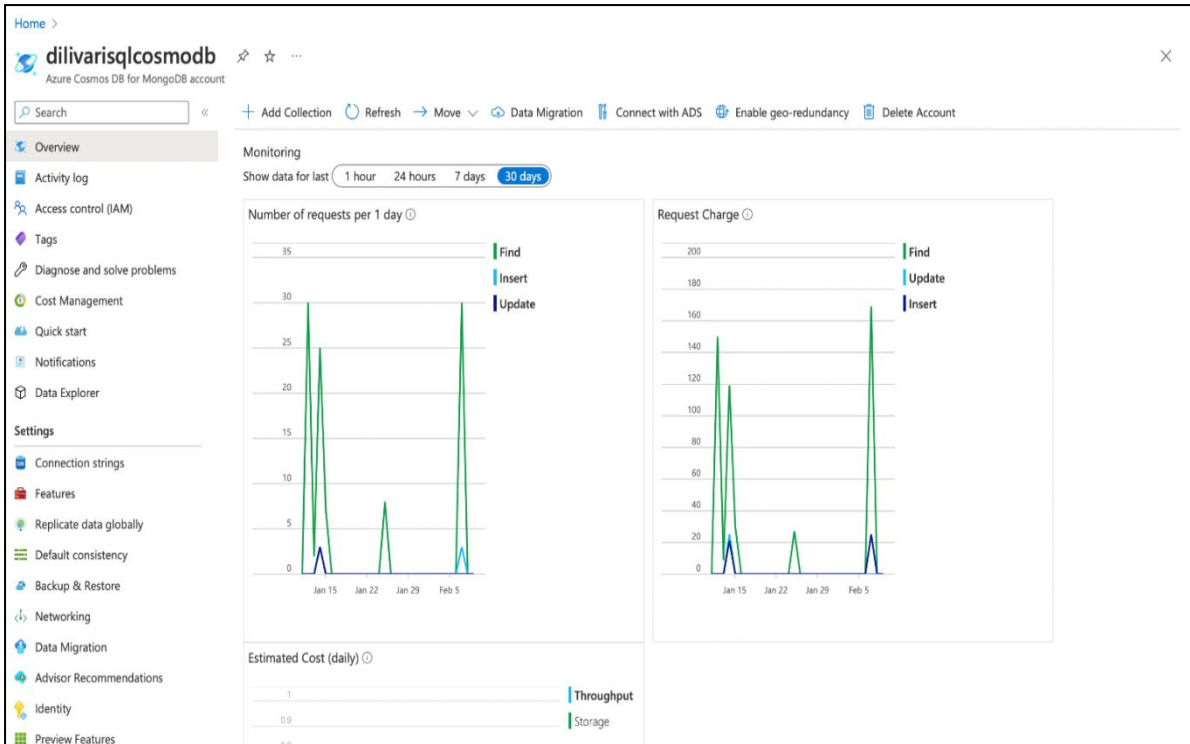
Additionally, testing was conducted based on one of the functionalities of the application. When a delivery request was created, it was created with the status “NEW”; following another user’s offer to make the delivery, the status was updated to “Offered”, and it changed to “Complete” upon being signed-off as delivered. The below images illustrate the status changes that were tested to validate that the updates were stored correctly in the database.

```
1 {
2   "_id" : ObjectId("6219dcb7305c7c0034af6844"),
3   "deliveryId" : "95a64078-101d-4975-bc65-f0572628c7e8",
4   "timeStamp" : 1645862071,
5   "status" : "New"
6 }
```

```
1 {
2   "_id" : ObjectId("622a706885285900383dd4de"),
3   "parcelType" : "Small Box",
4   "weight" : "14",
5   "parcelDetails" : "Beans",
6   "destinationAddress" : "worces",
7   "destinationLatitude" : -33.65389,
8   "destinationLongitude" : 19.4616074,
9   "pickupAddress" : "Paarl, South Africa",
10  "pickupLatitude" : -33.7342304,
11  "pickupLongitude" : 18.9621091,
12  "receptientEmail" : "sipamlam@gmail.com ",
13  "capturedby" : "sipamlam2@gmail.com ",
14  "deliveryId" : "4cd3102c-7c04-42d5-b92e-3b873eb00ff5",
15  "status" : "Offered",
16  "registrationNumber" : "CAA12356"
17 }
```

Figure 5.14: Validation of status changes to user updates

Stress testing is another aspect that was used in database testing. It is used to verify how the database copes under pressure or when there is a load of incoming requests at the same time. This kind of testing will be further explored in the next phase of the project, but it is worth noting that Microsoft Azure also has the capability of monitoring the database activity and the application performance as shown on the images below.



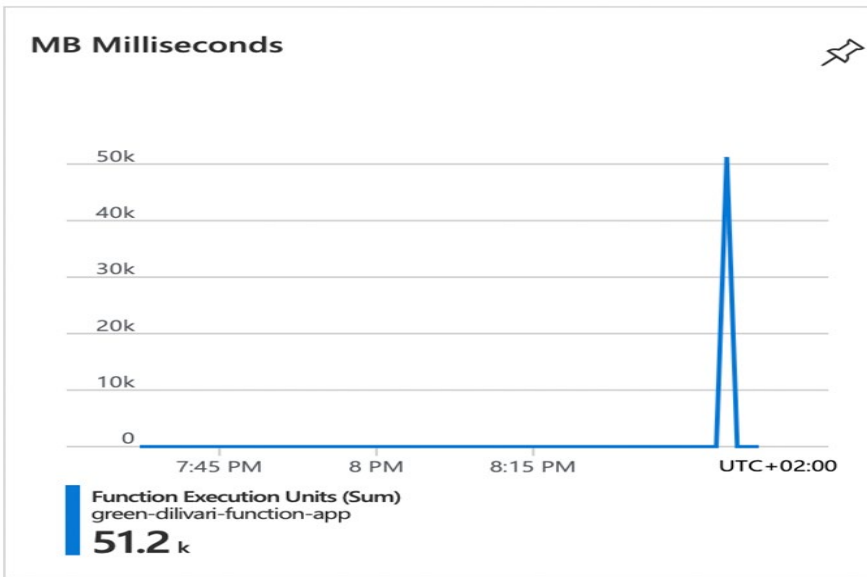


Figure 5.15: Microsoft Azure monitoring the database activity and the application performance

Further performance testing such as how the application performs in an area with limited network signal, stress testing and throughput testing will be covered in the next phase of the study. Further testing will also lead application management in terms of space and cost of space. Below is an image of another capability from Microsoft Azure to monitor and measure the throughput of the application.

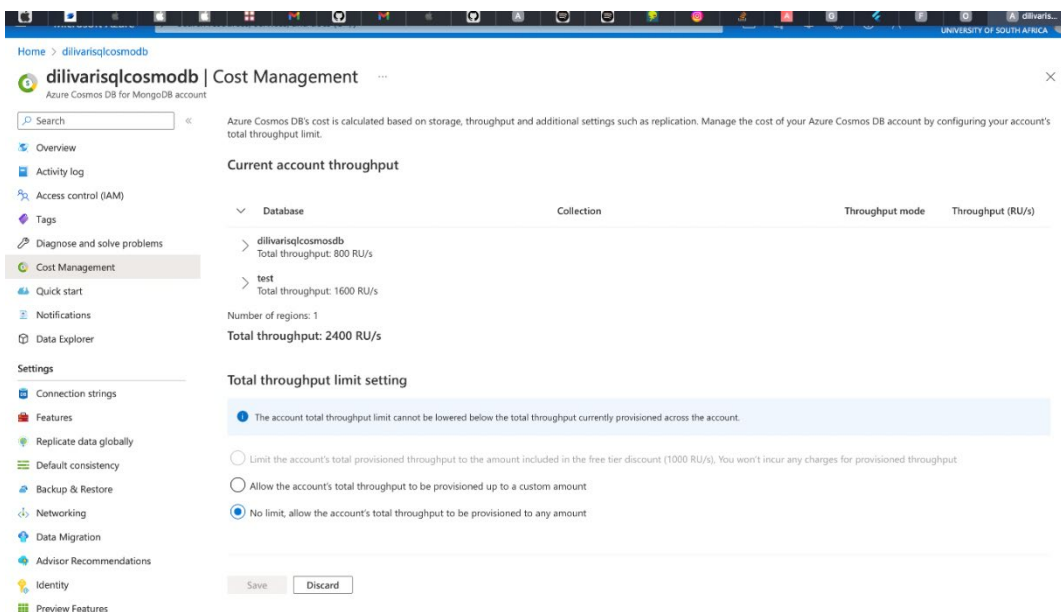


Figure 5.16: Microsoft Azure monitoring and measuring the throughput of the application

5.1.4. Confidentiality Security Testing

Security testing is a significant part of application testing; validating security measures uncovers critical security gaps including threats. Paul (2019) emphasises the importance of security testing highlighting the challenge imposed by hackers, most of whom are unethical. Many studies point to about six general security aspects that must should be tested (Okelo-odongo, 2014); these are integrity, confidentiality, non-repudiation, authentication, authorisation, and availability.

For security testing only, the log in function was tested to ensure that only registered users could gain access, through authentication testing.

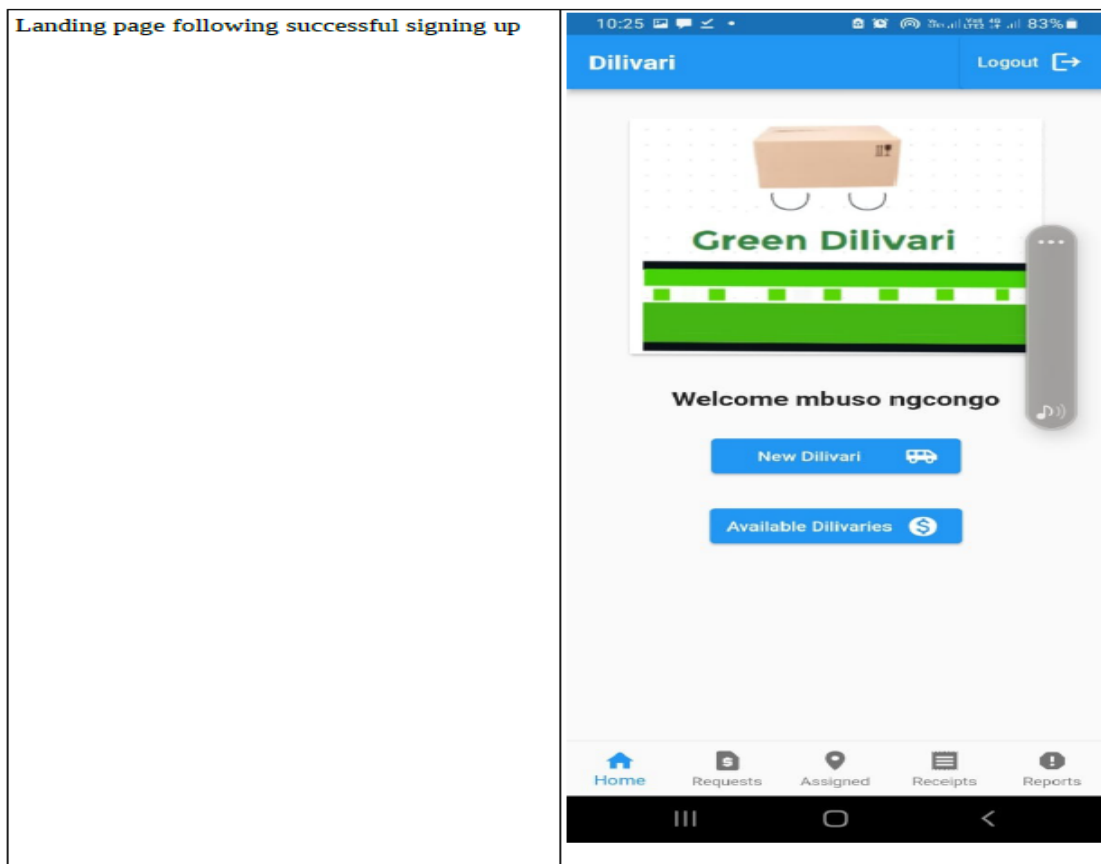


Figure 5.17: Testing of the log in functionality

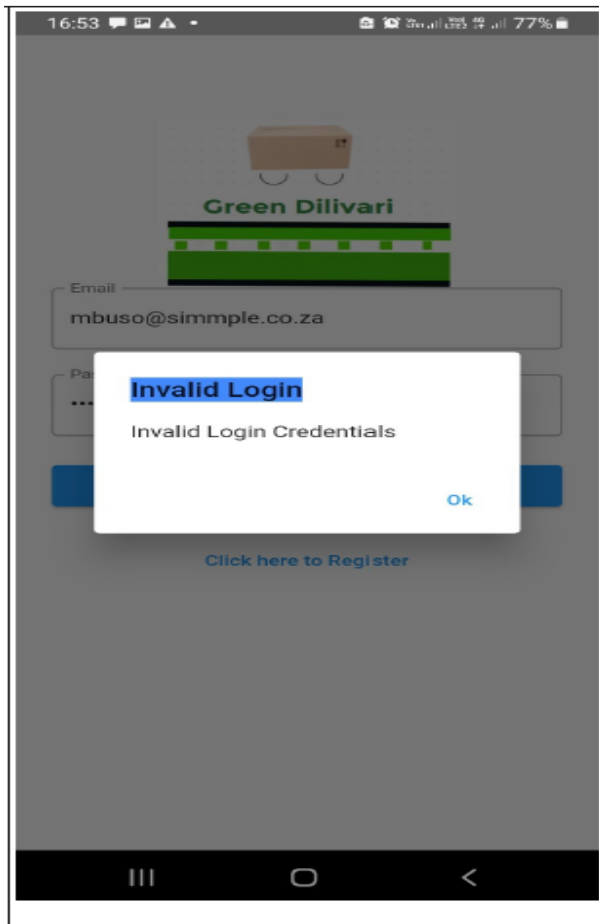
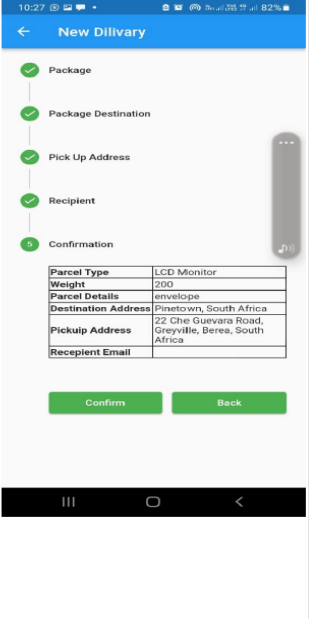
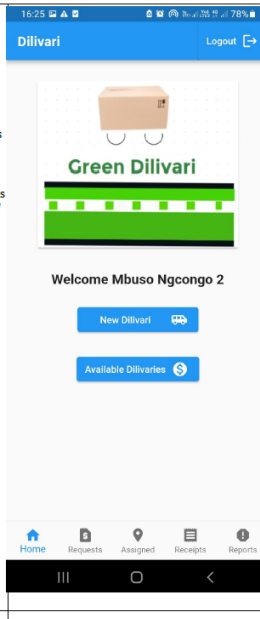


Figure 5.18: Testing email-password validation

For data integrity, more testing will be done in the next phase of the application where more data validation is set up from the development side. Availability, non-repudiation and confidentiality were also earmarked for the next phase of testing.

5.1.5. User experience testing

User experience determines the success or failure of an application making user experience testing very important. User experience testing validated that the application renders the desired user interface output responding to the sequence of the user actions. This testing checks how the application responds to different user inputs, such as menu bars, look and feel, toolbars, dialog messages and buttons. Because a lab-testing approach was used for testing, a few experts and stakeholders were given the opportunity to download, install, test or use the application to provide feedback on the experience. Below are a few responses from the user experience testing:

<p>Captured a New Dillivary</p> <p>pick up and delivery address correctly locates the address using auto complete, saving me trouble of typing complete address. I like this.</p> <p>I also like the fact it does not return a long list of addresses, just shows up to 3 items close to my input.</p> <p>I like this confirmation page, confirms all input of items. Maybe it is right moment to confirm price of delivery? And method of payment?</p> <p>Does not send verify email address. So i could have been a bored kid with data and actualy place order.</p> <p>location Notice this allows very generic address, which could potentially be a problem for working out cost and for actual pickup and delivery. Pinetown is a city, so to deliver in a city could be challenging when a parcel has to be actually delivered.</p> <p>I tested the app from Durban, and chose an address close to me (my neighbour's)</p> <p>scheduling There is no option to indicate when the parcel should be picked up, as my preferred time. If a user picks up this message and accepts it it may be a problem if the parcel should really be picked up a few days later.</p> <p>Clicking on confirm button nothing happens clicked several times until i was frustrated because i did not know what was happening. Was the delivery now in progress? Clicked on back and it worked.</p> <p>From a good practice design perspective: confirm should be on the right, and back should be on the left</p>	 <p>10:27</p> <p>New Dillivary</p> <ul style="list-style-type: none"> Package Package Destination Pick Up Address Recipient Confirmation <table border="1"> <tr><td>Parcel Type</td><td>LCD Monitor</td></tr> <tr><td>Weight</td><td>200</td></tr> <tr><td>Parcel Details</td><td>pinetowne</td></tr> <tr><td>Destination Address</td><td>Pinetown, South Africa</td></tr> <tr><td>Pickup Address</td><td>22 Che Guevara Road, Greyville, Berea, South Africa</td></tr> <tr><td>Recipient Email</td><td></td></tr> </table> <p>Confirm Back</p>	Parcel Type	LCD Monitor	Weight	200	Parcel Details	pinetowne	Destination Address	Pinetown, South Africa	Pickup Address	22 Che Guevara Road, Greyville, Berea, South Africa	Recipient Email		<p>New user was created for this test</p> <ul style="list-style-type: none"> * The password for user created previously could not be recognised by the system. * There is no "forgot password" option * When a username (email) exists within the system username was rejected, so user name has to be unique * Available deliveries icon is shown as \$. My understanding is users deliver and collect parcels and not necessarily money. Therefore an icon of a package/parcel is more appropriate  <p>16:25</p> <p>Dillivari Logout</p> <p>Green Dillivari</p> <p>Welcome Mbuso Ngcongco 2</p> <p>New Dillivary</p> <p>Available Dillivaries</p> <p>Home Requests Assigned Receipts Reports</p>
Parcel Type	LCD Monitor													
Weight	200													
Parcel Details	pinetowne													
Destination Address	Pinetown, South Africa													
Pickup Address	22 Che Guevara Road, Greyville, Berea, South Africa													
Recipient Email														

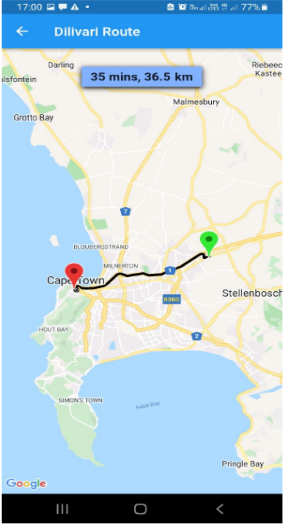
<p>After creating a new user, existing deliveries could be seen on the system</p> <ul style="list-style-type: none"> * 35 mins is misleading, emphasis should be on distance, maybe shown with following details <table border="1"> <tr><td>* 36.5 km distance</td></tr> <tr><td>* Package available on 30 May 2022, 16:30</td></tr> <tr><td>* Package dimensions: 0.4 x 1 x 1 m</td></tr> </table> <p>SPECIAL NOTES</p> <ul style="list-style-type: none"> * parcel will fit on an ordinary sedan vehicle * Urgent: required before 19:00 * Fragile glass 	* 36.5 km distance	* Package available on 30 May 2022, 16:30	* Package dimensions: 0.4 x 1 x 1 m	 <p>17:00</p> <p>Dillivari Route</p> <p>35 mins, 36.5 km</p> <p>Cape Town</p>	<p>General comments.</p> <p>The aesthetics of the application are pleasing. However, the colour co-ordination may need to be improved, for instance the choice of colours blue and which may not be good contrast for people with visual challenges. The font size for notes and for buttons in general is small. Some work is required to improve the aesthetics.</p> <p>The logic / workflow needs some work. More especially, constraints need to be in place to guide users to do the right thing and avoid making mistakes or missing essential information. This could prevent frustration from users.</p> <p>The program is not entirely deterministic, this is part of workflow work that is needed. For instance when in a screen the user should have option for yes and no. The user sometimes has to rely on back button built into the operating system to navigate their path, which may create frustration when user use different systems which allocate buttons differently.</p>
* 36.5 km distance					
* Package available on 30 May 2022, 16:30					
* Package dimensions: 0.4 x 1 x 1 m					

Figure 5.19: User feedback from the user experience testing

5.2. CHAPTER SUMMARY

Testing is a key measuring criterion for any software development process. It requires skill, time and understanding of the requirements. This chapter drew attention to testing and the findings. The findings were positive because the results exhibited and supported an achievement of the requirements and supported the drive to the next phase. It was, therefore, fortunate that several people partook in testing, and provided valuable feedback on possible improvements and changes to improve the application. A summary of the findings is captured in the table below. The only challenge was a

time constraint that will be factored in the next phase of the application and discussed in Chapter 6.

Table 5.2: Testing discoveries

Type of testing	Findings	Recommendations
Functional testing	<ul style="list-style-type: none"> • Sign-up and login functionality worked correctly. • Maps worked as expected in finding pick-up and destination addresses, route and estimated times. • Capturing and storing of delivery information worked as expected. • Creating a delivery request is working as expected. • Offering to make a delivery worked as expected. • Delivery was flagged accordingly per the different states of the delivery. 	<ul style="list-style-type: none"> • Functionality of viewing the password should be added to assist with the ease of access and checking misspelled passwords. • Maps should have boundaries added to assist in determining the out costs. • Scheduling is needed to add times for picking up parcels. • Confirm button should be on the right and back button of the left to follow good design perspectives. • Verification emails to be sent for each step of the parcel life cycle.
Compatibility testing	<ul style="list-style-type: none"> • Older versions of Android are incompatible with the application. 	
Performance testing	<ul style="list-style-type: none"> • To be done in the next phase of the study. 	
Confidentiality security testing	<ul style="list-style-type: none"> • Signup captured the details and correctly displayed user on login. • Validation worked correctly to enable access to correct username/password combination. 	<ul style="list-style-type: none"> • Functionality of viewing the password should be added to assist with the ease of access and checking misspelled passwords. • Email verification and confirmation functionality must be added to increase security.

<p>User experience testing</p>	<ul style="list-style-type: none"> • General look and feel were very native to Android. • Colours were coordinated nicely. • There was a sense of simplicity and consistency on the application. • Buttons did what they say they would do. 	<ul style="list-style-type: none"> • Special notes could be added to give more information about the parcel. • A cell number is more appropriate than an email address for a delivery. • The program is not entirely deterministic; part of the workflow work is needed. For instance, the screen should offer the user a yes-or-no option. The user sometimes has to rely on the back button built into the operating system to navigate their path, which may create frustration when the user uses different systems that allocate buttons differently.
--------------------------------	---	---

CHAPTER 6: SUMMARY OF FINDINGS AND DISCUSSIONS

This chapter presents a summary of the study, and exploration of the results followed by a discussion covering the objectives of the study. The findings will be detailed together with the achievements of the study.

6.1 CONTEXTUAL BACKGROUND FINDINGS

The aim of this study was to explore a digitally enabled distribution solution that understands, explains and implements a triple bottom line approach to the last mile distribution using technology, information systems and algorithms. This was initiated by examining existing literature aimed at achieving a triple bottom line, and how it could respond to the growing demand for an effective green last mile distribution solution. From literature, Mkansi (2013) is one such study that covered ground in paving the way and building a solid foundation for this study.

6.1.1 Green crowd mobile application (Dilivari) as a solution

This study gives evidence that a greener approach is feasible on the last mile distribution challenge of balancing the needs of the people, the business and keeping the planet into account. The digital age has exponentially influenced a shift in the way in which consumers do their shopping, from the normal in-store purchases to online purchases. With this shift, businesses are struggling to maintain and balance the trade-offs between the profit, people and planet. This study was an effort to address this challenge. A green digitally enabled mobile application was developed for making deliveries utilising users travelling in the same direction as the intended deliverer. With utilising a user travelling in the same direction, there is a positive impact on the environment because it eliminates additional means of making the delivery, minimises the negative impact of fuel combustion that results in air pollution, and subsequently generates greenhouse gases leading to global warming.

6.2 OVERALL AIM OF THE STUDY

The main purpose of this work was to explore how technology could be used to bridge the last mile distribution gap by developing a digitally enabled distribution model that understands, explains and implements a triple bottom line approach to the last mile

distribution with technology, information systems and algorithms, which are the key components to achieving a green solution. This aim was realised through the objectives outlined below:

6.2.1 Research objectives

These objectives were used to achieve the main purpose of the study:

1. Explore how the use of different technologies can contribute to alleviating the triple bottom line challenge of people, profit, and planet.
2. Create a database to facilitate the management of crowd data.
3. Compute an algorithm that uses artificial intelligence to integrate technologies that map and identify the crowd to retailers or businesses.

6.2.2. Research questions

The following research questions were utilised to achieve the overall aim of the study:

1. How can technology be used to alleviate the triple bottom line challenge of people, profit, and planet?
2. What techniques can be used for identifying, motivating and managing the crowd for the solution?
3. What algorithms are best suited to achieve optimal artificial intelligence of utilizing crowd logistics?

6.3. DISCUSSION OF THE FINDINGS

This section discusses the findings from Chapter 4 and Chapter 5 based on the research questions and concluded by a table with a summary of the findings.

6.3.1 Research question 1: How can technology be used to alleviate the triple bottom line challenge of people, profit, and planet?

An implication of global population growth is a negative impact on the environment. There are several factors that are meant to improve the quality of life from one generation to the next, such as industrial development and motor vehicles. However, these improvements have an underlying disadvantage that negatively impacts the

environment, endangering all living organisms, leading to the extinction of animals and plants. It is, therefore, in the best interest for all that scholars, professions and the global population develop with innovative solutions that will conserve the planet for future generations. This study is one such initiative aimed at balancing the triple bottom line between people, the planet and profit.

The application developed from this study gives evidence to the notion that technology can be used as a tool to bridge the gap between people, the planet and the all-important profit. Proven software development models offer a framework to work with in developing green solutions using technology. Chapter 4 detailed the tools which were used in this study; there are plenty more technologies being released to the public in this age of technology. In the initial phase of the study, the Android Studio integrated development environment (IDE) was used to create an android application. The challenges faced during development, integration and testing led to the decision to switch technologies from Android Studio to Flutter, and the programming language changed from java to dart. This decision was taken to halve the double effort of developing the application in Android and then again in iOS and having two code bases to maintain. Two software development models were employed for ensuring short sprints between development and testing after the shift in technology; the waterfall and agile models. With these two models, other technologies were also used to complete the application: Microsoft Azure Cloud Services, MongoDB, RESTful API, Google Maps API and Node JS.

The use of all those technologies highlighted the need for proper research on the available technologies which can be used to deliver a solution which would cater for people's needs and those of the planet while factoring the balance of making a profit. User feedback on the application evidence indicated people's familiarity with mobile applications and its usability without the need for guidance. This further supports the notion that applications are intrinsic to daily living; thus, their use in driving greener solutions is a realistically attainable goal.

This study is certainly a step in the right direction towards creating green solutions. Efforts such as this, and the work done by Mkansi (2013), need to be acknowledged for paving the way toward greener innovative studies and contributing knowledge on the last mile distribution as well as contributing evidence that technology can indeed

help create green solutions that would sustain our planet for future generations. This study lays a solid foundation for further studies on balancing the triple bottom line. Furthermore, this study adds on the understanding and the need for innovation to balance profit, people and planet.

6.3.2. Research question 2: What techniques can be used to identify, motivate and manage the crowd for the solution?

Each of the three Ps – people, planet, and profit – is concerned with various aspects. The ‘people’ aspect of the 3Ps is mainly concerned about health and welfare, social capital and the quality of life. The key profit factors include, among others, income versus expenditure and employment. The concern of the planet aspect is premised on natural resources, solid and toxic waste, energy consumption and water as well as air quality. There are many other concerns from each of the triple bottom line components. A sustainable last mile solution is one that would cater for all there; an example would be a solution which would reduce mileage, minimise emissions and carbon footprint without compromising business profits and the wellbeing of the people. There are a few different approaches and solutions. De Oliveira (2017) states that there is an effort to using lighter commercial vehicles for the last mile deliveries in urban areas. Such a solution reduces the use of fuel but, in contrast, fails as the demand increases the need for more vehicles to be on the road. It is on these drawbacks that this study aimed at achieving a well-rounded solution founded on using the coordination theory and information systems theory which will ultimately eliminate the empty kilometres traversed by last mile delivery vehicles after having made the deliveries.

Simatupang, Wright and Sridharan (2002) explains that “coordination” helps at managing interdependencies among activities done to attain a goal. This is translated to properly combining, adjusting and aligning information, knowledge systems, people, and funds for the achievement of making on-time deliveries on the last mile problem. Halawi and Mccarthy (2006) as well as Lerner (2004) stated a case for using the information system theory by explaining that it would bring an enhanced understanding of systems and the collection of information that would be transformed from being data without substance into models and useful forms of computer programs that would positively impact the “use” in addition to the user satisfaction. The combination of the

coordination theory and information systems theory into a mobile application provides greater opportunities towards balancing the triple bottom line. Using technology for greener solutions helps preserve the planet while also benefiting people by assisting individuals and business in revenue growth.

In conclusion, this study is evidence that a unique approach of combining two theories is cutting edge on the existing approaches that have been tried in addressing the triple bottom line on the last mile delivery. This study shows that technology can be used together with coordination and can motivate users as we are in the era where most individuals own a smart phone. The Green Dilivari application confirms the literature on how information systems can be used for greener solutions. This study's solution also supports a small business owner who may be unable to focus on all three aspects of the triple bottom line from the onset.

6.3.3. Research question 3: What algorithms are best suited at achieving optimal artificial intelligence of utilising crowd logistics?

The combination of information and information technologies with models such as computer algorithms help in transforming data into information and developing complex systems. Software development models, including the waterfall and agile development models, supported the design phase and, most importantly, the development phase on decision making such as identifying processes and tools used for development, the structuring of the architecture and facilitating the shift from one technology to the next when optimal results were unrealised.

On moving from the design phase to the implementation phase, several key features which were important on optimising the developed application to harvest optimal results. One key feature of the application is its ability to assist users on navigating between locations – the pick-up address and the destination address. On dealing with the locations and the process of identifying the shortest route with least traffic, integrating the Google Maps API tools helped with the development phase. Furthermore, the Maps API assisted in terms of identifying location and navigating between two locations ensuring that the directions are correct. The Google Maps API uses an algorithm to obtain information in order to calculate and decide on the shortest delivery route with the least traffic so as to reduce fuel consumption while ensuring

quick deliveries. Google Maps uses a Greedy algorithm that chooses the best choice without considering options that were previously taken. It relies heavily on the Dijkstra's algorithm for computing directions between two locations. Other algorithms are used within the application such as the search algorithm for logging in where the login credentials are searched within the database. Additionally, a similar approach to dynamic programming is used to store user profiles and details to avoid making database calls that would over-utilize resourcing and have a negative impact on the battery power and space of the mobile application.

Additionally, advances in technology such as the Dart programming language assisted in developing screens and adding a smoother flow for navigations while linking UI screens. Moreover, on housing the back end, where one would normally need to have servers to house a database and monitor transactions, space, security, integrity, stability and system availability, these responsibilities were removed using Microsoft Azure, thus, optimising the performance of the application while minimising the effort in the development work. Though Microsoft Azure shielded the responsibility to a degree, it still required an effort to set-up, configure and have all the required rights, groups and applications necessary to have a fully functional back end.

In a study on the potential impact of crowd shipping, Simoni et al. (2020) explain that crowd shippers are crucial for a sustainable solution for the last mile distribution problem. This study builds on such studies in addressing research question 3. The use of tools such as Google Maps aids in determining the shortest delivery route, with the least traffic so as to reduce the fuel consumption yet ensuring quick deliveries. The response to research question 3 similarly adds on the findings explored by Oliveira (2017) on the use of light weight vehicles. This study concurs that fewer delivery trucks will be a necessity while deploying a user already moving towards the destination within the same convenient route selected by the application would be beneficial to the triple bottom line.

Table 6.1: Summary of findings

Categories	Findings
<p>Objective 1:</p> <p>Explore how the use of different technologies can contribute to alleviate the triple bottom line challenge of people, profit, and planet.</p>	<p>The application developed from this study gives evidence to the notion that technology can be used as a tool to bridge the gap between people, the planet and the all-important profit. In implementing the Green Dilivari application, several technologies were used. In the initial phase of the project, the Android Studio integrated development environment (IDE) was used to create android applications. The challenges faced during development, integration and testing led to the decision to switch technologies from Android Studio to Flutter, and the programming language changed from java to dart. This decision was taken to halve the double effort of developing the application in Android and then again in iOS. Two software development models were employed to guarantee short periods between development and testing after the shift in technology; the waterfall and agile models. With these two models, other technologies were also used to complete the application: Microsoft Azure Cloud Services, MongoDB, RESTful API, Google Maps API and Node.js.</p> <p>The successful deployment of the application on both Android and iOS platforms resulted in the key finding that technology can be used to create greener solutions based on user feedback on the application, thus indicating people’s familiarity with mobile applications and its usability without the need for guidance. Furthermore, the use of two different software development models, agile and waterfall, is a key finding that while literature tends to pick one over the other, this</p>

	<p>study presented evidence that the two can be used to complement each other.</p>
<p>Objective 2</p> <p>Create a database to facilitate the management of crowd data.</p>	<p>The application of design science for digital artefacts is expensive and requires an interdisciplinary approach. For example, the problem under study is rooted in management studies, yet potential mediating solutions lie in computer science. This study aimed to achieve a well-rounded solution founded on using the coordination theory and information systems theory to ultimately eliminate the empty kilometers traversed by last mile delivery vehicle after having made the delivery. This is translated into combining, adjusting and aligning information, knowledge, systems, people and funds for the achievement of making on-time deliveries on the last mile problem. Using the information system theory to bring an enhanced understanding of systems and collect data that would be transformed from data without substance into models and useful forms of computer programs would positively impact understanding and assist on delivering user satisfaction. Therefore, a database is an essential element to the study because it captures the details of users, deliveries, locations. Because there are links between the entities, a relational database was developed. But the database structure and analysis are to be expanded on in the next phase of the study to:</p> <ul style="list-style-type: none"> • Extract information such as travel time, kinds of deliveries by race, region and age. • Determine and understand the expectations of both the requestor and deliverer. • Inform trends from the application based on the different locations. <p>All this data can then be used to create a better solution. The Green Dilivari application confirms the literature on how information systems can be used for greener solutions. The data collected and stored in the database of the application gives evidence that this study's solution also supports a small business owner who may be unable to focus on all three aspects of the triple bottom line from the onset.</p>
<p>Objective 3</p>	<p>The combined use of the agile and waterfall models enabled quick decision making in integrating different technologies and making features of the application function optimally while also creating a process on 'how' to select, integrate and test technologies used in creating the Green Dilivari application. One key feature of the application is its ability to assist users navigate between</p>

<p>Compute an algorithm that uses artificial intelligence to integrate technologies that map and identify the crowd to retailers or businesses.</p>	<p>locations – the pick-up address and the destination address. On dealing with the locations and the process of identifying the shortest route with least traffic, integrating the Google Maps API technology and Microsoft Azure tools assisted in shaping the solution.</p> <p>In a study on the potential impact of crowd shipping, Simoni et al. (2020) explain that crowd shippers are crucial for a sustainable solution for the last-mile distribution problem. The use of tools such as Google Maps use an algorithm to obtain information in order to calculate and decide on the shortest delivery route with the least traffic so as to reduce fuel consumption while ensuring quick deliveries. Google Maps uses a Greedy algorithm that chooses the best choice without considering options that were previously taken. It relies heavily on the Dijkstra’s algorithm for computing directions between two locations. Other algorithms are used within the application such as the search algorithm for logging in where the login credentials are searched within the database. Additionally, a similar approach to dynamic programming is used to store user profiles and details to avoid making database calls that would over-utilize resourcing and have a negative impact on the battery power and space of the mobile application.</p> <p>This study concurs that fewer delivery trucks will be needed while deploying a user who is already moving towards the destination within the same convenient route selected by the application. This would be beneficial to the triple bottom line, providing evidence that coordination is feasible while you have the use of IS in capturing data about the users, deliveries, and locations. Furthermore, the use of both theories, the coordination theory and information system theory, in building a mobile application provides greater opportunities towards balancing the triple bottom line by applying scientific knowledge for the practical purpose of using technology in the last mile distribution problem.</p>
<p>Theoretical findings</p> <ul style="list-style-type: none"> - Digital coordination theory - Triple bottom line 	<p>The combination of the coordination theory and information system theory yields a digital coordination theory. Put differently, technology is the moderator that helps operationalize management theories into practice.</p> <p>The combination of the coordination theory and information systems theory into a mobile application</p>

	provides greater opportunities towards balancing the triple bottom line. Using technology for greener solutions helps preserve the planet while also benefiting people by assisting individuals and business in revenue growth.
Methodological findings Design science	The application of design science for digital artefacts is expensive and requires an interdisciplinary approach. For example, the problem under study is rooted in management study, yet the potential mediating solutions lie in computer science. The evaluation of the artefacts is observed through modelling and simulation.

6.4. RESEARCH AIM ACHIEVEMENTS

This subsection highlights the achievements of the study. The objectives of this study were achieved by creating a mobile application that users could download and have the functionality to request the delivery of a parcel and that functionality to provide that delivery along the route the user is on. The application's key functionality is providing users the ability to make a delivery offer or offer to make a delivery while the location functionality identifies the route between the pick-up and destination addresses. These functionality features were developed and tested. However, due to time constraints, not all validations could be included in this phase of the study. The current state of the application evidence that technology has the ability to bridge the gap and strike a balance between people, profit and the planet.

6.5. SUMMARY FINDINGS AND RECOMMENDATIONS

Seminal or pioneering work done on this study suggests that balancing the triple bottom line is very well possible by providing supporting evidence from literature as explored in previous chapters. Taking into account the necessity for such a research, technology may be used to create a solution that benefits both end users and the environment while also generating a valuable and cost-effective solution. On the contrary, reasons cited on this study suggest that more time should be invested on

such a study. There are several components involved on this study that need careful attention and consideration. One is the tools used; new technologies are released on a daily basis intended to simplify life and business via more efficient databases, operating systems, development languages, IDEs and more. Time is required to explore the expanse of possible tools based on several selection criteria such as cost, efficiency, licensing, customisability, ease of use and many other features which would enable selection.

Secondly, more analysis is required within SDLC for the development languages, design and coding patterns that would cater for easy maintenance and improvements. Input from different experts such as application design, usability and testing experts would also add valuable inputs. The recommendations and suggestions made by users who tested the application give an indication that users have varying views and expectations, hence the need to gather input from different experts to yield a holistic application.

Thirdly, testing, as covered in the previous chapter, is very important in validating the software quality, and should be uncompromised to eliminate vulnerabilities, threats and to ensure that all requirements are covered and functioning as expected. Testing is very broad; it requires time and expertise but given the use of a lab-based approach, not all bases of testing were covered which leaves room for improvement on future work.

6.6. CHAPTER SUMMARY

This chapter revisited the aims of the study as captured at the first phase of the study and discussed whether the aim and the objectives were achieved. This chapter further showcased evidence to support existing literature by supplementing evidence from the testing done to re-affirm that a greener approach is possible to balance the triple bottom line using technology.

REFERENCES

Abrahamson, P., Salo, O. and Ronkainen J. 2002. "Agile software development methods: Review and analysis," *VTT Publications*, 112. Available at: <http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf>.%0Ahttp://www.vtt.fi/inf/pdf/publications/2002/P478.pdf.

Ahi, P. and Searcy, C. (2015) "Assessing sustainability in the supply chain: A triple bottom line approach," *Applied Mathematical Modelling*, 39(10–11), pp. 2882–2896. doi: 10.1016/j.apm.2014.10.055.

Ahmed, A. (2009) "Introduction to software testing management," *Software Testing as a Service*, pp. 1–18. doi: 10.1201/9781420099577.ch1.

Akanmu, S. A. *et al.* (2013) "Functional requirements of mobile application for fishermen," (December).

Al-Saqqa, S., Sawalha, S. and Abdelnabi, H. (2020) "Agile software development: Methodologies and trends," *International Journal of Interactive Mobile Technologies*, 14(11), pp. 246–270. doi: 10.3991/ijim.v14i11.13269.

Albarka Umar, M. (2020) "Comprehensive study of software testing: Categories, levels, techniques, and types software testing view project system analysis view project comprehensive study of software testing: categories, levels, techniques, and types," *International Journal of Advance Research, Ideas and Innovations in Technology*, 5(6), pp. 32–40. doi: 10.36227/techrxiv.12578714.v2.

Ali, U. and Arif, K. S. (2020) "A Comparative Study Mobile Application Testing Tools and their Challenges : A Comparative Study," (February). doi: 10.1109/ICOMET.2019.8673505.

Ameen, S. Y. and Mohammed, D. Y. (2022) "Developing cross-platform library using Flutter," *European Journal of Engineering and Technology Research*, 7(2), pp. 18–21. doi: 10.24018/ejeng.2022.7.2.2740.

Arslan, A. *et al.* (2016) "Crowdsourced delivery -- a pickup and delivery problem with ad-hoc drivers," *SSRN Electronic Journal*, (February). doi: 10.2139/ssrn.2726731.

Awwad, M., Shekhar, A. and Iyer, A. S. (2018) "Sustainable last-mile logistics

operation in the era of e-commerce,” *Proceedings of the International Conference on Industrial Engineering and Operations Management*, 2018(SEP), pp. 584–591.

Bahill, A. T. and Madni, A. M. (2017) *Discovering system requirements*. doi: 10.1007/978-3-319-43712-5.

Binetti, M. *et al.* (2019) “A sustainable crowdsourced delivery system to foster free-floating bike-sharing,” *Sustainability (Switzerland)*, 11(10). doi: 10.3390/su11102772.

Brotcorne, L. *et al.* (2019) “A managerial analysis of urban parcel delivery: A lean business approach,” *Sustainability (Switzerland)*, pp. 1–23. doi: 10.3390/su11123439.

Brown, J. R. and Guiffrida, A. L. (2014) “Carbon emissions comparison of last mile delivery versus customer pickup,” *International Journal of Logistics Research and Applications*, 17(6), pp. 503–521. doi: 10.1080/13675567.2014.907397.

Buldeo Rai, H. (2019) “Environmental sustainability of the last mile in omnichannel retail,” p. 250.

Buys, R. T. (2009) “M SC PL O E – C EO M SC PL O E – C EO,” I.

Chankov, S. M., Becker, T. and Windt, K. (2014) “Towards definition of synchronization in logistics systems,” *Procedia CIRP*, 17(April 2016), pp. 594–599. doi: 10.1016/j.procir.2014.02.034.

Chaves, R. *et al.* (2019) “Crowdsourcing as a tool for urban emergency management: Lessons from the literature and typology,” *Sensors (Switzerland)*, 19(23). doi: 10.3390/s19235235.

Chen, W., Mes, M. and Schutten, M. (2018) “Multi-hop driver-parcel matching problem with time windows,” *Flexible Services and Manufacturing Journal*. Springer US, pp. 517–553. doi: 10.1007/s10696-016-9273-3.

Costa, J., Diehl, J. C. and Snelders, D. (2019) “A framework for a systems design approach to complex societal problems,” *Design Science*. doi: 10.1017/dsj.2018.16.

Dabbagh, M., Lee, S. P. and Parizi, R. M. (2016) “Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches,” *Soft Computing*, 20(11), pp. 4497–4520. doi: 10.1007/s00500-015-

1760-z.

Devari, A. (2016) "Crowdsourced last mile delivery using social networks," p. 48.

Dubey, Jain, M. (2015) "International journal of engineering sciences & research technology comparative study: waterfall v/s agile model," 4(3), pp. 70–75.

Ducarme, D. (2019) "Sustainable solutions for ' last mile ' deliveries in the parcel industry."

Engineering, F. O. F. (2011) "Department of Industrial Management Coordination Mechanisms," (June).

Fojtik, R. (2020) "Swift a new programming language for development and education," *Advances in Intelligent Systems and Computing*, 1114 AISC(January), pp. 284–295. doi: 10.1007/978-3-030-37737-3_26.

Frehe, V., Mehmman, J. and Teuteberg, F. (2017) "Understanding and assessing crowd logistics business models – using everyday people for last mile delivery," *Journal of Business and Industrial Marketing*, 32(1), pp. 75–97. doi: 10.1108/JBIM-10-2015-0182.

Gate, W. Van (2017) "The waterfall model and the agile methodologies: A comparison by project characteristics," *Researchgate.Net*, (March), pp. 10–13. doi: 10.13140/RG.2.2.10021.50403.

Gatta, V. *et al.* (2018) "Public transport-based crowdshipping for sustainable city logistics: Assessing economic and environmental impacts," *Sustainability (Switzerland)*, 11(1), pp. 1–14. doi: 10.3390/su11010145.

Gay, A. C. and Norrman, E. (2016) "Improving the coordination of the supply chain - a case study of the battery charger manufacturer Micropower and its subsidiary Ecotec Master Thesis."

Germany, I. (2012) "Back to the future Deliver by bike : Growing in many countries."

Gilliland, S. (2017) "Develop a framework of human factors to assist organisations in managing EA acceptance," pp. 83–122.

Gorrod, M. (2004) "System design and implementation," *Risk Management Systems*, 1, pp. 153–184. doi: 10.1057/9780230510296_6.

Halawi, L. and McCarthy, R. (2006) "Which theory applies: An analysis of information systems research," *Issues in Information Systems*, 7(2), pp. 252–256.

Hussain, H. *et al.* (2022) "Comparative study of android Native and Flutter app development," (June), pp. 98–102.

Hwang, S. *et al.* (2020) "Designs and methods for implementation research: Advancing the mission of the CTSA program," *Journal of Clinical and Translational Science*, 4(3), pp. 159–167. doi: 10.1017/cts.2020.16.

Idan Arb, G. and Al-Majdi, K. (2020) "A freights status management system based on Dart and Flutter programming language," *Journal of Physics: Conference Series*, 1530(1). doi: 10.1088/1742-6596/1530/1/012020.

Imcs, A. Š. *et al.* (2020) "BalticLSC software user requirements specification priority 1 : Innovation BalticLSC Software User Requirements Specification," pp. 0–37.

Jain, V. and Sharma, A. (2013) "The consumer's preferred operating system : Android or iOS," 3(4), pp. 29–40.

Jha, A. N. and Chourasia, R. (2011) "Location based services in Android with Google maps integration," *International Journal of Advances in Engineering & Technology*, 1(1), pp. 14–20.

K.Flora, H., V. Chande, S. and Wang, X. (2014) "Adopting an agile approach for the development of mobile applications," *International Journal of Computer Applications*, 94(17), pp. 43–50. doi: 10.5120/16454-6199.

KamolsonSu, S. (2007) "Fundamentals of quantitative research Suphat Sukamolson, Ph.D. Language Institute Chulalongkorn University," *Language Institute*, p. 20. doi: 9781848608641.

Knott, D. (2015) *About This eBook*.

Koch, F. *et al.* (2013) "Citizen in sensor networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7685(June 2015), pp. 57–66. doi: 10.1007/978-3-642-36074-9.

Latake, P. T. and Pawar, P. (1201) "The greenhouse effect and its impacts on

environment,” *International Journal of Innovative Research and Creative Technology* www.ijirct.org, 1(3).

Lawanna, A. (2012) “The theory of software testing,” *Review Article*, 16(1), pp. 35–40. Available at: <https://www.researchgate.net/publication/236031163>.

Learning, C. and Reserved, A. R. (2010) *Database Systems*, 9th Edition, *Management*.

Lewis, W. (2008) “Software testing techniques,” *Software Testing and Continuous Quality Improvement*, 3rd Edition, pp. 557–627. doi: 10.1201/9781439834367.axg.

Lindholm, R. and Lindholm, R. (2012) “Defining requirements in a process and system development project.”

Maes, J. and Vanelslander, T. (2012) “The use of bicycle messengers in the logistics chain, concepts further revised,” *Procedia - Social and Behavioral Sciences*, 39(December), pp. 409–423. doi: 10.1016/j.sbspro.2012.03.118.

Managers, I. T. P., Guideline, T. and Sdlc, T. (2008) “System development lifecycle,” pp. 1–33.

Map, C. (2000) “Application architecture and modeling.”

Maryland (2009) “Phase 5 : Design phase,” 2(December 2009), pp. 1–20.

Maxwell, J. (1992) “Understanding and validity in qualitative research,” *Harvard Educational Review*, 62(3), pp. 279–301. doi: 10.17763/haer.62.3.8323320856251826.

Mccormick, M. (2012) “Waterfall vs. agile methodology.”

Mithun Satheesh, Bruno Joseph D’mello, J. K. (2015) “Web development with MongoDB and NodeJS.” Available at: www.packpub.com.

Mladenow, A., Bauer, C. and Strauss, C. (2015) “Crowdsourcing in logistics: Concepts and applications using the social crowd,” *17th International Conference on Information Integration and Web-Based Applications and Services, iiWAS 2015 - Proceedings*, (August 2016). doi: 10.1145/2837185.2837242.

Mobility, E. and Platform, M. (2013) “Data integrity in mobility,” pp. 1–8.

Molinary, S. (2018) "Google , Android and Location Tracking 1) How Google Collects Location Data," (September), pp. 1–14.

"MongoDB" (no date).

Mule, S. S. and Waykar, Y. (2015) "Role of Use Case Diagram in S/W Development," *International Journal of Management and Economics*. Available at: https://www.researchgate.net/publication/322991847_role_of_use_case_diagram_in_software_development#fullTextFileContent.

Net, A. S. P. (2020) "Online map application development using google maps API , SQL database , Online Map Application Development Using Google Maps API , SQL Database , and ASP . NET," (January 2013).

Odongo, B. (2017) "How crowdsourcing is transforming the face of last mile delivery 'Crowd logistics,'" pp. 1–73.

Officer, I. (2016) "for Business Analyst Appendix C a Template of User Requirements Document," (December).

Okelo-odongo, P. W. (2014) "Implementing data integrity and confidentiality in mobile phone based cyber foraging system. School of Computing and Informatics . October 2014," (October).

Olsson, J., Hellström, D. and Pålsson, H. (2019) "Framework of last mile logistics research: A systematic review of the literature," *Sustainability (Switzerland)*, 11(24). doi: 10.3390/su11247131.

Olsson, M. (2020) "A comparison of performance and looks between Flutter and Native applications: When to prefer Flutter over native in mobile application development," pp. 1–51. Available at: <http://urn.kb.se/resolve?urn=urn:nbn:se:bth-19712>.

Abrahamson, P., Outi Salo and Jussi Ronkainen (2002) "Agile software development methods: Review and analysis," *VTT Publications*, p. 112. Available at: <http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf>.%0Ahttp://www.vtt.fi/inf/pdf/publications/2002/P478.pdf.

Paper, W. (2018) "Delivering the goods e-comm logistics transformation," (October).

- Peng, R. and Xu, A. (2016) "Crowdsourced logistics – its development and potential."
- Petzer, D. (2016) "Quantitative research module," Doctoral Programme 2016, pp. 1–16.
- Practices, B. and Security, O. (2014) "Mobile Application Security," (July).
- Prasad, B. and Strand, N. (1993) "A flow-chart-based methodology for process improvement," (October 1993). doi: 10.13140/RG.2.1.2559.4327.
- Priya, D. (no date) "Guidelines for planning and development of software for buildings and building systems," *Applied Economics*.
- "REST API Developer Guide" (2022).
- Rodríguez, C. *et al.* (2016) "REST APIs: A large-scale analysis of compliance with principles and best practices," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9671(February 2018), pp. 21–39. doi: 10.1007/978-3-319-38791-8_2.
- Rougès, J.-F. and Montreuil, B. (2014) "Crowdsourcing delivery : New interconnected business models to reinvent delivery," *1st International Physical Internet Conference*, (1), pp. 1–19.
- Ruparelia, N. B. (2010) "Software development lifecycle models," *ACM SIGSOFT Software Engineering Notes*, 35(3), pp. 8–13. doi: 10.1145/1764810.1764814.
- Sahatqija, K. *et al.* (2018) "Comparison between relational and NOSQL databases," (May). doi: 10.23919/MIPRO.2018.8400041.
- Sampaio, A. *et al.* (2017) "Crowd-based City Logistics," pp. 1–14.
- Samra, H., Li, A. and Soh, B. (2020) "G3dms: Design and implementation of a data management system for the diagnosis of genetic disorders," *Healthcare (Switzerland)*, 8(3). doi: 10.3390/healthcare8030196.
- Scherr, Y. O. *et al.* (2018) "Service network design for same day delivery with Mixed autonomous fleets," *Transportation Research Procedia*, 30, pp. 23–32. doi: 10.1016/j.trpro.2018.09.004.

Senarath, U. S. (2021) "Waterfall methodology , prototyping and agile development By," (June). doi: 10.13140/RG.2.2.17918.72001.

Simatupang, T. M., Sandroto, I. V. and Lubis, S. B. H. (2004) "Supply chain coordination in a fashion firm," *Supply Chain Management*, 9(3), pp. 256–268. doi: 10.1108/13598540410544953.

Simatupang, T. M., Wright, A. C. and Sridharan, R. (2002) "The knowledge of coordination for supply chain integration," *Business Process Management Journal*, 8(3), pp. 289–308. doi: 10.1108/14637150210428989.

Slack, J. M. (2011) "System testing of desktop and web applications," 9 August, pp. 68–82.

Subhan, Z. and Bhatti, A. T. (2015) "Requirements Analysis and design in the context," 3(Vi), pp. 812–820.

"System Development and Life- Cycle Management (SDLCM) Methodology Handbook, Version 2.3 Foreword Nuclear Regulatory Commission (NRC) Management Directive 2.5, " Application Systems Life- Cycle Management, " establishes the policies for applications system" (2002), (July).

Thorogood, G. & (2009) "Requirement gathering methods," pp. 127–129.

Todorovic, V. *et al.* (2018) "Solutions for more sustainable distribution in the short food supply chains," *Sustainability (Switzerland)*, 10(10). doi: 10.3390/su10103481.

Vijiyalakshmi, P. (2016) "Software engineering issues of mobile application development," 6(5), pp. 5728–5732. doi: 10.4010/2016.1395.

Wasilewski, K. (2021) "A Comparison of Java , Flutter and Kotlin / Native Technologies."

Wicaksono, S. (no date) "Exploring the Market Potential of Bicycle Crowdfunding: a Bi-Level Acceptance Perspective."

"World Urbanization Prospects : The 2018 Revision" (2018).

APPENDICES

APPENDIX 1: TURNITIN REPORT

Document Viewer

Turnitin Originality Report

Processed on: 22-Sep-2023 19:16 SAST
ID: 2173825132
Word Count: 25115
Submitted: 1

TRIPLE_BOTTOM_LINE_IN_LAST_MILE_DISTRIBUTION_... By Mangaliso Phillip Sipamla

Similarity by Source	
Similarity Index	
16%	Internet Sources: 14%
	Publications: 6%
	Student Papers: 10%

exclude quoted | exclude bibliography | exclude small matches | mode: quickview (classic) report | print | download

2% match (Internet from 17-Dec-2021)
<http://www.tutorialspoint.com>

1% match (Internet from 02-May-2021)
<http://uir.unisa.ac.za>

1% match (Internet from 16-Jan-2023)
<https://co-foundry.co.za/4-types-of-mobile-and-web-apps-with-examples/>

<1% match (Internet from 25-Jun-2022)
https://uir.unisa.ac.za/bitstream/handle/10500/28897/dissertation_cohen_t.pdf?isAllowed=y&sequence=1

<1% match (Internet from 03-Sep-2022)
https://uir.unisa.ac.za/bitstream/handle/10500/28587/dissertation_mokoena_r.pdf?isAllowed=y&sequence=1

<1% match (Internet from 27-Jul-2023)
https://uir.unisa.ac.za/bitstream/handle/10500/30003/dissertation_joseph_ob.pdf?isAllowed=y&sequence=1

<1% match (Internet from 16-Mar-2023)
https://uir.unisa.ac.za/bitstream/handle/10500/29739/dissertation_tshehla_mf.pdf?isAllowed=y&sequence=1

<1% match (Internet from 28-Apr-2023)
https://uir.unisa.ac.za/bitstream/handle/10500/29884/dissertation_munyai_mp.pdf?isAllowed=y&sequence=1

<1% match (Internet from 10-Apr-2022)
https://uir.unisa.ac.za/bitstream/handle/10500/28584/dissertation_viviers_gm.pdf?isAllowed=y&sequence=1

<1% match (Internet from 11-Dec-2019)
<http://uir.unisa.ac.za>

APPENDIX 2: ETHICS CERTIFICATE



UNISA COLLEGE OF SCIENCE, ENGINEERING AND TECHNOLOGY'S (CSET) ETHICS REVIEW COMMITTEE

30 November 2020

Dear Prof. Mkansi

ERC Reference #: 2020/CSET/SOC/056
Name: Marcia Mkansi
Staff #: 90215028

**Decision: Ethics Approval from
30 November 2020 to 29 November 2023
(No humans involved)**

Researcher: Prof. Marcia Mkansi
Department of Operations Management, Mkansm@unisa.ac.za,
076 833 3274

Working title of research:

**Green crowdsourcing mobile application (Dilivari): Human-centred
decarbonization mobile application (Green)**

Qualification: Non-degree

Thank you for the application for research ethics clearance by the Unisa College of Science, Engineering and Technology's (CSET) Ethics Review Committee for the above mentioned research. Ethics approval is granted for 3 years.

*The **negligible risk application** was expedited by the College of Science, Engineering and Technology's (CSET) Ethics Review Committee on 30 November 2020 in compliance with the Unisa Policy on Research Ethics and the Standard Operating Procedure on Research Ethics Risk Assessment. The decision will be tabled at the next Committee meeting for ratification.*

The proposed research may now commence with the provisions that:

1. The researcher will ensure that the research project adheres to the relevant guidelines set out in the Unisa COVID-19 position statement on research ethics attached.
2. The researcher(s) will ensure that the research project adheres to the values and principles expressed in the UNISA Policy on Research Ethics.



University of South Africa
Preller Street, Muckleneuk Ridge, City of Tshwane
PO Box 392 UNISA 0003 South Africa
Telephone: +27 12 429 3111 Facsimile: +27 12 429 4150
www.unisa.ac.za

3. Any adverse circumstance arising in the undertaking of the research project that is relevant to the ethicality of the study should be communicated in writing to the College of Science, Engineering and Technology's (CSET) Ethics Review Committee.
4. The researcher(s) will conduct the study according to the methods and procedures set out in the approved application.
5. Any changes that can affect the study-related risks for the research participants, particularly in terms of assurances made with regards to the protection of participants' privacy and the confidentiality of the data, should be reported to the Committee in writing, accompanied by a progress report.
6. The researcher will ensure that the research project adheres to any applicable national legislation, professional codes of conduct, institutional guidelines and scientific standards relevant to the specific field of study. Adherence to the following South African legislation is important, if applicable: Protection of Personal Information Act, no 4 of 2013; Children's act no 38 of 2005 and the National Health Act, no 61 of 2003.
7. Only de-identified research data may be used for secondary research purposes in future on condition that the research objectives are similar to those of the original research. Secondary use of identifiable human research data require additional ethics clearance.
8. No field work activities may continue after the expiry date 29 November 2023. Submission of a completed research ethics progress report will constitute an application for renewal of Ethics Research Committee approval.

Note

The reference number 2020/CSET/SOC/056 should be clearly indicated on all forms of communication with the intended research participants, as well as with the Committee.

Yours sincerely,



Dr C Pilkington
Chair of School of Computing Ethics Review Subcommittee
College of Science, Engineering and Technology (CSET)
E-mail: pilkicl@unisa.ac.za
Tel: (011) 471-2130



Prof. E Mnkandla
Director: School of Computing
College of Science Engineering and
Technology (CSET)
E-mail: mnkane@unisa.ac.za
Tel: (011) 670 9104



Prof. B Mamba
Executive Dean
College of Science Engineering and
Technology (CSET)
E-mail: mambabb@unisa.ac.za
Tel: (011) 670 9230



URERC-25.04.17 - Decision template (V2) - Approve

University of South Africa
Preller Street, Muckleneuk Ridge, City of Tshwane
PO Box 392 UNISA 0003 South Africa
Telephone: +27 12 429 3111 Facsimile: +27 12 429 4150
www.unisa.ac.za

APPENDIX 3: EDITING CERTIFICATE

Ke.Nna
Publishing Services



This certificate serves to confirm that the thesis, *BALANCING THE TRIPLE BOTTOM LINE IN LAST MILE DISTRIBUTION: A DIGITAL MOBILE APPROACH*, was edited by Ms SehloDIMELA. She is contracted by the University of South Africa's College of Economic and Management Sciences to provide academic editing services. She holds a Masters in TESOL and is a certified Project Management Professional.

The services provided include:

1. Ensuring accuracy in grammar and punctuation to improve readability and clarity;
2. Consistency and structural enhancements to aid in creating a cohesive article that has a logical flow and appropriate tone; and
3. Formatting in alignment with the stipulated style guide

For any enquiries relating to the above, see below contacts:



CT SEHLODIMELA, MA(TESOL), PMP
Managing Director: Ke.Nna Publishing Services

+2782 075 5078



Tshegofatso.s@outlook.com



18 Lakeview
106 Haymeadow Crescent
Faerie Glen
0081

