REPUBLIC OF RWANDA

UNIVERSITY OF LAY ADVENTISTS OF KIGALI

(UNILAK)

KIGALI CAMPUS

P.O.BOX 6392 KIGALI, Phone: +250786024097

Email: info@unilak.ac.rw; Website: www.unilak.ac.rw

FACULTY OF COMPUTING AND INFORMATION SCIENCES

SOFTWARE ENGINEERING DEPARTMENT

EASING DRIVERS' STRUGGLES DURING CITY PARKING: AN IOT-BASED MOBILE APPLICATION APPROACH

CASE STUDY: KIGALI CITY

A final project (dissertation) submitted to the faculty of Computing and Information Sciences in partial fulfillment of academic requirements for the award of a Bachelor's Degree of Science in **Software Engineering.**

Submitted by:

JULIUS ARIES KANNEH, JR

Student ID: 17569/2021

Under the guidance of

Mr. Gilbert NSHIMYUMUREMYI

Kigali, February, 2024

DEDICATION

As a first-generation graduate, I extend my heartfelt gratitude to God Almighty. I dedicate this work to my beloved family and siblings who have been unwavering pillars of support throughout my academic journey.

Additionally, I extend this dedication to the aspiring engineers in my country. May this work serve as an inspiration, showcasing the immense potential we hold to contribute positively to the world through technology. It is a testament to our ability to analyze problems, engineer innovative solutions, and bring about meaningful change.

To all those who dare to revolutionize and make a significant impact with the gifts bestowed upon them by God, this work is dedicated to you. May it encourage and motivate the young generation to harness their abilities for the betterment of our world.

DECLARATION

I, Julius Aries Kanneh, Jr hereby declare that the project report entitled **EASING DRIVERS' STRUGGLES DURING CITY PARKING: AN IOT & MACHINE LEARNING BASED APPROACH** submitted in partial fulfillment of the requirement for the award of the degree of Bachelor of Science in Software Engineering in Kigali Branch is a record of bonafide project work carried out by myself under the guidance/supervision of Mr Gilbert NSHIMYUMUREMYI. I further declare that the work reported in this project has not been submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Signature: _____

Julius Aries Kanneh, Jr

Kigali:

Date: ____ / _____ / 2024

UNIVERSITY OF LAY ADVENTISTS OF KIGALI (UNILAK)

FACULTY OF COMPUTING AND INFORMATION SCIENCES

DEPARTMENT OF SOFTWARE ENGINEERING

B.P 6392 KIGALI, Tel+: + 25055107311/ 55104697

E-mail: Info@unilak.ac.rw ,Website: www.unilak.ac.rw

BONAFIDE CERTIFICATE

This is to certify that the project report entitled "**EASING DRIVERS' STRUGGLES DURING CITY PARKING: AN IOT & MACHINE LEARNING BASED APPROACH**" submitted by Julius Aries Kanneh, Jr (17569/2021) to the University of Lay Adventists of Kigali in partial fulfillment to the requirement of the award of the bachelor's degree in Software Engineering is a record of bonafide work carried out by him under my guidance.

SUPERVISOR

Mr. Gilbert NSHIMYUMUREMYI

Date: …… /…… /2024

DEAN OF FACULT

DR. ENAM NYESHEJA MUHIRE PhD

Date: …… /…… /2024

Accredited by the Ministerial Order N° 002/09 0f 09/04/2009 granting the Definitive Operating License.

ACKNOWLEDGMENT

I express my deepest gratitude to God Almighty, the source of my strength and unwavering support throughout this journey.

My sincere thanks go to my supervisor, Mr. Gilbert, whose invaluable guidance, insightful recommendations, and unwavering support played a pivotal role in the success of this project. His belief in the project's potential from the outset was truly inspiring, and I am grateful for the privilege of having his mentorship.

I extend my appreciation to my friends and mastermind groups for their critical reviews and valuable feedback, which significantly contributed to the improvement of this work.
I am also grateful to Mr. Dadley Toe, my esteemed mentor, for his guidance and support throughout my academic journey.

Also, I am grateful to Mr NZIRINGIRIMANG Gervais whose carpentry expertise was used with my supervision to build the physical prototype for this project.

Finally, I want to express my sincere appreciation to everyone who believed in my idea, offered support in any form, and provided constructive criticism. Your contributions, both large and small, made a significant difference.

Julius Aries Kanneh, Jr

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

API: Application Programming Interface

GWP: Global Warming Potential

IDE: Integrated Development Environment

IoT: Internet of Things

IR: Infrared

KVCS: Kigali Veterans Cooperative Society

ML: Machine Learning

MQTT: Message Query Telemetry Transport

NoSQL: Not Only SQL / Non-SQL

OCR: Optical Character Recognition

OpenCV: Open Source Computer Vision Library

RSVP: Reservation

SQL: Structured Query Language

ABSTRACT

The manual approach to parking in Kigali city significantly impacts drivers, wasting time and contributing to environmental concerns due to huge emission of greenhouse gases in search of an available parking lot. This research project aims to significantly reduce the time spent on parking procedures for drivers while concurrently mitigating greenhouse gas emissions associated with parking-related activities, thus contributing to Kigali's commitment to having a green city. The scope of the system is confined to commercial indoor parking lots, excluding street or outdoor parking areas.

Employing the Object Oriented Analysis Design Methodology, the project addresses the problem by developing a user-centric approach, focusing on enhancing the overall parking experience for drivers. Key features include a robust mobile application with intuitive user interfaces, real-time spot occupancy, accurate gate automation, and a digital payment processing approach. To achieve these goals, the system employs state of the art technologies such as Internet of Things (IoT), Flutter, Firebase, and Machine Learning.

The research findings and the developed system have the potential to offer practical solutions for urban public commercial parking lots, and drivers, fostering a more seamless and eco-friendly urban parking experience.

Key words: *Eco-friendly urban parking, Commercial indoor parking, Digital payment processing, Real-time spot occupancy, Gate automation, Mobile Application, Internet of Things.*

CHAPTER ONE: GENERAL INTRODUCTION

1.1 Introduction

Time is one of the priceless assets to humanity yet, we unconsciously squander it on activities like looking for available parking lots within a city. Too often, drivers' time and petroleum are consumed driving around a city in search of parking lots. I believe the time spent on searching for parking lots can be delegated to activities of more significance which will have a positive impact on the individual's life and well-being as the proper use of time is hugely associated with our well-being and life (Gichuki, 2023). This is why this research project, "Easing drivers' struggles during city parking: An IoT-based mobile application approach", is focused on optimizing the time spent by drivers to locate and park their vehicles within metropolitan areas, taking Kigali as my case study. This project proposes an all-in-one parking solution for Kigali city to streamline the entire process of finding available public parking, making payments, and automating the gate opening and closing using Flutter for mobile application development, Internet of Things (IoT), and Machine Learning.

1.2 Background

As the world advances, technologies evolve, and everything is becoming fast and rapid, one of the most sought-after factors of life is our precious time. Time is one of the most mysterious forces in the universe (The Scientific World, 2020). It is the most valuable source because we, as humans, can't take it back. Time is free, but it's priceless. You can't own it, but you can use it. You can't keep it, but you can spend it. Once you've lost it you can never get it back. (Gichuki, 2023). Due to the realization of the significance of time to humanity, most if not all of technological advancements have been geared toward optimizing the time it takes to accomplish a task and to make the process easier.

Dating back to the Industrial Revolution which had a transformative economic, social, and cultural impact, and played an integral role in laying the foundations for modern society (History.com Editors, 2009), was highly influenced by humans' desire to produce more in less time with less effort.

The act of drivers driving around the city in search of available parking lots contributes to making the earth warmer and "thickening the earth's atmospheric blanket" due to the emissions of greenhouse gasses by the petroleum consumption vehicles. The combustion of

fossil fuels (coal, natural gas, and oil) for energy and transportation from motor vehicle exhausts also produces greenhouse gasses that contribute to climate change (United States Environmental Protection Agency, 2023). The main greenhouse gas produced by vehicles is carbon dioxide (CO2), but they also produce nitrous oxide and methane (Green Vehicle Guide, n.d.). The constant emission of carbon dioxide (CO2) on road travel accounts for three-quarters of transport emissions. Most of this comes from passenger vehicles – cars and buses – which contribute 45.1%. The other 29.4% comes from trucks carrying freights that affect the environment. Transport accounts for around one-fifth of global CO2 emissions [24%] if we only consider CO2 emissions from energy (Ritchie, 2020). Therefore, reducing the amount of time drivers spend driving around the city to find a parking lot will reduce the emission of carbon monoxide, which is harmful to humans, into the environment.

1.3 Problem Statement

The manual approach to parking in Kigali city poses significant challenges for drivers, particularly personal vehicle owners who make up 67.5% of respondents from a survey conducted. The process of locating available parking lots not only contradicts the primary goal of vehicle ownership for efficient transportation but also results in considerable time wastage, adversely affecting drivers' productivity and mental well-being. This time-consuming search contributes to environmental issues, including increased greenhouse gas emissions such as carbon dioxide, as drivers circle the city in search of parking. The study underscores the critical need to address the inefficiencies in the existing parking system, prioritizing the time wastage issue while acknowledging the essential role of secure parking environments. Despite recognizing other challenges, the research focuses on the substantial time lost by drivers, aiming to provide insights and solutions to enhance the overall efficiency and sustainability of urban parking.

1.4 Motivation

Drivers meandering the city in search of parking lots affect two major components of human existence: time and the earth’s potential to inhabit life. Drivers wasting time looking for parking can lead to increased traffic congestion, delays, and frustration. This wasted time can have a ripple effect on other aspects of life, potentially affecting work productivity, personal schedules, and overall well-being. On the other hand, the prolonged idling of vehicles

searching for parking contributes to greenhouse gas emissions, air pollution, and increased fuel consumption. These factors negatively impact the environment, diminishing its capacity to support life.

The desire and passion to solve problems and make the world a better place than I met has fueled my decision to study Software Engineering. While researching the problem, realizing the huge impact a single vehicle emission of greenhouse gasses has on the environment by meandering for a few minutes in search of a parking lot raises an alert for call to action. It dawns on me that unconsciously, humanity has been destroying Earth while hoping to inhabit it for a while. It is like setting a house on fire while you are still living there and have nowhere else to go.

Furthermore, understanding the significance of how important time is to humanity, I couldn't help but to invest my time to save the time of others. Imaging the impact of improved lives and the reduction of the emission of greenhouse gasses in Kigali by overcoming the challenges involved in the manual system of parking in Kigali city is my inspiration for this research.

1.5 Objectives of Study

This project's objectives are meticulously structured into two distinct categories: general objectives and specific objectives. The general objective describes the aim of this research project, providing a broad framework for its direction and success. On the other hand, Specific objectives, delve into the granular details of the project, outlining the tangible deliverables and milestones that must be achieved to attain the overarching goals.

1.5.1 General Objectives

This project aims to develop a robust all-in-one IoT-based system to help drivers within the city of Kigali easily locate and book a slot within a parking lot to reduce the time spent in search of a parking lot and minimize the emission of greenhouse gasses within the atmosphere.

1.5.2 Specific Objectives

To achieve the overall objective or aim of this project mentioned above, the following are the specific objectives:

1. Develop a parking spot occupancy IoT system to determine whether or not a specific spot within a parking lot is occupied.

2. Develop a robust user-friendly mobile application to be used by drivers to locate and book parking lots while helping monitor their booking history.

3. Provide a more convenient and easy means to make payments using the mobile application to avoid time wastage while making payments physically.

4. Develop an automated gate at the parking lot using IoT and Machine Learning to detect and recognize vehicles by their plate numbers to reduce the time spent on having someone manually check and open the gate.

1.6 Challenges

To achieve the mentioned specific objectives efficiently, different emerging technologies like IoT, Flutter, Firebase, and Machine Learning are being leveraged. Due to the nature of these technologies being relatively new, it poses a lot of challenges in implementing the solution due to limited available resources on these technologies and the time constraints of delivering the system. Though these challenges exist, the motivation to bring to life and achieve the aim of this project surpasses the underlying challenges.

1.7 Methodology and Techniques

In this research project, a specific methodology, and a few techniques will be used as follows.

1.7.1 Methodology

The Object-Oriented Analysis and Design used as the methodology for this project due to the hetero nature of the project. Following this methodology brings modularity, reusability, etc while allowing a real-world mimicking of the and scenario into the system design and implementation.

In terms of implementation of the system, the Agile methodology will be used because of its iterative nature which is best suited for this project being developed to address the underlying problem.

1.7.2 Techniques

To achieve the above-mentioned specific objectives and answer the research questions of this research, three techniques of data collection were used: observation, interview, and survey.

- **Observation:** A thorough observation was done at various packing within the city of Kigali by the researcher.
- **Interview:** An interview was conducted with few drivers within the city of Kigali which helps give a realistic understanding of the problem being addressed.
- **Survey:** An anonymous survey was conducted by the researcher where drivers and vehicle owners within the city of Kigali were the respondents.

1.8 Scope of the Research

This research project aims to streamline the parking experience for drivers at commercial indoor parking lots within Kigali, minimizing time spent on parking procedures and mitigating greenhouse gas emissions. The scope is limited to commercial indoor parking lots, excluding street or outdoor parking areas.

1.9 Statement of Assumptions

For the optimal realization of the proposed solution in attaining the research objectives, certain assumptions are integral. Clarifying these conditions is imperative for a comprehensive understanding of the methodology employed in this research endeavor.

1. It is assumed that the driver's mobile phone or tablet on which the mobile application is installed is connected to the internet.
2. It is assumed that the vicinity of the parking lot has an active wifi connection that is accessible to the spot occupancy and automated gate IoT systems.
3. It is assumed that the vicinity of the parking lot has electricity to power on-the-spot occupancy and automated gate IoT systems.
4. It is assumed that the parking lot is a commercial parking lot.

5. It is assumed that the parking lot is for the general public.
6. It is assumed that the parking lot is an indoor parking lot.
7. It is assumed that the parking lot has both entry and exit gates.

1.10 Expected Results

On completion of this project, the following are the expected results:

1. **Improves Drivers' Livelihood:** Drivers using the solution are expected to be much happier, peaceful, and free from stress due to the efficient optimization of their time while parking in the city of Kigali.

2. **Boost Drivers' Productivity:** Now that drivers using this solution will spend less time to locate and park their vehicles within the city of Kigali, it is expected that their productivity and performance in other areas of significance will increase since they can now reallocate the wasted time on parking to something of significance.

3. **Reduction of Greenhouse Gasses Emission:** It is expected that there will be a significant reduction in the emission of greenhouse gasses within the atmosphere due to the less time drivers will spend meandering the city in search of an available parking lot. This reduction will reduce the negative effect of Global Warming on Earth and climate change.

4. **Reduces Driver's Petroleum Consumption:** Driving around the city in search of parking lots increases the consumption of the vehicle's petroleum which therefore increases the expense of the Driver. This solution is expected to reduce the drivers' petroleum consumption thus reducing the drivers' expense on petroleum for their vehicles.

5. **Streamline Payment:** The solution will ease the payment for parking lot service since the mobile application will have an internal electronic payment method that is almost automated.

1.11 Organization of the Report

This report is organized into six chapters, from one to six.

Chapter one is the general introduction of the report where the research topic is introduced, the background of the study is properly layout, the problem statement, motivation of the study, objectives, challenges, methodology and technique, the scope of the project, the statements of assumption, expected results, and organization of the study are presented to give readers a solid understanding of the research project.

Following Chapter One is Chapter Two which focuses on Literature Review which presents an objective, critical summary of published literature and developed solutions relevant to this topic under consideration for research.

Chapter three, the Analysis of Existing Systems, delves into currently existing solutions to the underlying problem this research project is addressing. This section outlines the challenges stemming from both the manual system and currently implemented systems, ultimately putting forth proposed solutions for the identified problems.

Chapter four is specifically focused on the development of the proposed system. This section delves into the architecture of the proposed system, design methodologies, models, and tools, and provides an in-depth exploration of the Software Requirement Specifications (SRS).

In chapter five, Implementation of the Proposed System, the tools and technologies used to implement the proposed solutions are presented. The most important user interfaces of the systems are also presented here.

Finally, chapter six concludes the report about the objectives of the research highlighting the solutions to the problems revealed in chapters one and two. The chapter ends with recommendations for further improvement on the proposed solution.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

Over the past few years, a lot of research has been conducted to find a more convenient approach to save drivers time while parking within big/populated cities and to reduce the carbon emission of vehicles. Innovative car park systems are not a recent transportation industry novelty due to the high increase of personal vehicle owners in those cities. Though a lot of literature has been published on the topic, which has led to the deployment of some of those solutions, very few cities in Africa have adopted a robust smart parking system. Smart parking systems have been deployed at a few parking facilities in Kigali, however, these systems still have problems of additional fuel consumption by drivers, increased carbon emission, and a significant waste of drivers' time in search of available parking lots. This chapter of my research delves into a thorough review of some of the significant research on this topic to conclude the significance of my research. More to that, a detailed revision has been done on the existing systems that are being deployed in Kigali.

2.2 Definition of Key Terms

2.2.1 IoT

The Internet of Things (IoT) is the concept of internetworking of physical devices, connected devices, smart devices, and other devices embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data (Sciforce, 2018).

2.2.2 Machine Learning

Machine Learning is an AI technique that teaches computers to learn from experience. Machine learning algorithms use computational methods to “learn” information directly from data without relying on a predetermined equation as a model. (Math Works, n.d.)

2.2.3 Raspberry Pi

The Raspberry Pi is a series of tiny credit card-size computers developed for computer science education purposes. It can function as an embedded system development board and at the same time functioning as a full-fletch computer.

2.2.4 Ultrasonic Sensor

Ultrasonic sensors are electronic devices with two heads, trigger and echo, that calculate the target's distance by emission of ultrasonic sound waves and convert those waves into electrical signals which are calculated to determine the distance of the target from the sensor.

2.2.5 ESP32 Module

ESP32 is a series of low-cost, low-power System-on-a-chip (SoC) microcontrollers with integrated Wi-Fi and dual-mode Bluetooth (Wikipedia, n.d.).

2.2.6 OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products (OpenCV, n.d.).

2.2.7 Computer Vision

Computer vision is a field of artificial intelligence that enables computer systems to capture digital images, video, and other visual input and then derive meaning from that data, so it can make recommendations based on the captured information (Hilson, n.d.).

2.2.8 Real-time Database

A Real-time Database refers to a database system that uses streaming technologies to handle workloads whose state is constantly changing (Wikipedia, n.d.). Real-time databases are optimized for low-latency query performance, high-throughput data ingestion, compute-efficient processing, and handling rapidly changing data (Dev, 2023).

2.2.9 Firebase

Firebase is a cross-platform backend as a service that handles the server-side tasks of app development, such as database management, user authentication, cloud storage, scalability, and a lot more without worrying about infrastructure.

2.2.10 Flutter

Flutter is an open-source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase (Flutter by Google, n.d.).

2.2.11 NoSQL

NoSQL is a non-relational type of database that stores data in a format that's different from relational tables. Data are stored in key-value, multimedia, document, columnar, graph formats, external files, etc which makes the schemas of NoSQL very flexible and thus contributes to building modern applications.

2.2.12 Spot Occupancy

NoSQL is a non-relational type of database that stores data in a format that's different from relational tables. Data is stored in key-value, multimedia, document, columnar, graph formats, external files, etc which makes the schemas of NoSQL very flexible and thus contributes to building modern applications.

2.3 Review of Similar Works

A lot of academic research has been conducted over the past decades on smart parking systems but a few countries in the world have deployed some of the proposed solutions from the academic research. Narrowing it down to Rwanda there is even less deployment of practical solutions and relatively few research conducted in the country on the topic.

2.3.1 The Smart Car Parking System in Rwanda

Esperance Nyiransabimana, Emmanuel Ndashimye, and Richard Musabe's conference paper on "The Smart Car Parking System in Rwanda" focuses on finding and booking available parking within the city using IoT. The conference paper aims at designing the smart car parking system, to detect automatically available parking slots, to send data about parking status to the cloud, to display it to the smart phone using android application and to compare the existing data to new designed system. The smart car parking system developed in this work enables vehicle occupancy, monitoring and managing of available parking space in real-time. However, this system does not have employed a more robust means to digitalize the payment and the gate automation is is not smart enough to detect which vehicle is

accessing entering or leaving the parking. Also, there is no means throughwhich the driver is able to book a specific parking slot within the parking lot using a more intuitive user interface.

2.3.2 iParker—A New Smart Car-Parking System Based on Dynamic Resource Allocation and Pricing

iParker, proposed in (A. O. Kotb, Y. -C. Shen, X. Zhu and Y. Huang, 2016), solves the current parking problems by offering guaranteed parking reservations with the lowest possible cost and searching time for drivers and the highest revenue and resource utilization for parking managers. New fair pricing policies are also proposed that can be implemented in practice. The new system is based on mathematical modeling using mixed-integer linear programming (MILP) with the objective of minimizing the total monetary cost for the drivers and maximizing the utilization of parking resources.

2.3.3 An IoT-based Eco-Parking System for Smart Cities

This paper aims to design green parking and a truly sustainable solution that reduces CO2 emissions, congestion, and commuting times. This is achieved by developing a proof-of-concept of an original IoT-based and eco-friendly parking system. This system, named Eco-Parking, uses a tag or mobile application for parking id authentication, relies on IR sensors to monitor the real-time parking space availability using MQTT, and allocates the car park space according to vehicular emission class. It is clear that this paper does not cover other aspect like payment and gate automation.

2.3.4 Improved utilization for “smart parking systems” based on paging technique:

This paper proposes a new technique “paging technique” which increases the utilization factor of parking slots. The proposed method takes advantage of the idle time that exists between two successful parking services in the same slot. Besides, it investigates the possibility of using the idle times from different parking slots to provide a continuous parking time for an additional car. The paging technique is optimally implemented using mixed-integer linear programming that maximizes the utilization factor for the parking slots with minimum car transitions. Moreover, a data model for the parking management system has been constructed while considering the three major customers, namely, regular, prepaid, and

walk-in customers. The difference between fixed and dynamic pricing for parking has been investigated.

This work like majority of the other works reviewed are focused on just a portion of the bigger process of giving drivers a seamless parking experience considering time and the environment as a major factor.

2.3.5 Kigali Veterans Cooperative Society (KVCS):

Kigali Veterans Cooperative Society (KVCS) employed an e-payment method for street parking using a short code linked with mobile money which is a brilliant approach (The New Times Reporter, 2016). This solution however uses a minimum form of digital payment while the entire parking experience including the time calculation is strictly manual. Further, there is no means through which drivers can be aware whether or not a destination they are going to has available parking or not.

2.4 Comparative Study

Though a lot of literature work has been done towards developing smart parking solutions for spot occupancy, parking lot finder, parking fair payment, parking lot gate automation, etc, there is still a gap in implementing these systems in Africa due to limited resources and the slow pace of emerging technology penetration in the continent. From a thorough review of other literature work on the topic, there are solutions being proposed but very few are applicable in real-world scenarios for the African context, starting with Rwanda.

In summary, most of the literature work on the topic also focuses on spot occupancy, parking lot finder, gate automation, payment methods, or a combination of one or two but not all of them. A variety of approaches have been employed in different literature.

2.5 Gap of the Study

Even though dozens of significant literature works have been proposed on the topic of smart parking and they are all tailored towards resolving similar problems being addressed in this research, the following are some gaps or shortfalls this research is aimed at narrowing dramatically:

1. **The gap between literature work and implementation:** most proposed systems around this topic exist only in literature or academic work. Due to this, there has not

been a real-world impact of the solution due to the lack of strategies for deploying them into society. The research works reviewed focus a lot more on the academic which has led to a huge challenge in implementing the systems for real-world use cases and hence leaves the actual problem unsolved through literature work about a solution.

2. **Lack of a well-rounded user-friendly solution:** None of the literature reviewed above provides an all-in-one solution for parking in the city of Kigali. A well-rounded parking lot solution includes spot occupancy, a handy automated payment solution, automatic check-in and check-out, and an automatic gate.

2.6 Research Contribution

Due to the gaps in the reviewed work literature work on the topic, this research will make the following contributions:

1. Provide a well-rounded parking lot solution harnessing the use of emerging technologies like IoT, cross-platform mobile application development, and Machine Learning technologies to streamline the parking experience of drivers and significantly reduce emission of greenhouse gases, time and petroleum spend in search of available parking lots.

2. Provide a relatively comprehensive mobile management tool for parking lot managers or owners.

CHAPTER THREE: EXISTING SYSTEM ANALYSIS

3.1 Introduction

A detailed analysis of existing parking systems, both manual and smart, is presented in this chapter. This chapter identifies, describes, and analyzes the working principles, features, and problems found with the existing parking lot systems and how drivers locate a parking lot in Kigali. The entire process of drivers locating a parking lot, parking their vehicles, checking out, and paying parking charges is analyzed, presenting the findings as proof of why Park Ease, my proposed smart parking system, is an ideal solution for Kigali at this time. Further, the requirement documentation for the new system is presented which will set the stage for the remaining chapters of this report and project implementation. Lastly, a conclusion is made from the analysis of the existing system in comparison with the proposed system highlighting distinctly the value of my proposed solution over existing solutions.

3.1 Working Principles of Existing Systems

Currently, there are two kinds of parking lot systems in Kigali, manual and smart systems. Despite the high penetration of technology in the world, there are very few smart parking systems being deployed in Kigali compared to the manual systems.

3.1.1 Smart Parking Systems in Kigali

There are a couple of smart parking systems that are currently being deployed in Kigali at various facilities. The international airport of Kigali has deployed a smart parking system that has an automated parking gate and a cashless payment using service points within the facility of the airport. The system uses cameras and plate number recognition algorithms to detect vehicles accessing and leaving the facility. While accessing the parking lot the driver has to stop at the gate and press a button on the gate to generate a token, kind of, which will be used by the driver to pay for the hours spent within the parking, that triggers the gate to open. When the driver is about to leave the facility, he has to walk to the service point within the parking lot to get the instructions to be followed to make payment for the hours spent. After payment is successful, the system gives the driver approximately 10 minutes to exit the parking. The system recognizes each vehicle with its plate number so, when a driver pays the system allocates 10 minutes to exit to the plate number. This means that, while the driver is driving out, the camera at the gate scans the plate number of the vehicle using plate number

recognition algorithms and checks the status of the vehicle to know if it satisfied all conditions(payment done and spent less than or equal to 10 minutes after payment). The gate will not be opened if any of the conditions evaluates to false. To determine whether or not the parking lot is fully occupied is done manually because the scope of most if not all of the parking lots does not cover spot occupancy.

Though the above description uses the Kigali International Airport as a case study, most if not all of the smart public parking systems in Kigali use a similar approach or a less technically inclined approach according to my research so far.

3.1.2 Manual Parking Systems in Kigali

Due to many factors including but not limited to, the economy status of the parking lot management, exposure to advanced technology, cost of smart parking systems, drivers' potential to pay for additional parking charges, unavailability of a well proven all in one parking solution, etc, most of the parking lots in Kigali uses a manual approach. The manual approach requires the driver to drive to a parking lot, and inquire if there are available parking slots within the parking lot, a security guard at the entrance will open the gate while the driver drives in and searches for or is directed to a specific slot. On the other hand, if there isn’t an available parking slot, the driver has to be cruising the city in search of available parking which consumes more time, petroleum, and is environmentally unfriendly.

This system requires the physical presence of a record-keeping to record the time the driver checks in and checks out to determine the charges to be paid. The payment of the parking lot fees is done manually using cash or mobile money and the record is taken manually also. Within the manual parking approach, the entire workflow is done in an old-fashioned way.

3.2 Existing System Problem Analysis

Both the manual and smart parking systems being used in Rwanda one way or another have contributed to solving some problems of parking within the city of Kigali. Below, I present problems of the existing public parking lot systems, both manual and smart, in Kigali and conclude with why both systems combined still have problems and need to be addressed.

3.2.1 Manual Public Parking System in Kigali Problem Analysis

Due to the rapid growth of personal vehicle owners, in the 1920s, cities across America started to allocate space for parking lots that were either owned and managed privately by commercial and retail associations or owned by public entities and maintained by private operators. The problem of cars being left unattended on streets or in public areas was a major problem for cities at the turn of the 20th century (Ben-Joseph, 2020). This proves that the manual approach of public parking lot systems in Kigali has succeeded in solving the above-mentioned problems in addition to the security of those vehicles thereby giving the vehicle owners peace of mind while they go about their other activities in the city. Though the manual approach has contributed immensely to solving the problem being addressed in the 20th century, as technology evolves and time goes on, due to enormous research, this manual system has a lot of problems as follows:

1. **Time wastage:** A lot of time is spent by drivers cruising the city to find an available parking lot. Worse enough if a parking lot is found but there is no parking slot available. There is also a waste of time in trying to pay charges since the payment process is manual, hence it will require a cue.

2. **Additional Fuel Consumption:** Due to the extra time in search of a parking lot, more fuel will be consumed which when accumulated over a month or year will be a very astronomical amount being wasted.

3. **Environmentally Unfriendly:** The more the driver drives around in search of a parking lot, the more fuel is being consumed and the more emission of greenhouse gasses into the atmosphere which contributes to GWP making the earth unsuitable for life.

4. **More Labor Force Required:** Because every task within the parking lot is manual, it requires human interactions which in turn is expensive.

5. **Improper Management of Records:** Because all records are being managed manually, all of the challenges that come with hard copy record keeping are also a problem.

6. **Susceptible to Error:** Because everything is done by humans and humans are prone to error, there is a high risk of mistakes when calculating the price of a client who parks at the facility.

3.2.2 Smart Public Parking Systems in Kigali Problem Analysis

Though the smart parking systems in Kigali have contributed to digitalizing the manual parking systems and have solved some of the problems it, there are a couple of challenges that are not being addressed. Though there are a lot of other problems with the current smart parking systems being used in Kigali, I will only address those within the scope of this research focus. After analyzing existing smart public parking lot systems in Kigali, the following are the problems I have realized taking into consideration the scope of this research:

1. **Time Wastage:** Though this system significantly reduces the amount of time spent on parking at public parking lots in Kigali, it however still wastes some of the precious time of the drivers. While analyzing the entire workflow of this system, I have realized that it will take between 45 seconds to about 2 minutes to stop at the gate, press a button on the attached to the get, wait for the machine to print a ticket, take out the ticket, before the gate opens, and for the driver to pick up and start moving in. Eliminating this process within the workflow will save about 2 minutes which is a significant win taking into consideration the significance of time in this era.

2. **Relatively Environmentally Unfriendly:** Though the effect is relatively low, it however still contributes to additional emission of greenhouse gasses for worse case scenarios where drivers may only realize that there is no parking slot after accessing the facility and driving around the lot to locate a slot.

3. **Lack of Ease and Automation in Payment Processing:** though payment using this system can be done electronically, it requires the drivers to physically go to the service point for instructions or help. This is inconvenient and relatively consumes time especially when there is a cue because the airport is most of the time.

4. **Lack of Slot Availability Status:** There is no way for drivers to be informed ahead of time whether or not there is a parking slot available within the parking. Take, for example, Kigali International Airport, if a driver drives in they may find out that there is no slot available. This leads to inconvenience and because his time has started counting if he tries to exit the parking immediately, he will be forced to pay a minimum of an hour.

5. **Lack of Remote Interface between Drivers and the System:** This system has no easy, handy and accessible interface between the driver and the system. That is, there is no way for the driver to book a slot remotely, automate his payment process, and access his/her history of parking lot usage or payment.

3.2.3 Conclusion of Analysis

After a careful study and analysis of the existing public parking systems in Kigali with a focus on the problems they are solving and the approach they have taken, there is proof of an enormous contribution and progress towards solving the problems this research project is addressing. Though significant progress has been made by the existing systems, the approach the smart public parking systems have taken has led to the problem not being resolved more efficiently. Also, the issue of spot occupancy detection in real-time, parking lot locator and availability status, and automating the payment process was not handled leaving the solution incomplete. This is why, this research is aimed at filling the gaps in the existing systems to provide a well-rounded solution for metropolitan area parking that will remarkably save drivers' time, reduce fuel consumption, and relatively reduce the emission of greenhouse gasses within the atmosphere, contributing to a more environmentally friendly city.

3.3 Proposed Solution

After a diligent review analysis of existing smart and manual public parking systems in Kigali, concluding the fact that there is no well-rounded public parking system that adequately answers the question, "How can drivers in Kigali city easily find available parking lots without consuming more fuel and wasting time in a more environmentally friendly way?", a prototype of an all-in-one IoT and ML based Smart Parking System is proposed in this research to answer the above question and achieve the research objectives. The following

are the key features/components of the solution along with their detailed description of how they are solving the underlying problems:

1. **Real-time Spot Occupancy:** This component of the system is used to determine whether or not a specific slot within a given parking lot is available. This component of the system helps to determine whether or not a parking lot has space which helps to reduce the wastage of time of the driver, reduces the emission of greenhouse gasses, and reduces the consumption of fuel.

   An ultrasonic sensor is placed at each slot within a parking lot which calculates the distance from the target (vehicle) and sends a signal indicating the slot occupancy to the Firebase Real-time Database. The slot is considered occupied if the distance between the ultrasonic sensor and the target (car) is less than 30cm otherwise, not occupied.

2. **Available Parking Lot Locator/Finder:** This feature of the system uses the Real Time Spot Occupancy and Mobile application components of the system to help drivers locate/find parking lots around them and to easily check their availability. This feature remarkably saves time, and fuel consumption and of course reduces the emission of greenhouse gasses. If all slots within a given parking area are flagged as occupied in the Firebase Real-time Database, that parking lot is flagged as unavailable on the map of the mobile application to avoid drivers driving to the parking lot to find out that there is no available slot.

3. **Parking Lot Gate Automation:** To save the 45-120 seconds spent at the gate of the existing system during the check-in process as mentioned in the analysis of the existing system, this component is developed. Cameras are placed at the gate to perform real-time image capture of every vehicle that approaches the gate. The image is immediately passed as input to the OpenCV to perform image processing and object detection. The processed image is then passed as input to the pytesserract ML model to extract the license plate number as text from the image which is stored in the Firebase Real-time Database as a checked-in vehicle. The component of the system runs on a Raspberry Pi capable of running complex ML models.

The same procedure is followed while checking out but the system will check whether the plate number extracted from the image taken by the camera in real-time has been paid or not. The gate will only open if that plate number has a paid flag otherwise, it remains closed.

4. **Seamless Parking Fair Payment:** The payment process can be done manually using the mobile application through the stripe payment API.
   That is the user can initiate the payment from the mobile application or respond to the push notification sent to the user's device when the driver is checking out.

5. **Robust Mobile App for Drivers:** A very easy, user-friendly cross-platform mobile application will be developed using the Flutter UI toolkit Firebase to power the backend. The mobile application will be used as a direct means through which users(drivers) can interact with the system to locate available parking lots, check the number of slots available within a specific parking lot, book a parking lot, make payment for parking fairs manually or automatically, view booking and payment history, etc. The application will leverage the data streamed from the Spot Occupancy and Parking Lot Gate Automation components of the system to the Firebase real-time database. The same database instance will be used for all of the components of the system which makes the flow of data within the system seamless. The Firebase Real-time Database makes the process very fast and efficient.

CHAPTER FOUR: ANALYSIS & DESIGN OF NEW SYSTEM

4.1 Introduction

From the analysis of the existing system and other related research, this chapter focuses on the analyzing and designing the proposed system, Park Ease, which is tailored to address the limitations and weaknesses of the existing parking system within the city of Kigali. This chapter will identify the development methodology, the models, the CASE (Computer Aided Software Engineering) tools, the system architecture design, and the functional and non-functional specifications of the system.

4.2 Development Methodology: Object-Oriented Analysis and Design Methodology (OOADM)

The development methodology for this system design is the Object-Oriented Analysis Design and Methodology(OOADM) which is entirely based on Object-oriented programming (OOP) - a programming paradigm that uses objects(instances of classes) to organize and structure code. OOAD extends these principles to the entire software development process which brings the following benefits to the entire system: Modularity and Reusability, Encapsulation, scalability and flexibility, efficient code maintainability, improved understanding of the visual representation of the system components because of the UML diagrams, Enhanced Software Quality, and Reusable Design Patterns. Due to the above benefits of using the OOADM and the heterogeneous nature of this system, this methodology has been selected to best fit the system design.

4.3 Development Models

A model is a representation of an important aspect of the real world (UNILAK CIS Department, n.d.). Because the OOADM is being used as the methodology for this system design, the following models are being designed to give a comprehensive description of each aspect of the system: use case diagram, activity, sequence diagram, class diagram, and deployment diagram.

4.3.1 Use Case Diagram

Use case diagrams are a type of Unified Modeling Language (UML) - a standardized modeling language used in software engineering and other fields to document, design, and

communicate the structure and behavior of a system diagram - that summarizes the details of a system's function by depicting outside users' interactions with it. They are a great way to communicate high-level concepts about a system to managers or other business stakeholders. They are an effective tool for general functionality analysis without the need to get into the specifics of how those functionalities are implemented on a technical level. Use case diagrams are also an important part of the system planning process because they allow the team to visualize the system's functional requirements and translate them into design priorities (Gliffy, 2023).

4.3.1.1 System Use Case Description

A use case description is a description that provides a comprehensive understanding of a specific use case within a system. A use case itself is a description of how a system interacts with an external entity (known as an actor) to accomplish a specific goal. The use case description elaborates on the use case by providing more detailed information about its functionality, interactions, and possible scenarios. The use case descriptions for the major use cases of this system, ParkEase is presented below in a tabular format with each use case being represented using a table.

Table 4.1: User Registration Use Case Description

Name	User Registration
Description	This use case describes the process of a user registration process.
Primary Actor	Driver
Secondary Actor	Mobile App (The System)
Preconditions	1. The user has downloaded the ParkEase App on mobile phone from Play Store or App store. 2. The user has access to internet on their device.
Main Flow	1. The user clicks Register button on the landing page. 2. The System presents the registration form. 3. The user enters the required form information and submit it. 4. The System validates the user's information. 5. The System creates an account for the user. 6. The system notifies the user of account creation success. 7. The System displays the login form to the user.

Alternative Flow	1. If the user enters an email that is already registered, the system notify the user to use another email.
Exceptions	1. If step 4 fails, the user is notified the problem and is instructed on a way forward. 2. If step 5 fails, the user is notify of a system error and instructed on how to go forward. The system starts the registration process from step 2.
Post Conditions	1. The user account has been created and the user can safely login to the mobile app.

Table 4.2: Authentication Use Case Description

Name	User Authentication
Description	This use case describes the process of a user authentication process.
Primary Actor	1. Driver 2. Parking Lot Manager 3. System Administrator
Secondary Actor	Mobile App (The System)
Preconditions	1. The user possesses a valid account with the parking system. 2. The user has the right authentication credentials. 3. The user has access to the internet.
Main Flow	1. The user opens the mobile app. 2. The navigate to the login form. 3. The System displays the login form. 4. The user enters his/her correct user credentials. 5. The system validates the user credentials. 6. If the user account is valid, the System login the user to the app.

Alternative Flow	1. The user can reset his/her password if it was forgotten.
Exceptions	1. If the user credential is incorrect, the user is notified and is requested to try again. 2. If the user is not connected to the internet, the system notify the user to connect to the internet to proceed.
Post Conditions	1. The user is authenticated. 2. The user is redirected to the dashboard.

Table 4. 3: Manage System Config & Managers Use Case Description

Name	Manage System Config & Managers
Description	This use case describes the system wide configurations including managing parking lot managers, and etc.
Primary Actor	1. System Admin
Secondary Actor	1. Mobile App (The System)
Preconditions	1. The User must be authenticated and connected to the internet. 2. The User must be authorized as admin.
Main Flow	1. The User onboard a parking lot within the system. 2. The User creates an account for a parking lot manager. 3. The User assigns a parking lot to a parking lot manager. 4. The User generates report.
Alternative Flow	-
Exceptions	1. If the user is not connected to the internet, The User is notified and the process starts from the previous step of when the internet failed.
Post Conditions	1. The System is configured and ready for operation.

Table 4.4: Reservation Use Case Description

Name	Reservation
Description	This use case describes the process of a driver reserving a specific slot within a parking lot.
Primary Actor	1. Driver
Secondary Actor	1. Mobile App (The System) 2. Lot Occupancy System
Preconditions	1. The Driver must be logged in to the mobile app. 2. The Driver must have access to the internet.
Main Flow	1. The Driver navigates to the nearby menu. 2. The System displays the map with all of the available parking lots as markers. 3. The User selects the parking lot he wishes to park at. 4. The System display a reservation screen with the a short form. 5. The User enters the reservation details and submit the form. 6. The System validates the form fields for any error. 7. The System creates a reservation document in the Firestore database. 8. The lot Occupancy System updates the occupancy of the slot. 9. The System notifies the user of the reservation success the initiation of a 10 minute counter. 10. The System starts a 10 minutes countdown.
Alternative Flow	1. From step 2, The User can choose to enter a location he would like to park at and continue the flow. 2. If the 10 minutes count down ends without the User checking in, the reservation is cancelled automatically. 3. The User can cancel the reservation at any point.

Exceptions	1. If the reservation details is invalid, the system notifies The User and moves back to step 4. 2. If the reservation document creation fails, the user is notified and the process moves back to step 4. 3. If the user is not connected to the internet, The User is notified and the process starts from the previous step of when the internet failed.
Post Conditions	3. The User has successfully reserve a parking slot within the parking lot. 4. The user is notified of the reservation success. 5. The User is redirected booking screen.

Table 4.5: Check-In Use Case Description

Name	Check In
Description	This use case describes the process of a vehicle (Driver) checking into a parking Lot.
Primary Actor	1. Driver
Secondary Actor	1. Gate Automation System 2. Lot Occupancy System
Preconditions	1. The Gate Automation System must be connected powered on. 2. The Gate Automation System must be connected to the internet.
Main Flow	1. The Driver drives towards the park lot's entry gate. 2. The Gate Automation System powers on the camera and capture the image of the vehicle's license plate. 3. The Gate Automation System extracts the plate number from the image captured.

	4. The Gate Automation System checks in the Reservation collection within the Firestore database for an active reservation from the plate number. 5. The Gate Automation System creates a check-in log for the vehicle. 6. The Gate automation system updates the occupation status of the slot that was reserved. 7. The Occupation System updates the occupation status of the slot to True (meaning occupied). 8. The Gate Automation System update the status of the entry gate to True(meaning open) 9. The Gate Automation System rotates the servo motor 180 degrees clock-wise, hence the gate opens. 10. The Gate Automation System runs a 10 seconds countdown to allow the vehicle to pass through the gate. 11. The Gate Automation System activates the ultrasonic sensor to read the distance of vehicle away from the sensor. 12. The Gate Automation System checks if the vehicle has passed using the ultrasonic distance read. 13. The Gate Automation closes the gate (rotates the servo motor 0 degrees counter-clock wise).
Alternative Flow	1. If there is no active reservation for the Driver, the Gate Automation System creates a reservation for a guest user and continues the flow from step 5. 2. If the vehicle has not passed, the Gate Automation System loops on steps 10, 11, and 12 until the vehicle passes.
Exceptions	1. If the Gate Automation System is not connected to the internet or have an internet failure, a beep alarm is made.
Post Conditions	2. The Driver is checked in.

Table 4.6: Check Out Use Case Description

Name	Check Out
Description	This use case describes the process of a vehicle (Driver) checking out of a parking Lot.
Primary Actor	1. Driver
Secondary Actor	1. Gate Automation System 2. Lot Occupancy System 3. Mobile App
Preconditions	1. The Gate Automation System must be connected powered on. 2. The Gate Automation System must be connected to the internet. 3. The Mobile App must have access to the internet. 4. The Driver(Vehicle) must have been checked in and not yet checked out.
Main Flow	1. The Driver drives towards the park lot's exit gate. 2. The Gate Automation System powers on the camera and capture the image of the vehicle's license plate. 3. The Gate Automation System extracts the plate number from the image captured. 4. The Gate Automation System checks in the Check-in Log collection within the Firestore database to be sure that the vehicle is was check-in. 5. The Gate Automation System Sends a push notification to the driver's device to authorize the payment. 6. The Driver authorizes the payment using from the prompt of the push notification. 7. The mobile app computes the parking bill. 8. The Mobile App shows the payment form. 9. The Driver enters his/her payment details and submits.

	10. The Mobile app calls the Stripe payment API and process the payment. 11. The Mobile App creates a payment collection within the Firestore Database. 12. The Gate Automation System updates the reservation payment status to True. 13. The Gate automation System updates the occupation status of the slot that was reserved. 14. The Occupation System updates the occupation status of the slot to True (meaning occupied). 15. The Gate Automation System updates the status of the exit gate to True(meaning open) 16. The Gate Automation System rotates the servo motor 180 degrees clock-wise, hence the gate opens. 17. The Gate Automation System runs a 10 seconds countdown to allow the vehicle to pass through the gate. 18. The Gate Automation System activates the ultrasonic sensor to read the distance of vehicle away from the sensor. 19. The Gate Automation System checks if the vehicle has passed using the ultrasonic distance read. 20. The Gate Automation closes the gate (rotates the servo motor 0 degrees counter-clock wise).
Alternative Flow	1. If there is no Check-in Log for the Driver, the Gate Automation System alerts the by beeping. The Driver gets helped by a staff at the parking. 2. The driver can authorize the payment from the app if he doesn't get the prompt for any reason. 3. If the vehicle has not passed, the Gate Automation System loops on steps 10, 11, and 12 until the vehicle passes.
Exceptions	2. If the payment process fails, the user is notified and payment

	process goes back to where it fails for continuation. 3. If the Gate Automation System is not connected to the internet or have an internet failure, a beep alarm is made.
Post Conditions	3. The Driver is checked out successfully.

Table 4.7 Generate Report Use Case Description

Name	Generate Report
Description	This use case describes the process of generate basic reports of the system.
Primary Actor	1. Parking Lot Manager 2. System Admin
Secondary Actor	1. Mobile App (The System)
Preconditions	1. The User must be authenticated and connected to the internet. 2. The User must be authorized as an admin or parking lot manager.
Main Flow	1. The User navigates to the Report Menu 2. The System renders the report screen. 3. The User selects the kind of report to generate. 4. The System generates the report. 5. The System displays the report to the user. 6. The User can print, export, or share the report the report.
Alternative Flow	-
Exceptions	1. If the user is not connected to the internet, The User is notified and the process starts from the previous step of when the internet failed.
Post Conditions	4. The System is configured and ready for operation.

4.3.1.2 System Use Case Diagram

Figure 4.1: System Use Case Diagram

4.3.2 Activity Diagram

4.3.2.1 Introduction

Activity Diagrams are used to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. The sequential and concurrent processing of activities can be depicted using an activity diagram. That is, it focuses on the condition of flow and the sequence in which it happens. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths while the activity is being executed. An activity diagram has the following significant notations (GeeksforGeeks, 2024).

4.3.2.2 Main System Activity Diagram

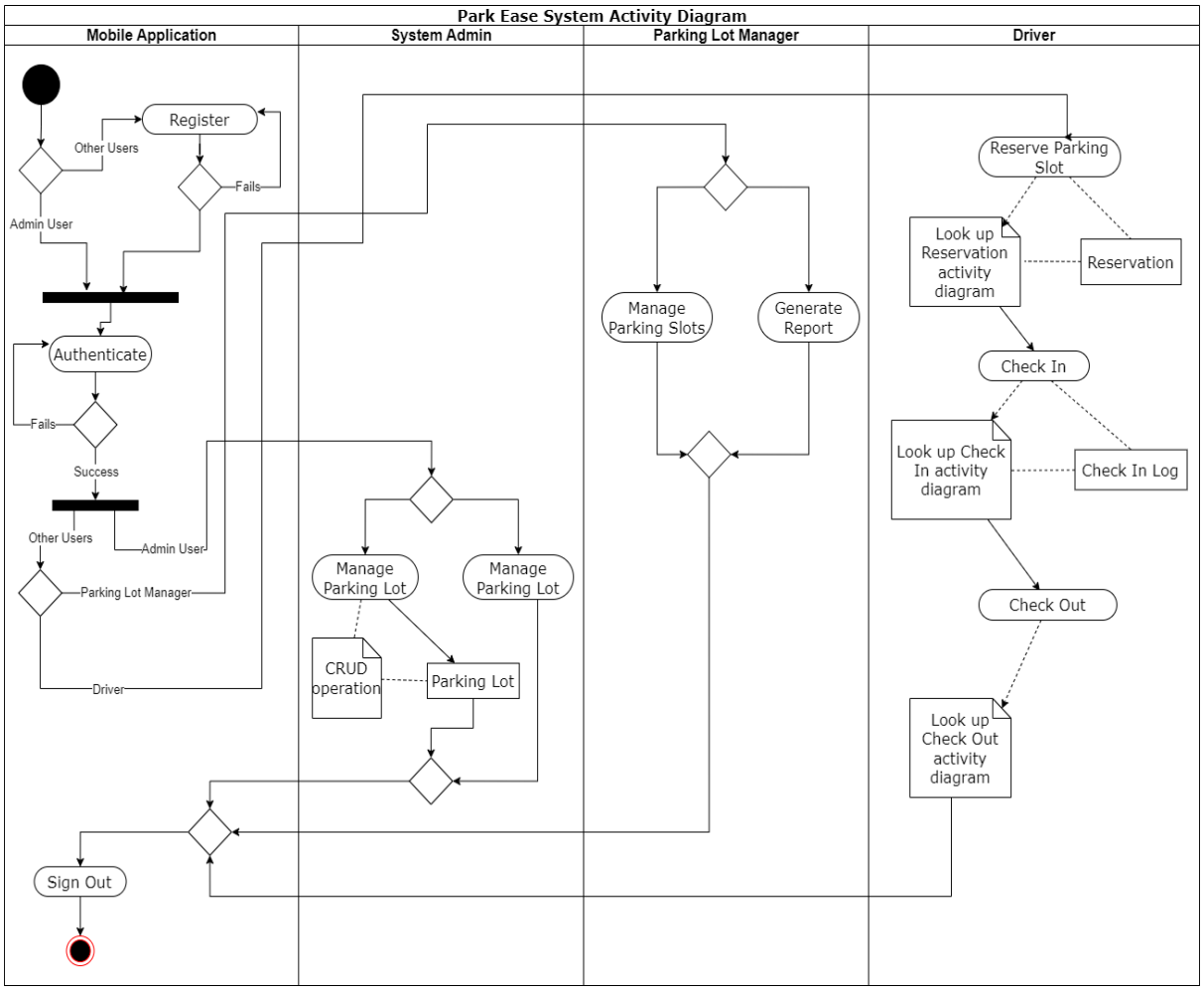

Figure 4.2: System Level - Main Activity Diagram

4.3.2.3 Reservation Activity Diagram

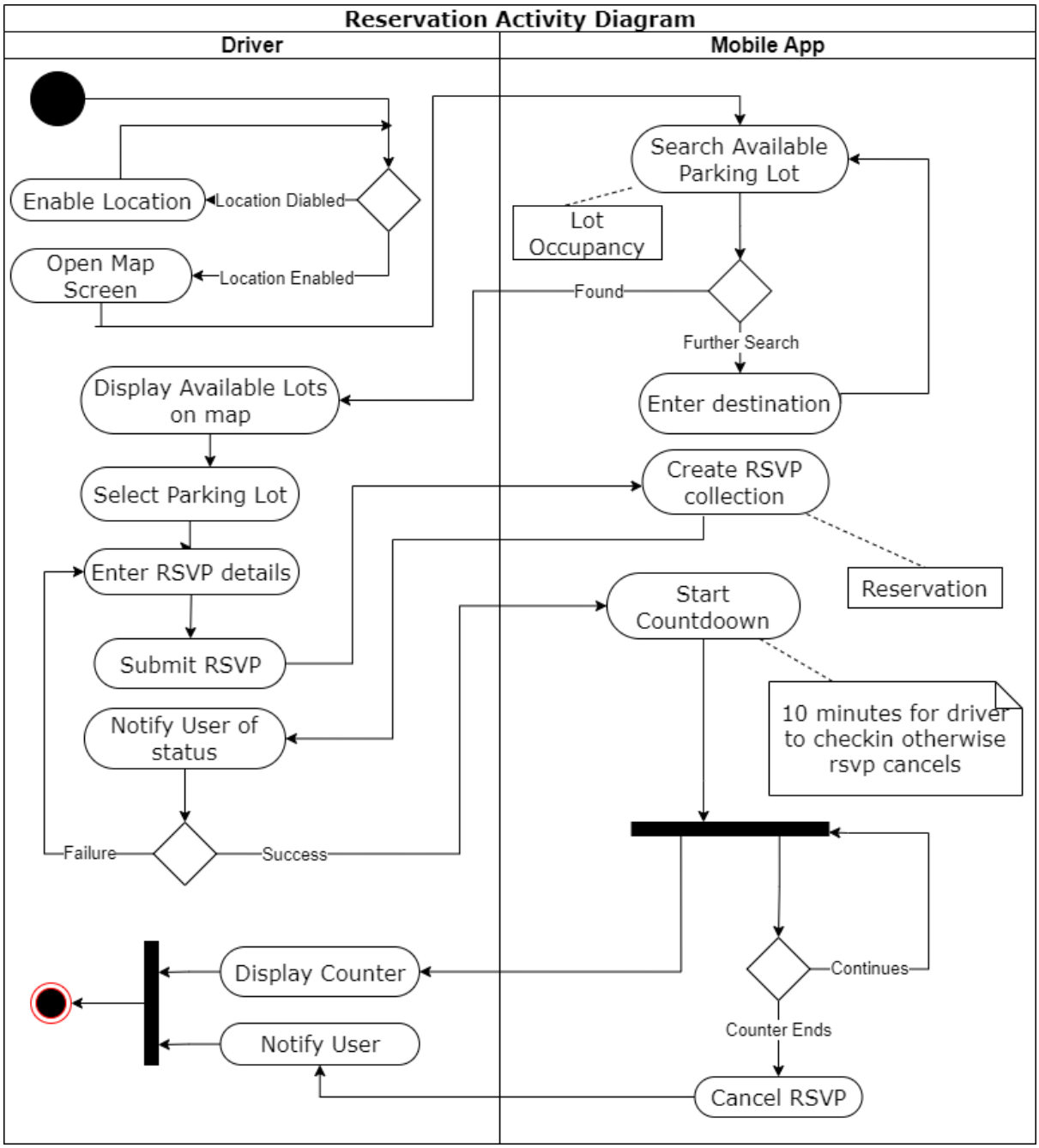

Figure 4.3: Sub Level - Reservation Activity Diagram

4.3.2.4 Check-In Activity Diagram

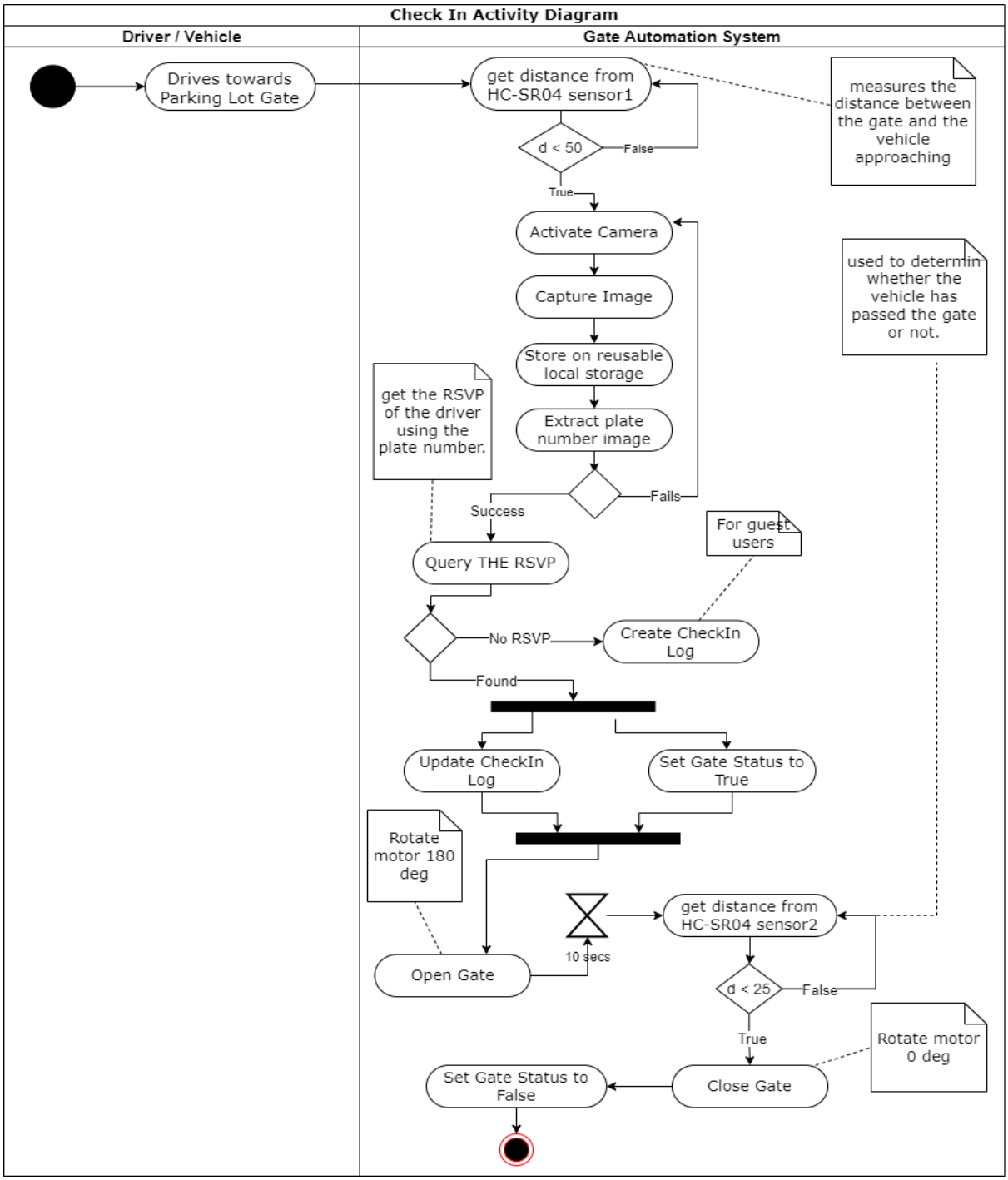

Figure 4. 0.4: Sub Level – Check-In Activity Diagram

4.3.2.15 Check-Out Activity Diagram

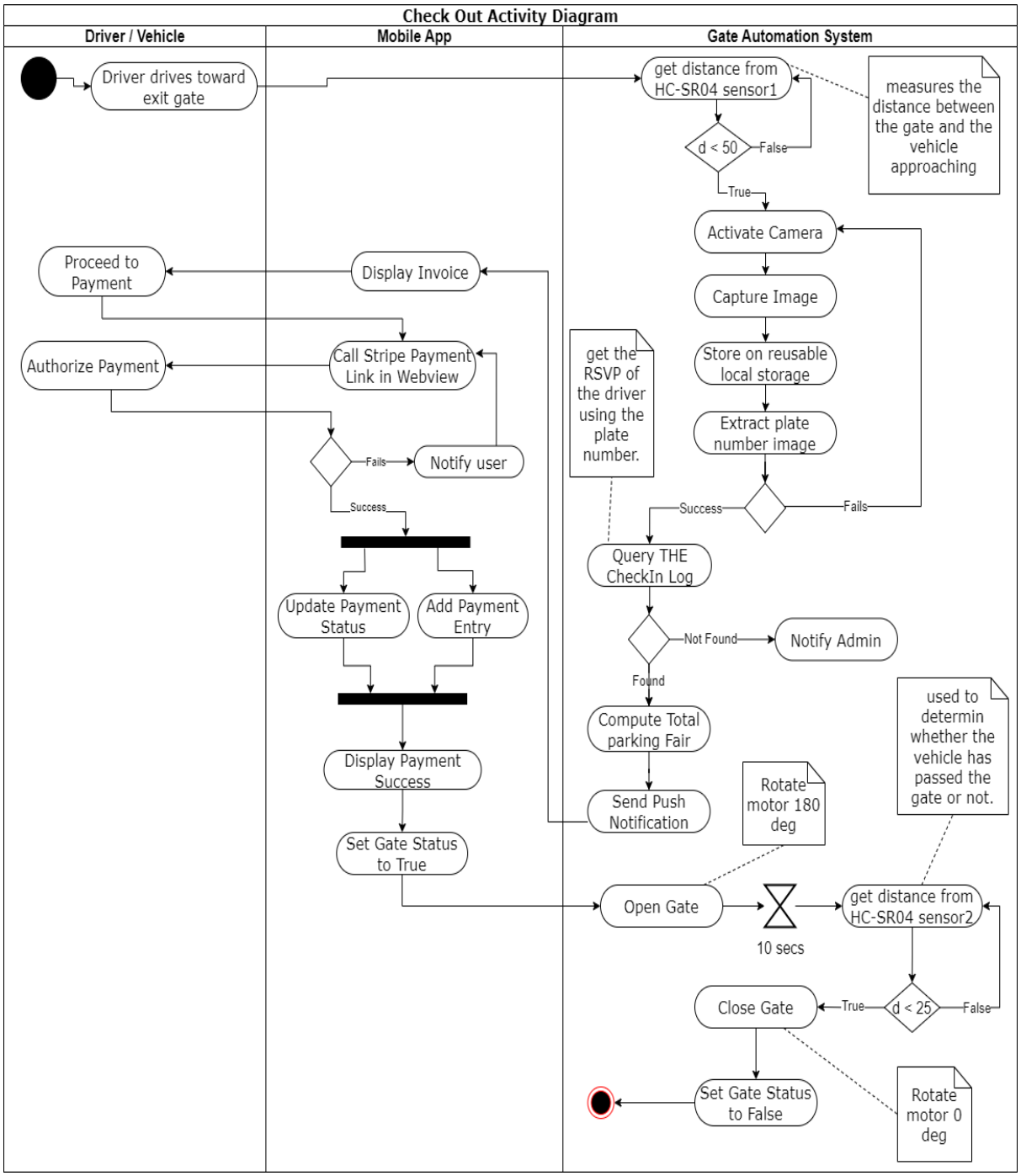

Figure 4.5: Sub Level – Check-Out Activity Diagram

4.3.3 Sequence Diagram

4.3.3.1 Introduction

A sequence diagram, sometimes termed as event diagrams or event scenarios is one of the most commonly used interaction diagrams. It depicts the interaction between the objects in a sequential order i.e. the order in which these interactions occur. It describes how and in what order the objects in a system function (GeeksforGeeks, 2024).

4.3.3.2 Reservation Sequence Diagram

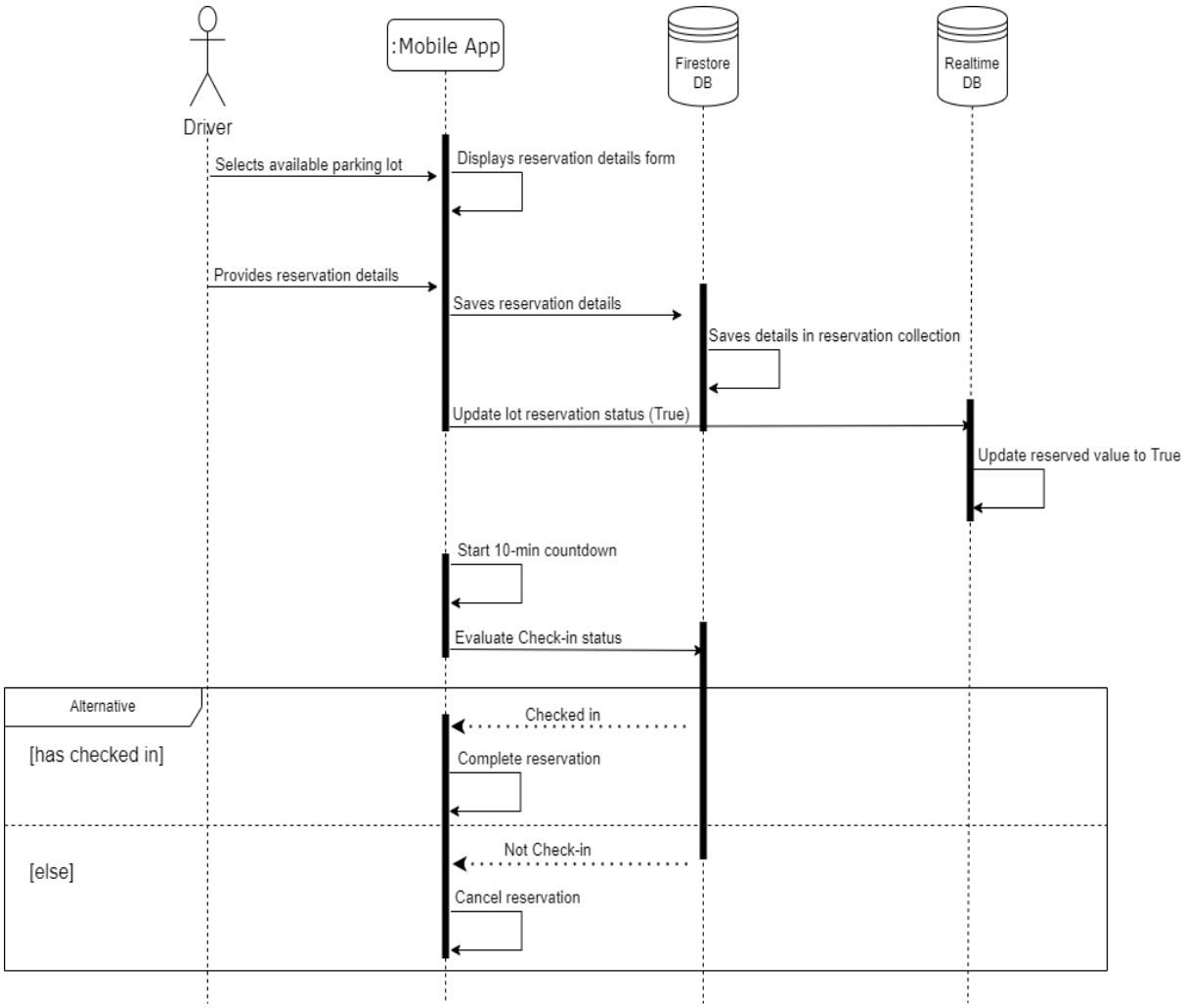

Figure 4.6: Sequence Diagram of the Reservation Use Case

4.3.3.3 Check In Sequence Diagram

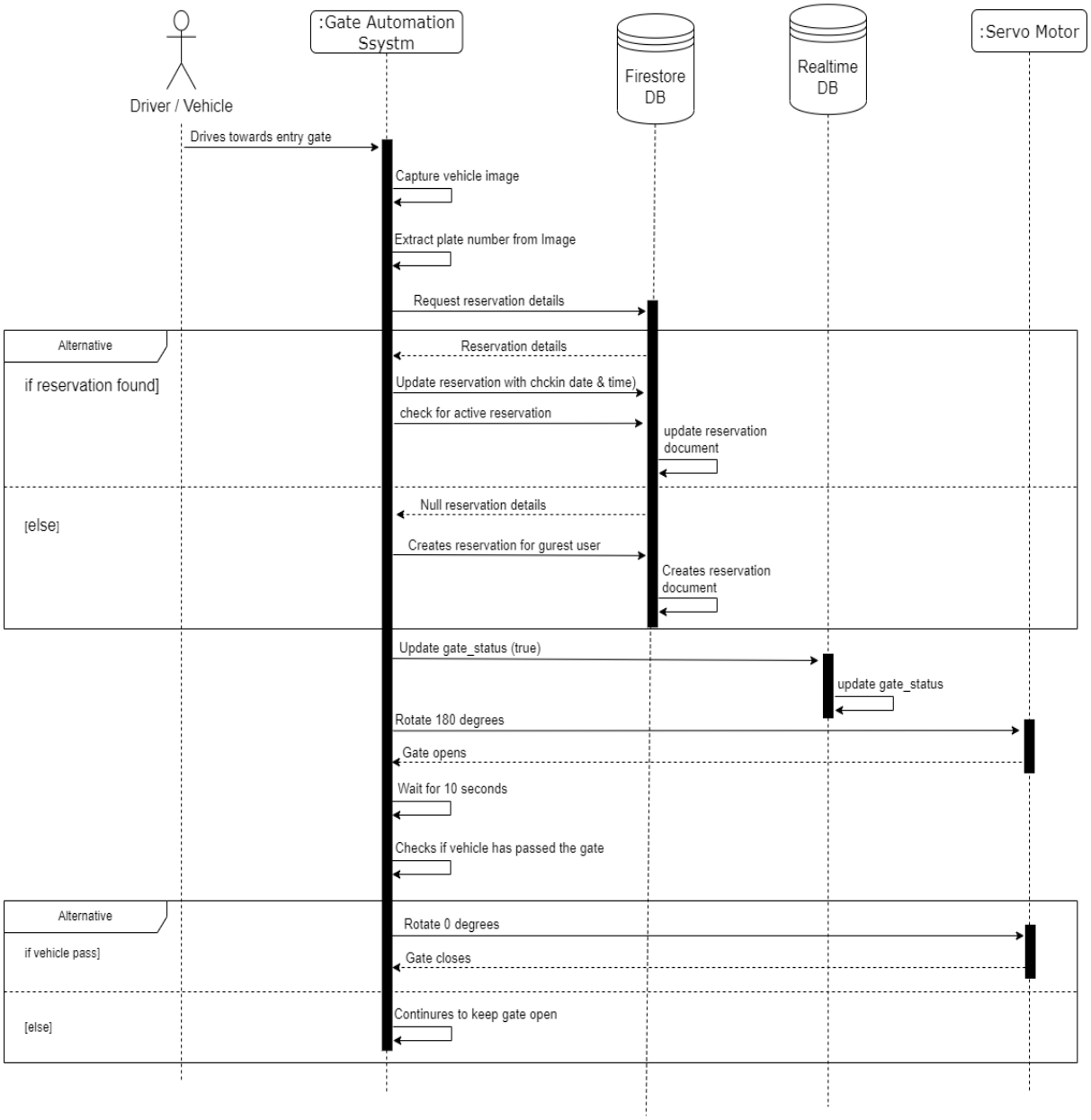

Figure 5.7: Sequence Diagram of the Check-In Use Case

4.3.3.4 Check-Out Sequence Diagram

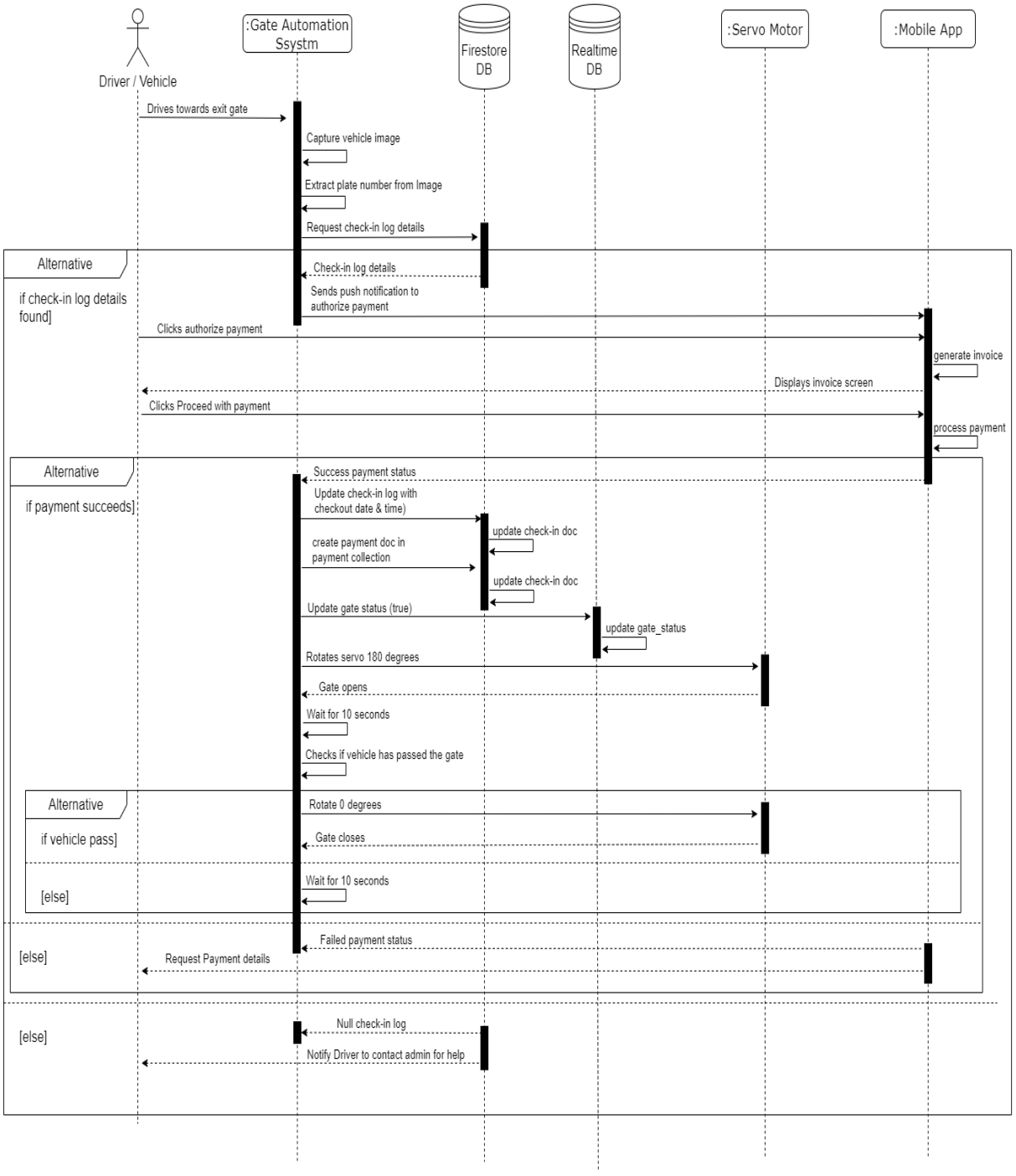

Figure 4.8: Sequence Diagram of the Check-Out Use Case

4.3.4 Class Diagram

4.3.4.1 Introduction

Class diagrams are a type of UML (Unified Modeling Language) diagram used in software engineering to visually represent the structure and relationships of classes within a system i.e. used to construct and visualize object-oriented systems. In these diagrams classes are depicted as boxes, each containing three compartments for the class name, attributes, and methods. Lines connecting classes illustrate associations, showing relationships such as one-to-one or one-to-many (Jain, 2024).

4.3.4.2 UML Class Notation

Class notation is a graphical representation used to depict classes and their relationships in object-oriented modeling.

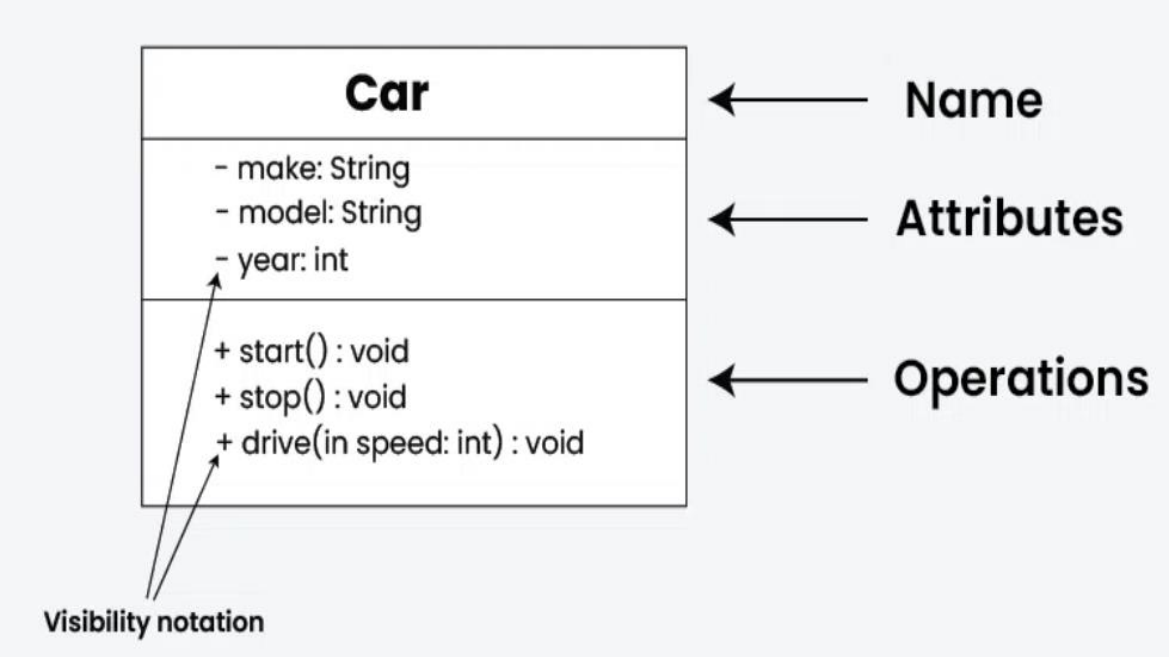

Figure 4.9: UML Class Notation

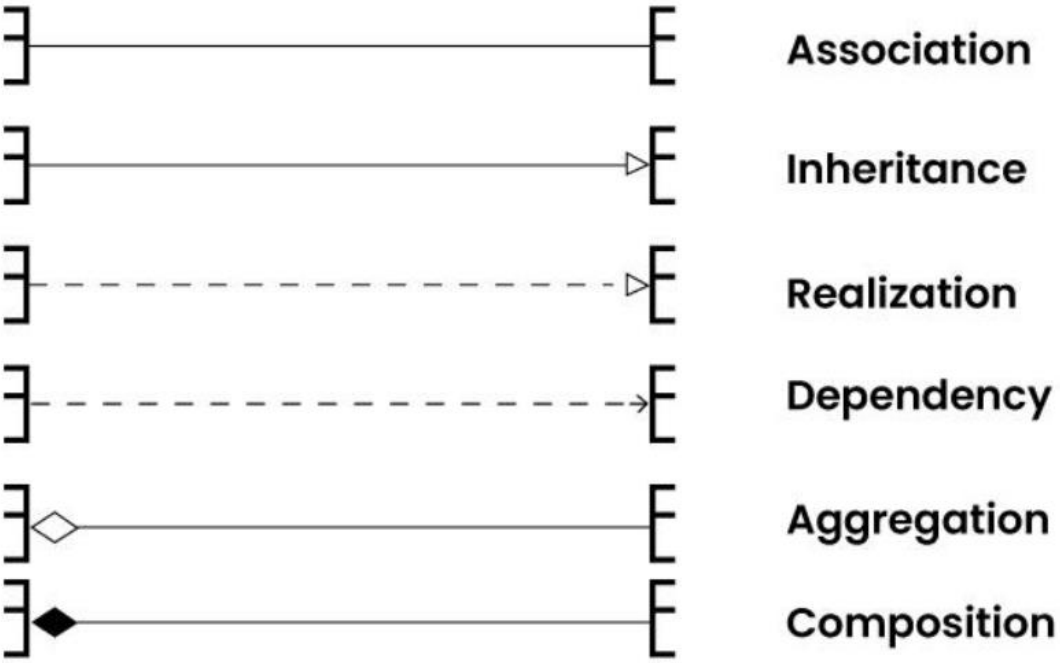

Figure 4.10: Relationships Representation

4.3.4.5 System Class Diagram

Due to the complexity and heterogeneous nature of this system, a NoSQL database was used due to its relaxed constraints and flexible nature. Therefore, the below class diagram is not exactly designed as the actual database. The class diagram provides a more detailed understanding of the relationship between the classes. Though the actual database structure is the same as the class diagram, they both are very similar only that few rules that are adhered to in SQL are relaxed in NoSQL databases to compensate for complex systems like ParkEase. The class diagram below has been designed in a generic way relative to the database type (SQL or NoSQL) so it can give a comprehensive guide to anyone irrespective of his/her technical knowledge, reviewing it.

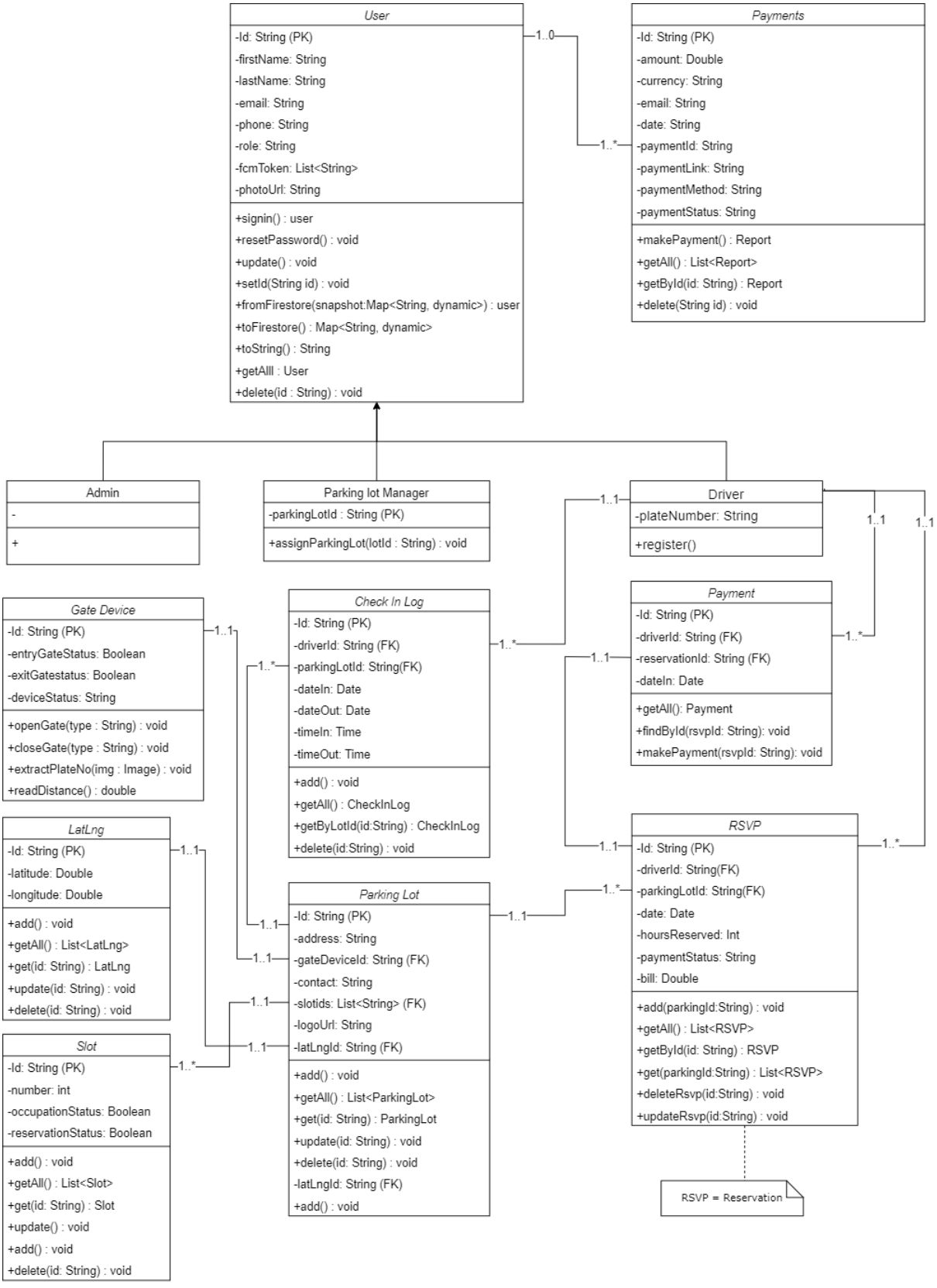

Figure 4.11: System Class Diagram

4.3.5 Deployment Diagram

A UML deployment diagram depicts a static view of the run-time configuration of hardware nodes and the software components that run on those nodes. UML deployment diagrams show the hardware for your system, the software that is installed on that hardware, and the middleware used to connect the disparate machines to one another (Ambler, 2010, 139-147).

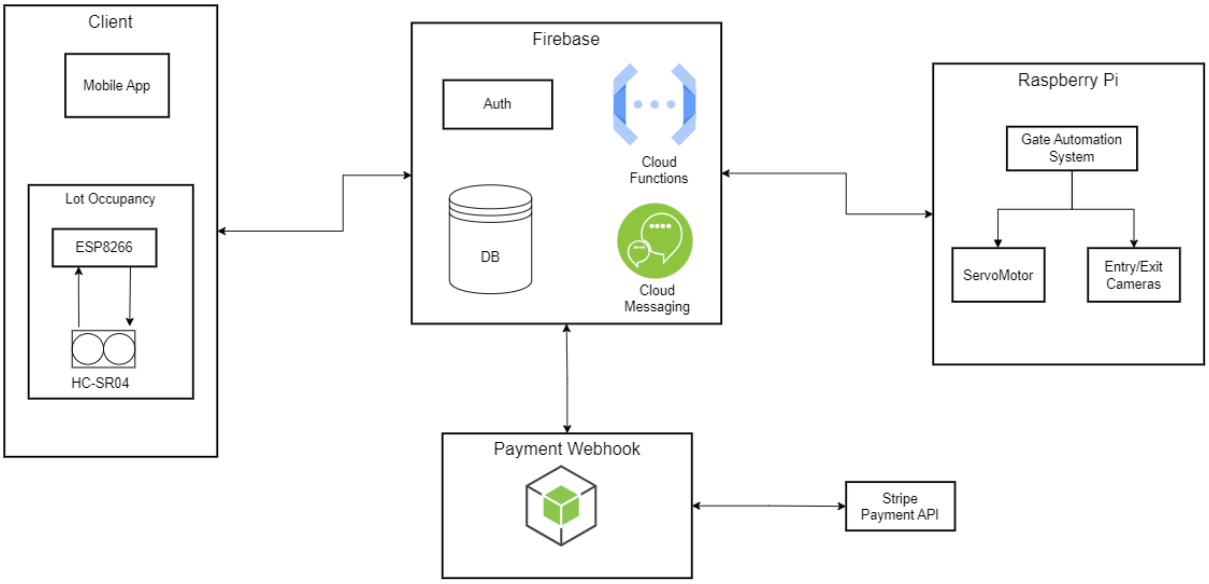

Figure 5.12: System Deployment Diagram

4.4 Design Tools

4.4.1 Draw.io

The desktop version of Daw.io was used to design all of the diagrams that model this system. Draw.io, formally called Diagrams.net is a cross-platform graph drawing software developed in HTML5 and JavaScript. Its interface can be used by developers, system designers, engineers, etc to create diagrams such as flowcharts, wireframes, UML diagrams, organizational charts, network diagrams, and a lot more. Draw.io is available free of charge as an online web app, and as an offline desktop application for Linux, macOS, and Windows. Supported storage and export formats to download include PNG, JPEG, SVG, and PDF. It also integrates with cloud services for storage including Drop box, One Drive, Google Drive, GitHub, and GitLab (Wikipedia, 2023).

4.5 Functional and Non-Functional Requirement Specifications

4.5.1 Introduction

A functional requirement is a technical feature of software that helps systems behave and operate. A functional requirement is a technical feature of software that helps systems behave and operate (Indeed, 2022).

4.5.2 ParkEase Functional Requirements

4.5.2.1 User Authentication and Authorization:

- Drivers must first create an account.
- Parking Lot manager accounts must be created by the system admin.
- All users of the system must authenticate including all the lot occupancy system which is acting as a client.
- Differentiate between various user roles (admin, parking lot manager, and driver) with corresponding access permissions.

4.5.2.2 Real-time Parking Slot Availability:

- Display real-time information on the availability of each parking lot.
- Provide an updated count of available slots for each parking lot.

4.5.2.3 Reservation System:

- Allow users to reserve parking spaces in advance.
- Specify reservation duration and provide confirmation details.

4.5.2.4 Automate Entry and Exit at the Parking Lot:

- Implement automated entry and exit systems for authorized vehicles.
- Identify vehicles by their plate number using Machine Learning in real time.
- Automatically open and close the gate for authorized vehicles to enter and leave the parking lot.

4.5.2.5 Payment Integration:

- Integrate a secure payment system for drivers to pay parking fairs.

4.5.2.6 Payment Integration:

- Develop a user-friendly mobile application for seamless interaction with the smart parking system.

4.5.2.7 Alert and Notifications:

- Send notifications to users about reservation confirmations.

4.5.2.8 Reporting and Analysis:

- Allow the system admin to generate reports on parking usage, revenue, and space occupancy.

4.5.3 ParkEase Non-Functional Requirements

Non-functional requirement specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system (Martin, 2023).

4.53.1 Performance:

- The system should respond to user queries and interactions within 2 seconds.
- The system should support a minimum of 1000 concurrent users.
- The system should support a minimum of 1000 concurrent users.

4.53.2 Reliability:

- The system should have an availability of 99.9%.
- The system should be capable of recovering from failures within 5 minutes.

4.53.3 Security:

- User authentication should be implemented using strong encryption methods.
- The system should enforce proper access control, ensuring that only authorized users can access certain functionalities.

4.53.4 Security:

- The user interface should follow industry best practices for design and navigation.
- End-users should be able to learn and use the system with minimal training.

4.53.5 Scalability:

- The system should be able to scale horizontally to accommodate a 20% increase in the number of parking spaces.

4.53.6 Maintainability:

- The system should be able to scale horizontally to accommodate a 20% increase in the number of parking spaces.

4.53.7 Interoperability:

- The system should seamlessly interact with License Plate Recognition Systems through API.
- The system should seamlessly interact with other third party APIs required for the system's functionality.

4.53.8 Environment:

- The system should be designed to minimize power consumption, especially for components running 24/7.
- It should not have a significant impact on the surrounding environment.

4.53.9 Documentation:

- Comprehensive documentation, including user manuals, administrator guides, and API documentation, should be provided.
- Documentation should be maintained and updated with each system release.

4.6 System Architecture Design

4.6.1 Introduction

The system architecture diagram is an abstract depiction of the system's component architecture. It provides a succinct description of the system's component architecture in order to assist in component-component relationships and system functioning (Taneja, 2024). It serves as a blueprint that provides a holistic view of how different elements of a system interact and work together to achieve specific objectives.

4.6.2 High Level Architecture

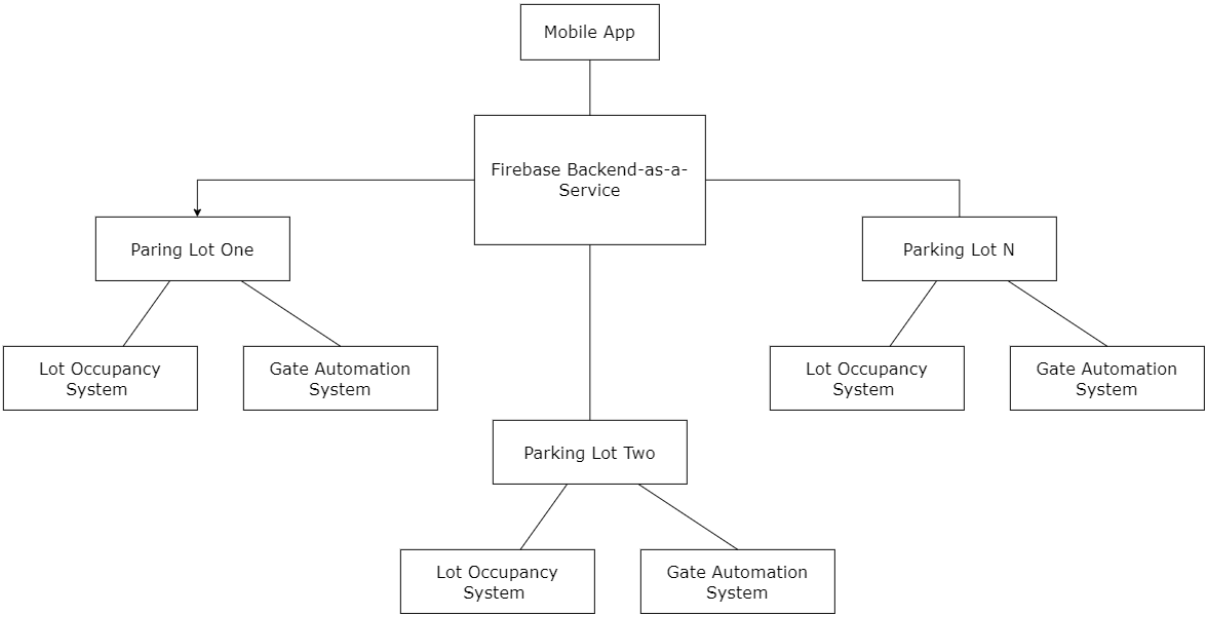

Figure 4.13*: High Level System Architecture*

4.6.3 Detailed System Architecture

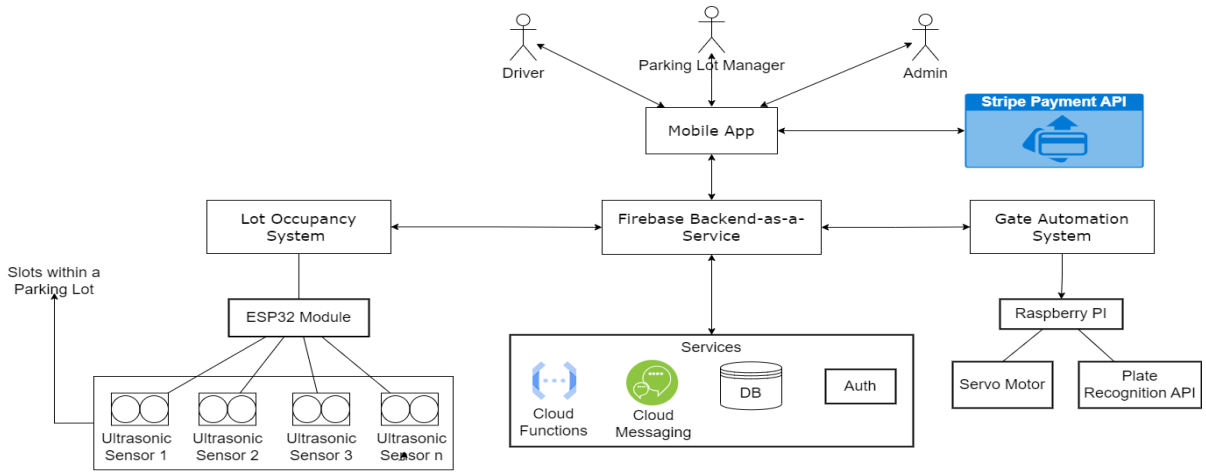

Figure 4.14: Detailed System Architecture

CHAPTER FIVE: PROPOSED SYSTEM IMPLEMENTATION

5.1 Introduction

This chapter describes the implementation and evaluation of the designed models of the ParkEase system as depicted in Chapter 4 of this report. Further, section 5.2 provides technical information about the implementation of the system including the development environment, development platforms, tools, server used. Next section 5.3 shows screenshots of significant parts of the system including pictures of the system's physical prototype. Finally, the last section, section 5.4 describes the various tests that were done on the system.

5.2 System Implementation

The system's implementation covers diverse use of technologies, frameworks and programming languages. The following are the various technologies, languages, and frameworks that were used in the implementations of this system.

- **Internet of Things (IoT):** Sensors, actuators, microcontrollers (ESP2), and Raspberry Pi are communicating over through serial communication and the internet protocol to implement this system.

- **Flutter:** To allow the system to run across multiple platforms like Android, IOS, Web, Desktop(Windows and Linux), the Flutter framework which is an open source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase.

- **Machine Learning:** Machine Learning is used to extract the plate number of every passing through the entry and exit gates. Using Machine Learning helps to increase the accuracy of the plate number while having challenges like the lighting, orientation of the images and many other factors that have huge potential to affect plate number recognition.

- **Embedded System:** The lot occupancy and the gate automation systems are simply embedded systems that work seamlessly allowing the system to run all through if it is powered.

- **Arduino C++:** The Arduino C++ was used to develop the Spot Occupancy of the system.

- **Dart:** The flutter framework uses Dart hence, the Dart programming language which is a modem Object Oriented programming language was used for developing the mobile application.

- **Python and MicroPython:** Python was used on the Raspberry Pi to program the Gate Automation system. Specifically, micro python was used to implement the embedded system component like controlling the servo motor of the gate automation system.

- **JavaScript:** JavaScript was used to develop the cloud functions which are used for the push notification and interact with the database. The cloud functions were used as APIs which are fully hosted on the Google Cloud Platform infrastructure and they were being used by the system.

- **Google Maps:** Google maps were used to display available parking lots within the specified vicinity. The direction, search, and other APIs of Google maps were also used.

- **Firebase:** Firebase is being used as the Backend-as-a-service of the entire system. The Authentication service, Firebase Real-time Database, Firestore database, Firebase Cloud Functions, and Firebase Cloud Messaging were used in the implementation of this system.

- **NodeJS:** NodeJS was used to develop a server that the webhook from Stripe API can interact with to communicate with the Firebase Cloud Functions so that when a payment is complemented, a corresponding payment collection can be created within the Firestore database.

5.2.1 Development Environment

The following Integrated Development Environment (IDE) was used to implement the system. Various IDEs were used due to the different use of programming languages and frameworks. Table 5.1 describes the hardware used in the implementation of this system.

- **Visual Studio Code:** Due to the light weight and rich support for many frameworks like Flutter, the visual studio code IDE was used to develop the mobile application of this system. It was also used to access the Raspberry pi through SSH and hence interact with the pi remotely. Finally it was used to develop and deploy the cloud functions using JavaScript.

- **Thony Python:** The Thorny Python IDE was used to develop the micro-python code. The Thorny IDE was used because it is a suitable programming environment for micro-python and it comes preinstalled on the Raspberry Pi as the Raspberry pi was used as the development board.

- **Arduino IDE:** The Arduino IDE was used for developing the embedded system using Arduino C++. It was selected as it is more comfortable and provides much more support to program embedded systems.

Table 5.1: Hardware Requirements of the system

Hardware Components	QTY	Descriptions
ESP32 Board	1	Used as the development board of the Spot Occupancy system.
Raspberry PI 3B or above	1	Used as the development board of the Gate Automation system.
Ultrasonic sensors (HC-SR04)	As per parking slot	Mounted on the ESP32 board and is used to detect whether or not a parking lot is available. Two sensors will be deployed at the entry and exit gate. One is used to trigger the camera when a vehicle is approaching white the other is used to check if the vehicle has passed the gate or not.
Servo motor	2	Used to simulate the gate. One will be placed at the entry and another one at exit.
Jumper Wires	multiple	Used for connecting all the components.
USB cameras	2	Used to capture the images of vehicles entering and leaving the parking lot. One will be placed at the entry and another one at exit.

5.3 System User Interface and Prototype Images

5.3.1 Introduction

Similar to UI/UX design, one important goal of HCI is to create technologies that are useful, usable, and helpful for the target audience. User-friendly interfaces, both in the case of physical and digital devices, are the ones that improve customer satisfaction. A critical and analytical approach to HCI research understands that the user-friendliness of any interface varies based on the goals of the users and the use settings (Ramotion, 2023).

Due to the above, the UI of this system has been tailored to providing a seamless experience for the users.

The below are the various screenshots of the mobile app along with images of the physical prototype of the system.

5.3.2 Landing & Authentication Interfaces

When the user first launches the mobile application after downloading, the landing screen is displayed. Upon taping “Get Started”, the registration screen is displayed that allows the user(driver) to create an account. The user can navigate to the login screen by scrolling down the register screen and clicking the “Already have an account? Login” text button. If the who already has an account forgets his/her password, it can reset as seen in the reset screen below.

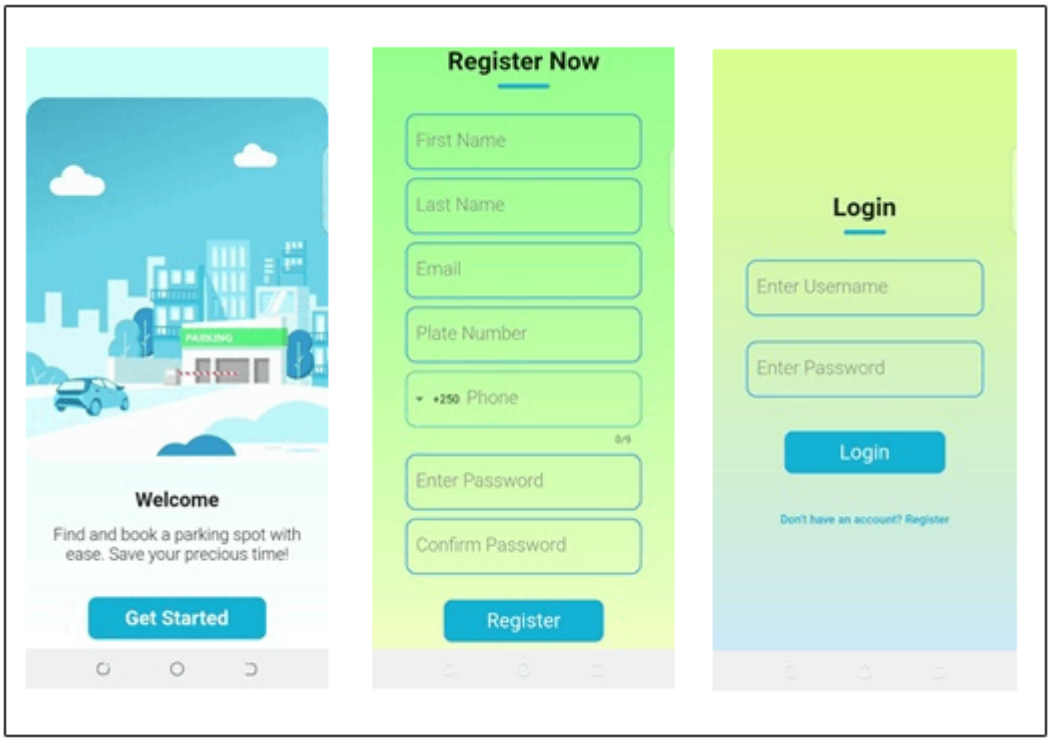

Figure 5.1: Landing & Authentication Screenshots

5.3.3 Home Interfaces

After successful authentication, the user gets redirected to the one of the below home screens depending on the user role (driver, manager, admin). Amin and parking lot managers have the same user interface for the home screen but authorization is employed to restrict access to specific user role base on its use case.

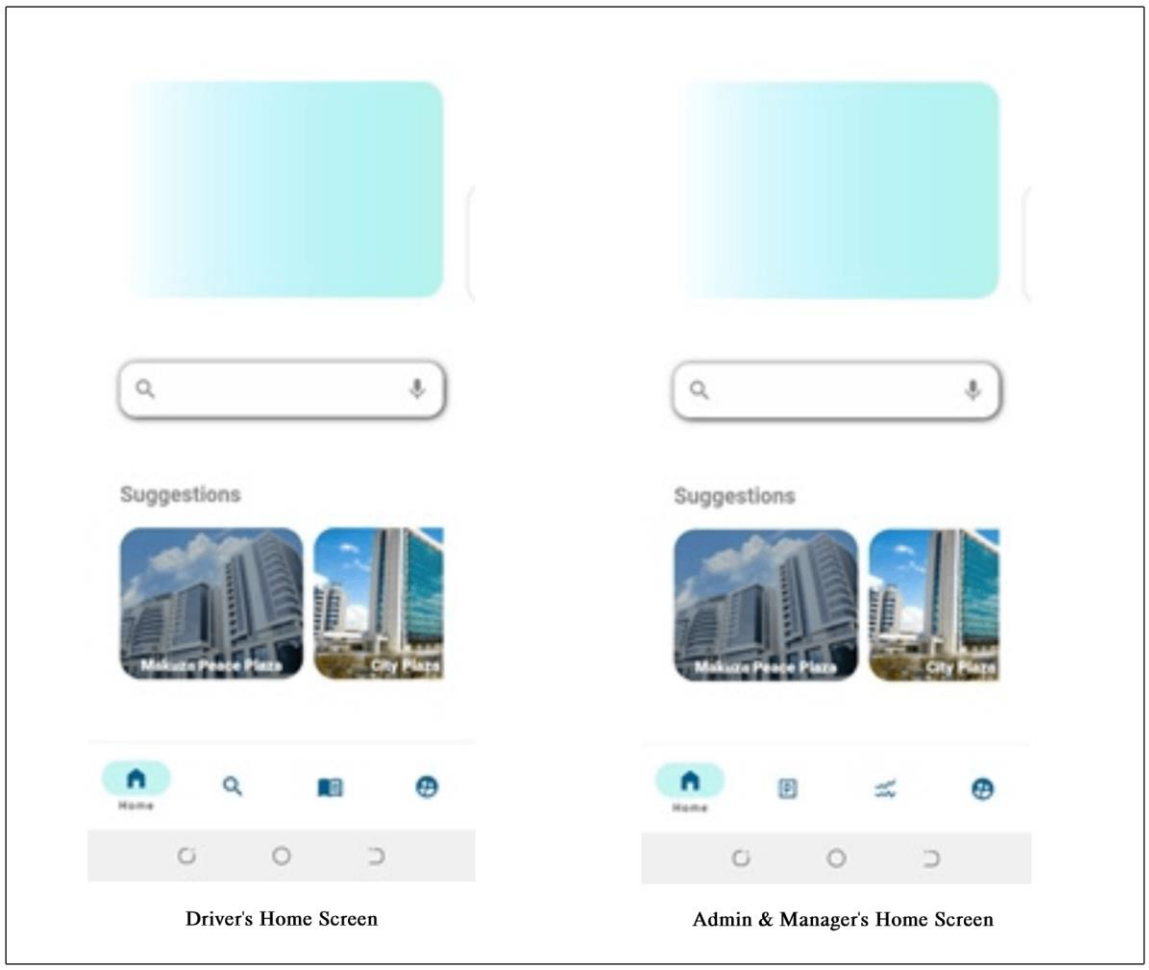

Figure 5.2*: Home Screenshots*

5.3.3 Manage Parking Lot Interfaces

The below screens displays the how the admin of the system can enroll and manage parking lots of the system. Only the admin has access to the below screens and functionalities.

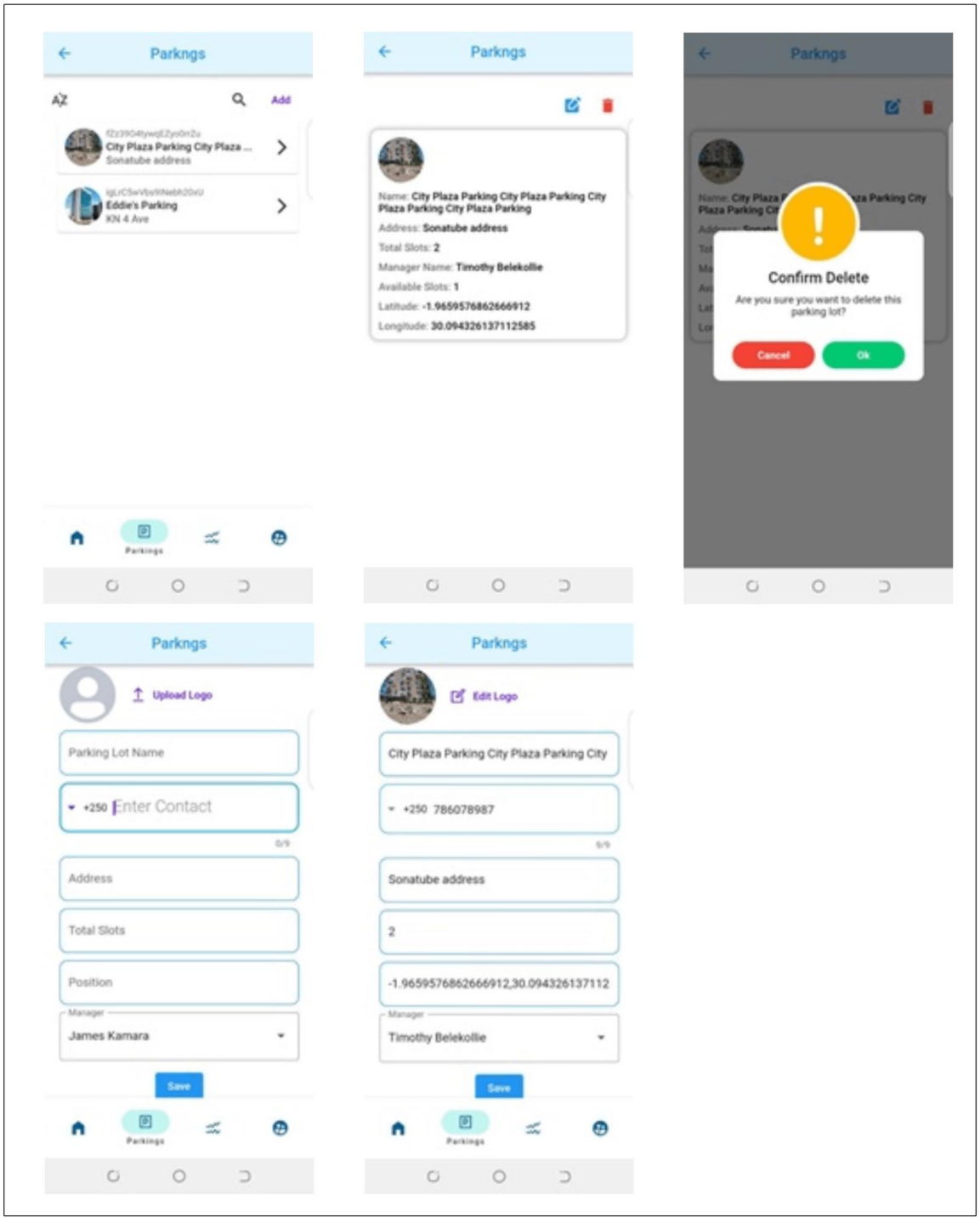

Figure 5.3: Manage Parking Lot Screenshots

5.3.4 Manage Parking Slot Interfaces

The below screens displays the how the parking lot manager of a specific parkinig lot can manage or perform CRUD (Create Read Update & Delete) operations on the parking slots.

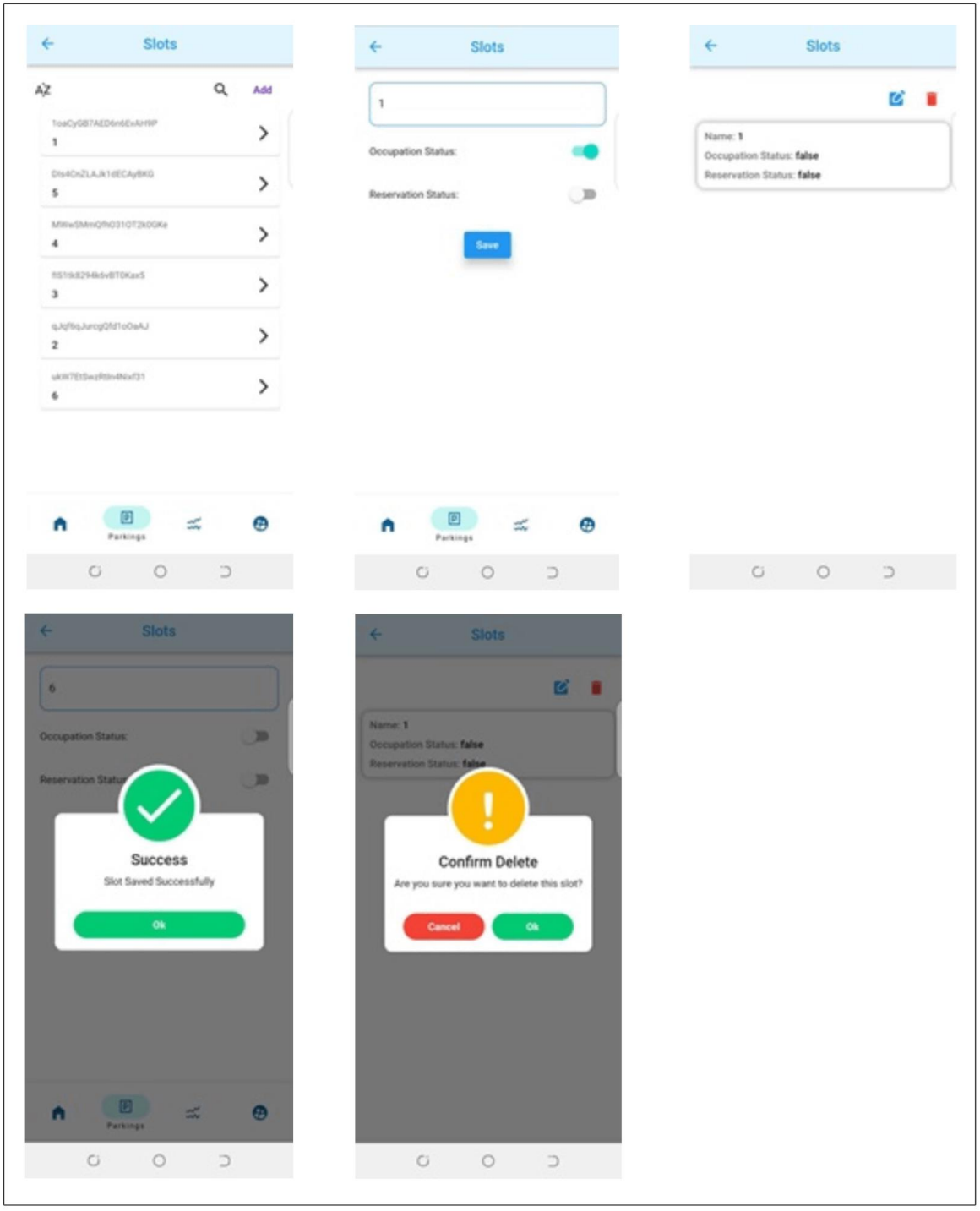

Figure 5.4: Manage Parking Slots Screenshots

5.3.5 Reserving Parking Lot Interfaces

The below screenshots display the process of a driver booking a specific parking slot within a specific parking lot. The map shows the available parking lots as a green marker within the current location of the user. The search screen shows the user searching for an area he wishes to park.

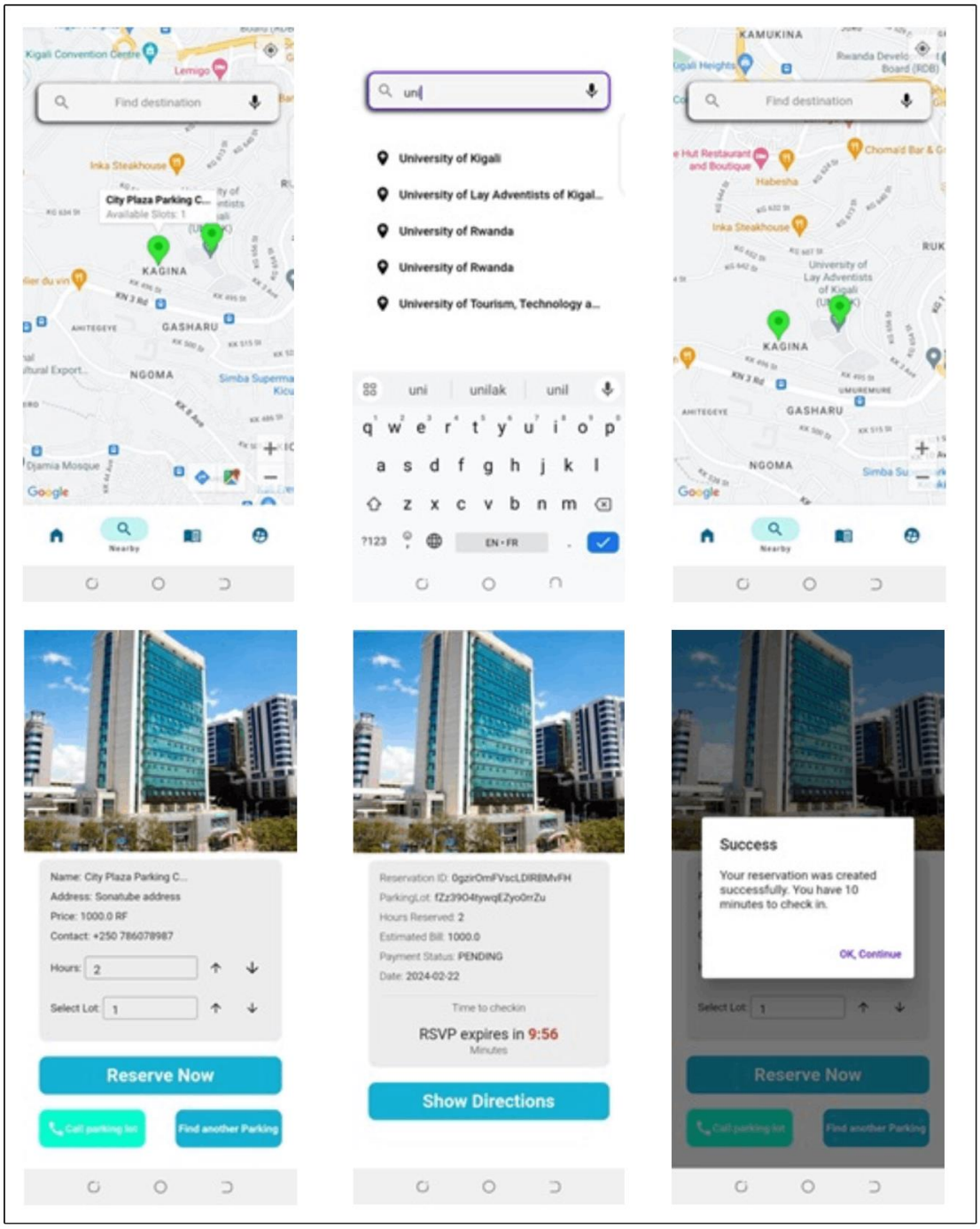

Figure 5.5: Reserving Parking Lot Screenshots

5.3.6 Booking History Interfaces

The below screenshots displays the booking history of the user. The user can navigate between pending, cancelled, checked-in history. The screenshot also shows the details screen of each booking when the user decides to see details of the booking.

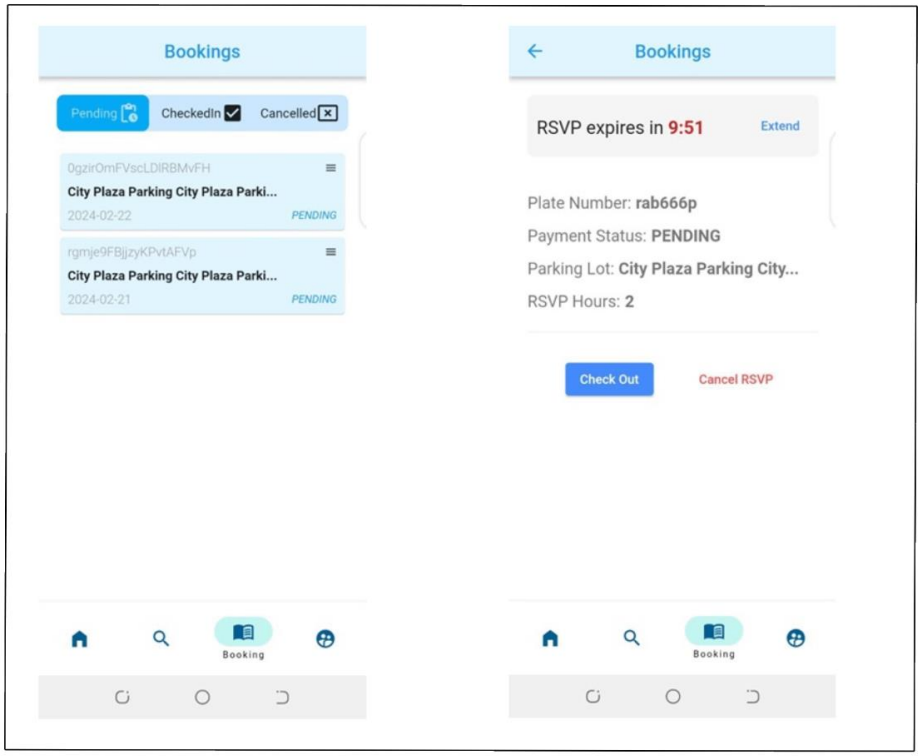

Figure 5.6: Booking History Screenshots

5.3.7 Payment Interfaces

The below screenshots displays the invoice screen, and the payment (checkout) screen. The checkout screenshot is taken in twice to capture content below that require scrolling.

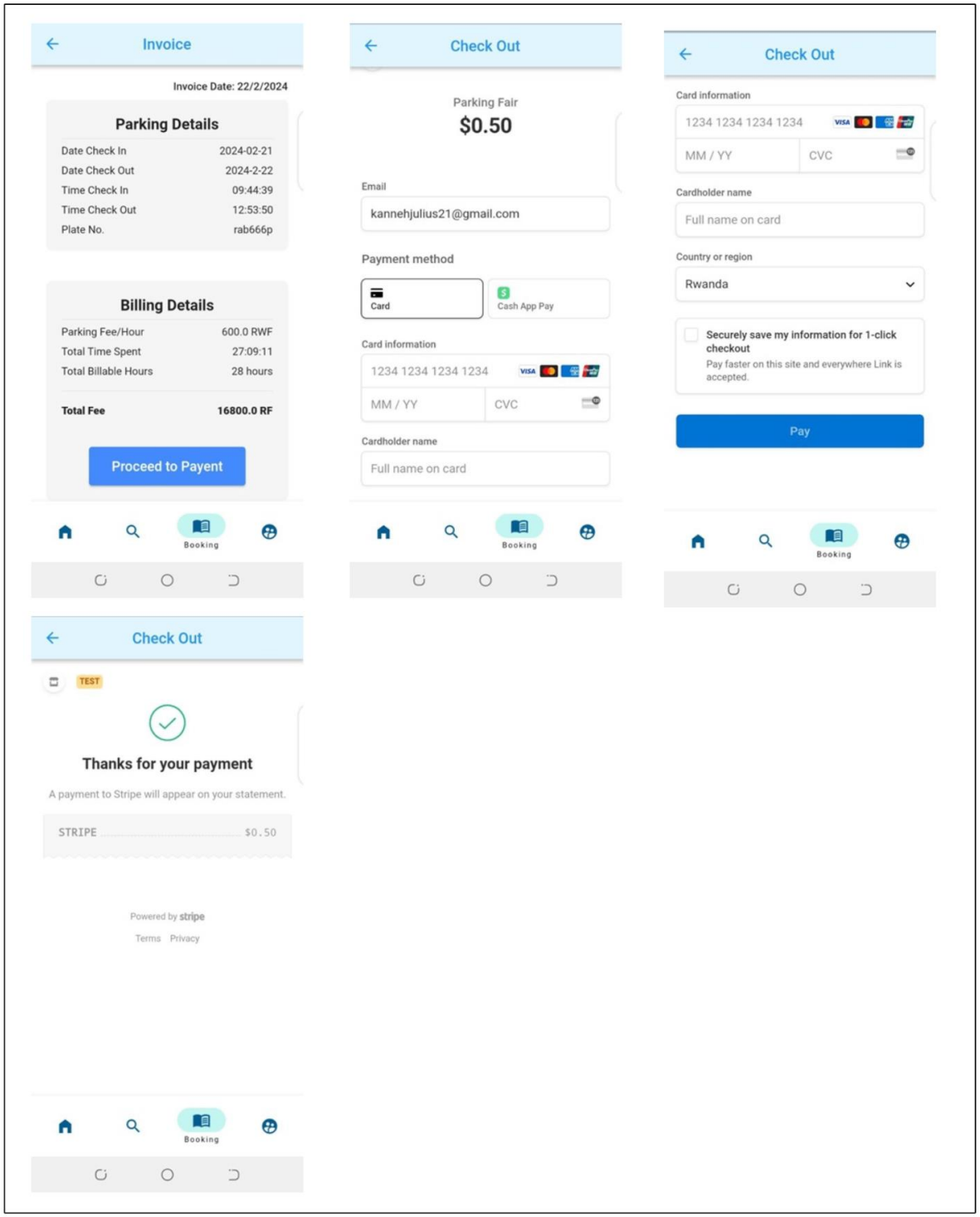

Figure 5.7: Payment Screenshots

5.3.8 Reports Interfaces

The below screenshots displays the reports for both the admin and parking lot managers
.

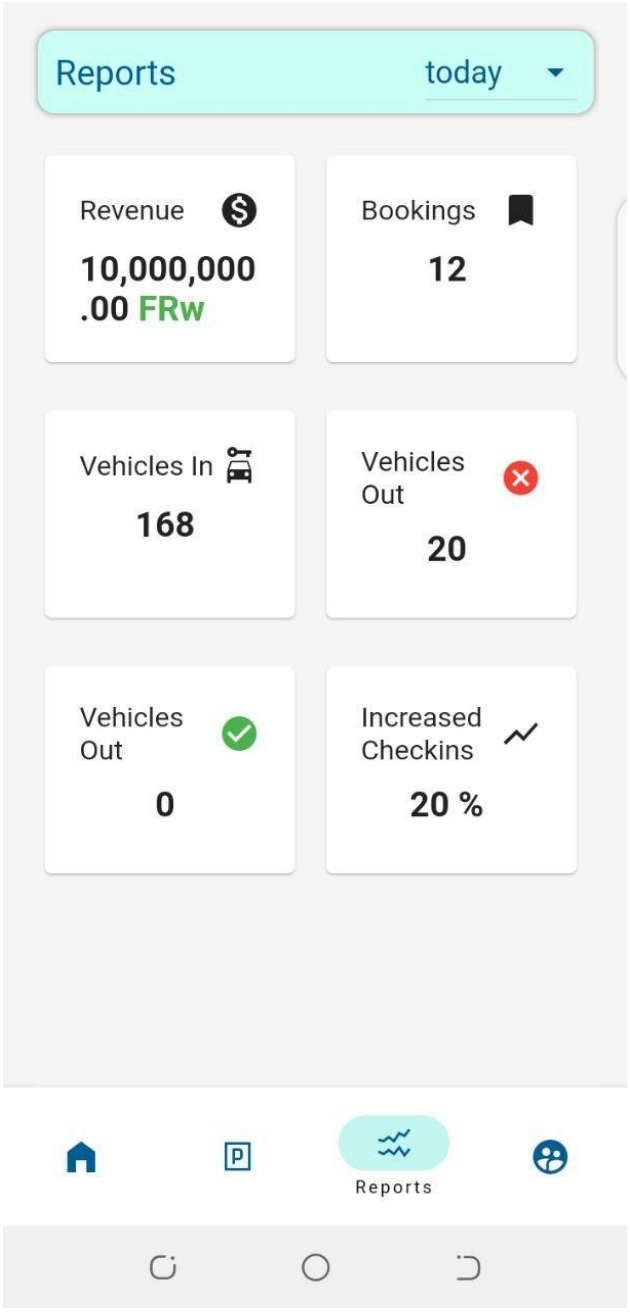

Figure 5.8: Reports Screenshots

5.3.9 User Profile Interfaces

The below screenshots display the profile screen of a logged in user irrespective of the user role (admin, driver, parking lot manager).

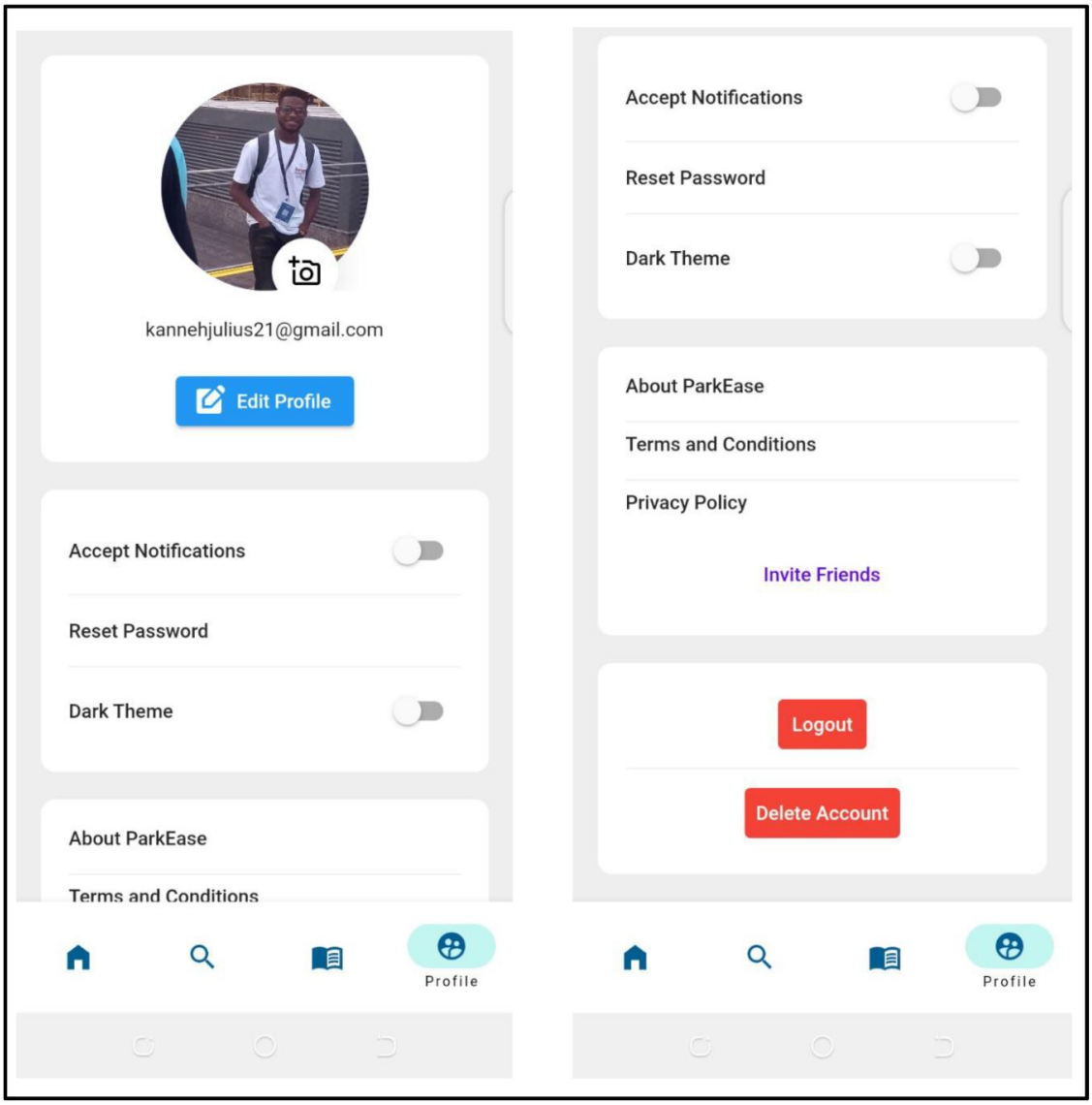

Figure 5.9: User Profile Screenshots

5.3.10 Images of the system prototype

Below are images of the system's prototype that was built to represent a model of how the system will work in real-world use case. The physical prototype helps in understanding the details of how the system works because all of the functionalities of the IoT system is represented with this prototype.

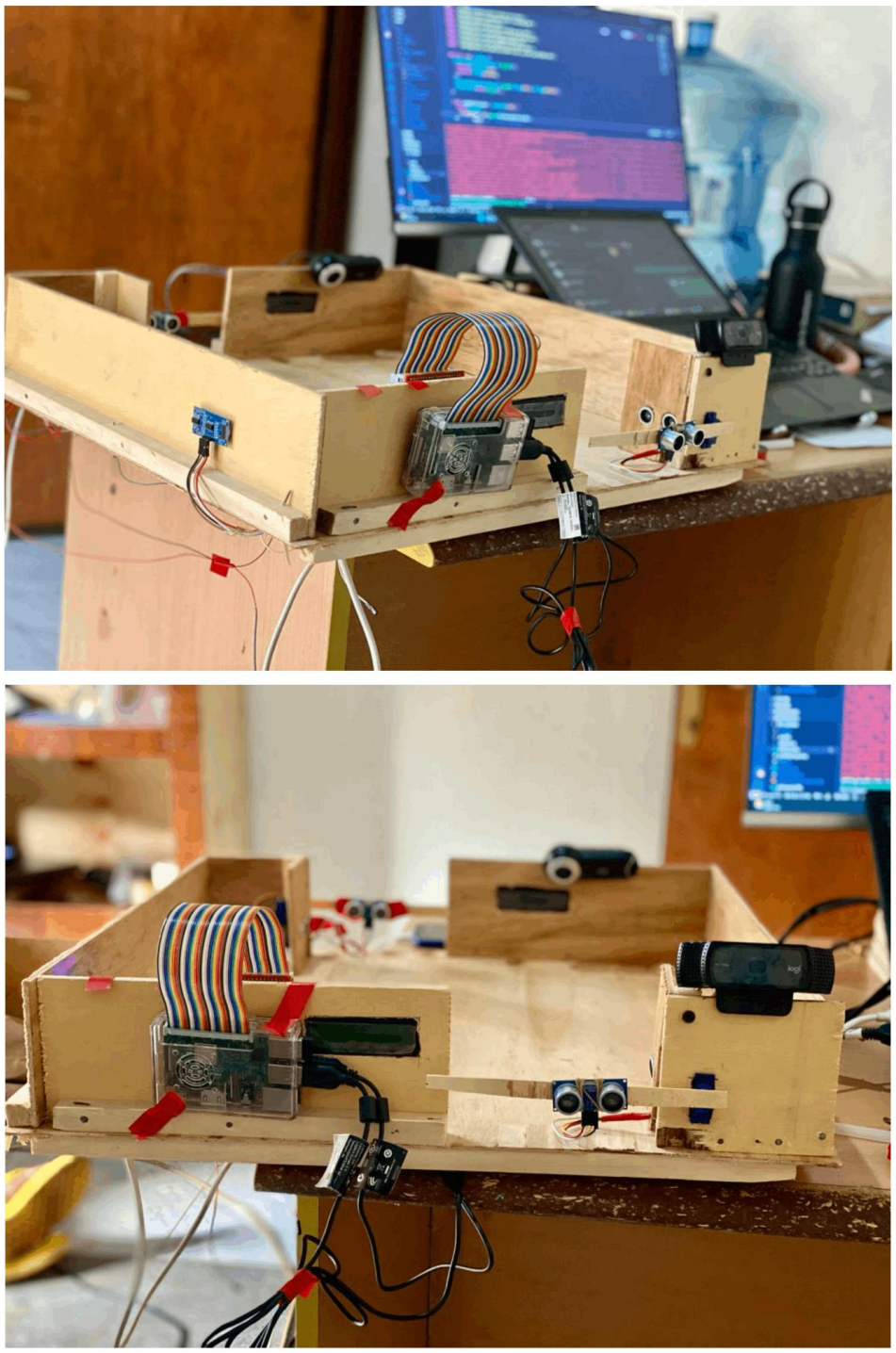

Figure 5.10: Physical Prototype Entire Setup Images

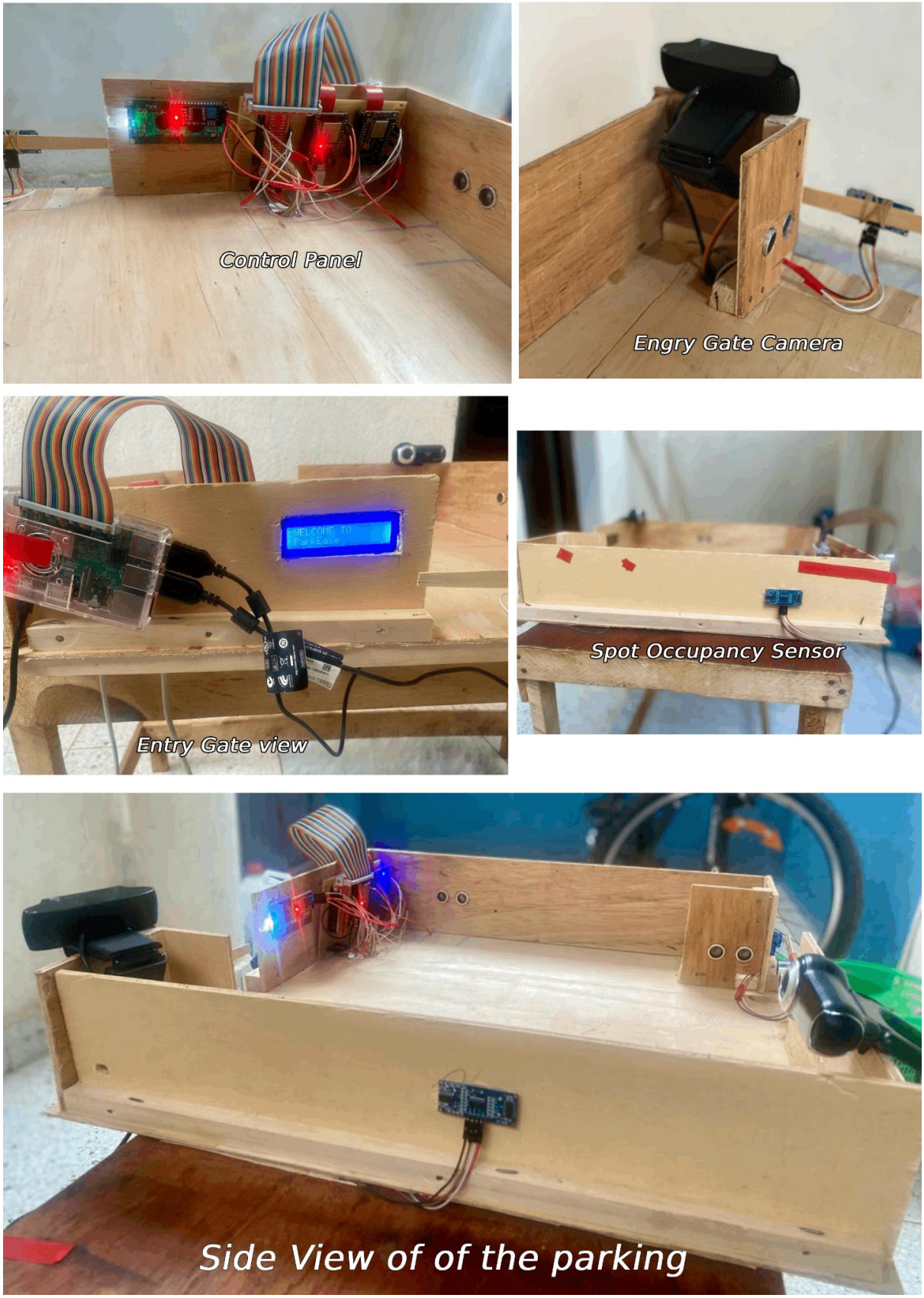

Figure 5.11: Physical Prototype Section Images

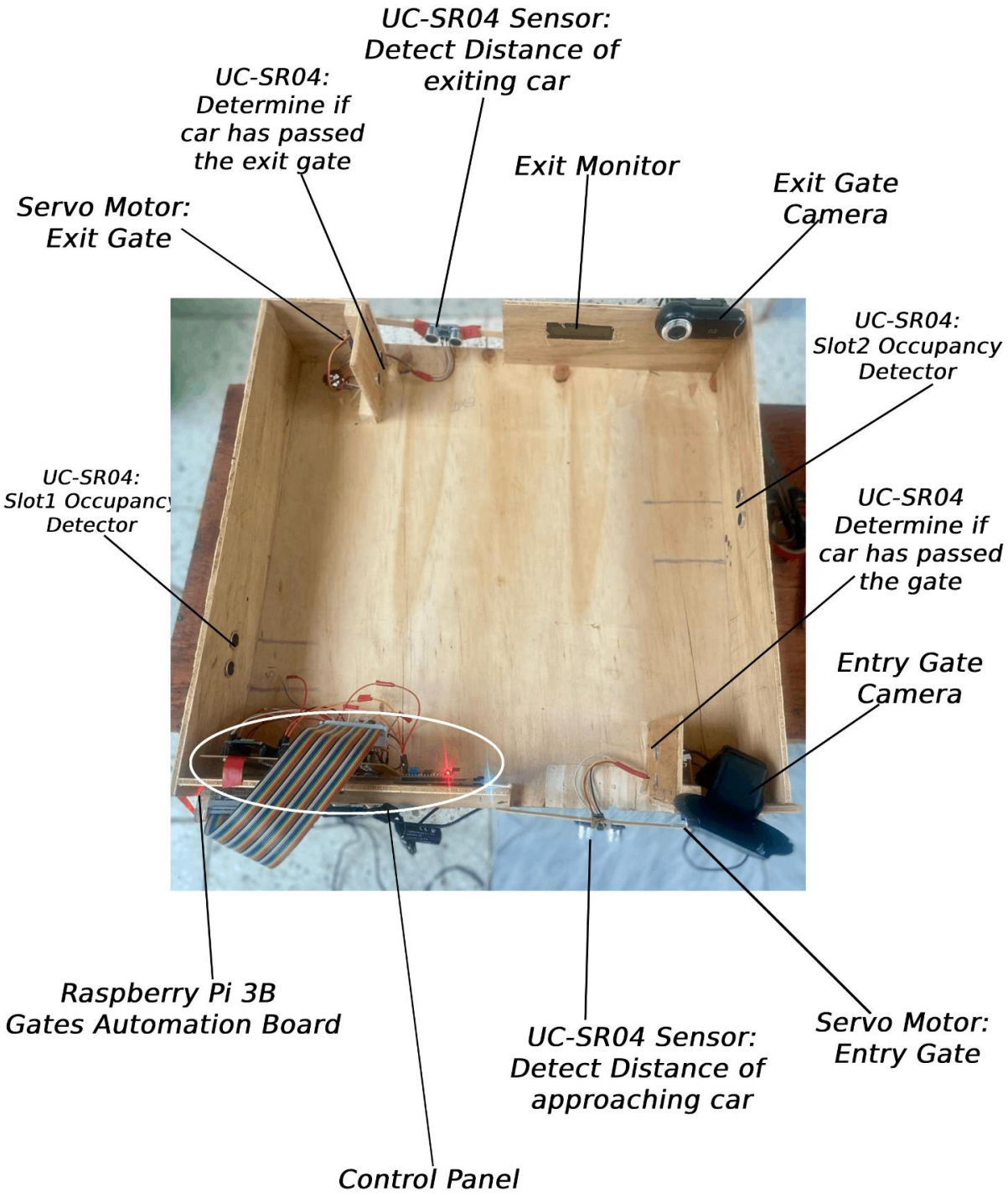

Figure 5.12: Labeled Physical Prototype Image

CHAPTER SIX: CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

In this formal technical report, the author has presented a comprehensive solution designed to significantly reduce the time drivers spent in parking their vehicles, reduce the emission of greenhouse gases, and effectively manage parking facilities. The proposed system, ParkEase, utilizes state-of-the-art technologies and innovative methodologies to tackle the challenges encountered in parking management. ParkEase provides a seamless and user-centric experience for drivers, optimizes operations for parking lot managers, and delivers valuable insights to administrators via comprehensive reporting and analytical functionalities.

The system's architectural design ensures scalability, security, and reliability, making it suitable for diverse parking scenarios. The author firmly believes that ParkEase has the potential to revolutionize parking management by enhancing efficiency, improving customer satisfaction, and generating additional revenue streams for parking operators.

The author sincerely hopes that ParkEase serves as a valuable resource for researchers, practitioners, and parking management professionals seeking to optimize their operations and provide a superior parking experience for their customers. The author encourages continued research and development in this domain to further advance parking management solutions.

6.2 Recommendation

In the process of developing ParkEase, a comprehensive parking management system, several recommendations emerged that could further enhance its capabilities and user experience:

1. **Dynamic Pricing:** Implement a dynamic pricing algorithm that adjusts parking rates based on real-time demand and historical data. This approach optimizes parking utilization and generates additional revenue during peak hours. This will be implemented based on the city in which the system is being deployed after research to proof that it is necessary.

2. **Mobile Payment Options:** Integrate with popular mobile payment methods and platforms, such as Mobile Money, Apple Pay and Google Pay, to allow users to pay for parking seamlessly using their smartphones. Additionally, offer the option to save payment information for future transactions and implement a payment wallet within the app.

3. **Subscription:** Integrate a payment subscription that allows users or companies to subscribe for x amount of months, weeks, or days as desired.

4. **Electric Vehicle (EV) Charging Integration:** Partner with reputable EV charging providers to install state-of-the-art charging stations at parking locations. Collaborate to develop a unified payment system for both parking and charging, making the process convenient for EV owners.

5. **Data Analytics and Insights:** Leverage advanced data analytics tools to analyze parking usage patterns, such as peak hours, average occupancy rates, and user preferences. Utilize this data to identify opportunities for improving parking management and optimizing revenue generation.

6. **Integration with Public Transportation:** Partner with local public transportation authorities to provide real-time information about nearby public transportation options, including bus routes, train schedules, and station locations. This integration facilitates multimodal transportation and reduces the need for personal vehicles which in turn reduces the emission of greenhouse gases

7. **Gamification and Loyalty Programs:** Introduce gamified elements, such as challenges, badges, and leaderboards, to make the parking experience more engaging and enjoyable for users. Implement a loyalty program that rewards users for frequent parking and encourages them to continue using the system. Users who uses bus stop instead of their personal vehicle will also be rewarded with a “Earth Saver / Go Green” batch. The user will also be presented with an estimate of his/her carbon footprint which will serve as conscientizer to the user to work towards reducing his/her carbon footprint. As the user carbon footprint is being reduced, the system will reward him/her.

8. **Feedback and Rating System:** Develop a user-friendly feedback and rating system that allows users to provide feedback on their parking experience. Utilize this

feedback to identify areas for improvement and make necessary adjustments to enhance customer satisfaction.

9. **Expansion to Multiple Locations:** Improve the system to be scalable to explore the potential for expanding the parking system to additional locations, taking into consideration factors such as population density, parking demand, and proximity to transportation hubs. This expansion would provide greater convenience to users and increase the overall impact of the system.

6.3 References

Ahmed, M., Hamad, M. S., Abdel-Khalik, A. S., Hamdan, E., Ahmed, S., & Elmalhy, N. A. (2023). Improved Utilization for “Smart Parking Systems” Based on Paging Technique. *IEEE Transactions on Intelligent Transportation Systems*. IEEE. 10.1109/TITS.2023.3323097

Ambler, S. W. (2010). *The Elements of UML(TM) 2.0 Style*. Cambridge University Press. 10.1017/CBO9780511817533

Ben-Joseph, E. (2020, September 3). *From Chaos to Order: A Brief Cultural History of the Parking Lot*. The MIT Press Reader. Retrieved November 21, 2023, from https://thereader.mitpress.mit.edu/brief-cultural-history-of-the-parking-lot/

Design Science Research. (2022, February 10). *Methods in Design Science Research*. Design Science Research. Retrieved November 16, 2023, from https://design-science-research.de/en/post/methods-in-dsr/

Dev. (2023, April 25). *Real-time Databases: What developers need to know*. Tinybird. Retrieved November 20, 2023, from https://dev.to/tinybirdco/real-time-databases-what-developers-need-to-know-2i9b

Flutter by Google. (n.d.). Flutter - Build apps for any screen. Retrieved November 21, 2023, from https://flutter.dev/

GeeksforGeeks. (2024, January 12). *Activity Diagrams | Unified Modeling Language (UML)*. GeeksforGeeks. Retrieved January 14, 2024, from https://www.geeksforgeeks.org/unified-modeling-language-uml-activity-diagrams/

GeeksforGeeks. (2024, January 16). *Sequence Diagrams | Unified Modeling Language (UML)*. GeeksforGeeks. Retrieved January 18, 2024, from https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/

Gichuki, P. (2023, June 27). *Here's Why Time Is Important: A Full Guide*. Traqq. Retrieved November 13, 2023, from https://traqq.com/blog/the-importance-of-time-in-our-lives/

Gliffy. (2023, May 5). *Use Case Diagrams: UML Tutorial (With Examples)*. Gliffy. Retrieved January 14, 2024, from https://www.gliffy.com/blog/use-case-diagrams

Green Vehicle Guide. (n.d.). *Vehicle emissions*. Green Vehicle Guide. Retrieved November 15, 2023, from https://www.greenvehicleguide.gov.au/pages/UnderstandingEmissions/VehicleEmissions

Hilson, G. (n.d.). *What Is Computer Vision & How Does It Work?* Verizon. Retrieved November 20, 2023, from https://www.verizon.com/business/resources/articles/s/what-is-computer-vision-technology/

History.com Editors. (2009, October 29). *Industrial Revolution*. Industrial Revolution: Definition, Inventions & Dates | HISTORY. Retrieved November 13, 2023, from https://www.history.com/topics/industrial-revolution/industrial-revolution

Hoffstaetter, S. (n.d.). *pytesseract · PyPI*. PyPI. Retrieved November 21, 2023, from https://pypi.org/project/pytesseract/

Indeed. (2022, June 24). *Functional vs. Nonfunctional Requirements (With Examples)*. Indeed. Retrieved January 23, 2024, from https://www.indeed.com/career-advice/career-development/common-functional-and-non-functional-requirements

Jain, S. (2024, January 18). *Class Diagram | Unified Modeling Language (UML)*. GeeksforGeeks. Retrieved January 20, 2024, from https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/

Kotb, A. O., Shen, Y.-C., Zhu, X., & Huang, Y. (2016, May 05). A new smart car-parking system based on dynamic resource allocation and pricing. *IEEE transactions on intelligent transportation systems*, *17*(9), 2637-2647. Google Scholar. 10.1109/TITS.2016.2531636

Martin, M. (2023, November 25). *What is Non-Functional Requirement in Software Engineering?* Guru99. Retrieved January 23, 2024, from https://www.guru99.com/non-functional-requirement-type-example.html

Math Works. (n.d.). *What Is Machine Learning? | How It Works & Tutorials*. MathWorks. Retrieved November 20, 2023, from https://www.mathworks.com/discovery/machine-learning.html

The New Times Reporter. (2016, December 11). *KVCS introduces cashless parking fee payment solution*. The New Times. Retrieved January 31, 2024, from https://www.newtimes.co.rw/article/136145/News/kvcs-introduces-cashless-parking-fee-payment-solution

OpenCV. (n.d.). *About*. OpenCV. Retrieved November 20, 2023, from https://opencv.org/about/

Paidi, V., Fleyeh, H., Håkansson, J., & Nyberg, R. G. (2018). Smart parking sensors, technologies and applications for open parking lots: A Review. *IET Intelligent Transport Systems*, *12*. IET. 10.1049/iet-its.2017.0406

ParkMe. (2023, June 16). *ParkMe*. ParkMe. Retrieved December 4, 2023, from https://www.parkme.com/kigali-rw-parking

Ramotion. (2023, April 5). *Human-computer Interaction: Significance for UX Designers*. Ramotion. Retrieved January 23, 2024, from https://www.ramotion.com/blog/human-computer-interaction-for-ux-designers/

Ritchie, H. (2020, October 6). *Cars, planes, trains: where do CO2 emissions from transport come from?* Our World in Data. Retrieved November 13, 2023, from https://ourworldindata.org/co2-emissions-from-transport

The Scientific World. (2020, October 29). *What is the Importance of Time in Human Life?* The Scientific World. Retrieved November 13, 2023, from https://www.scientificworldinfo.com/2020/10/importance-of-time-in-human-life.html

Sciforce. (2018, September 18). *Internet of Things (IoT). What is IoT? | by Sciforce | Sciforce*. Medium. Retrieved November 20, 2023, from https://medium.com/sciforce/internet-of-things-iot-what-is-iot-248a2af925bc

Tamam, I., Wang, S., & Djahel, S. (2020). An IoT-based Eco-Parking System for Smart Cities. *2020 IEEE International Smart Cities Conference (ISC2)*, 1-6. Semantic Scholar. 10.1109/ISC251055.2020.9239041

Taneja, S. (2024, January 8). *System Architecture - Detailed Explanation*. InterviewBit. Retrieved January 23, 2024, from https://www.interviewbit.com/blog/system-architecture/

UNILAK CIS Department. (n.d.). *UNILAK-CIS-STUDENTS_Final Year project_GUIDE* [Guide for CIS final year students at the University of Lay Adventists of Kiglai to use as a guide for their dissertation.].

United States Environmental Protection Agency. (2023, April 18). *Understanding Global Warming Potentials | US EPA*. Environmental Protection Agency. Retrieved November 15, 2023, from https://www.epa.gov/ghgemissions/understanding-global-warming-potentials

United States Environmental Protection Agency. (2023, October 10). *Overview of Greenhouse Gases | US EPA*. Environmental Protection Agency. Retrieved

November 15, 2023, from https://www.epa.gov/ghgemissions/overview-greenhouse-gases

Visual Paradigm. (2023, October 13). *Understanding Sequence Diagram Notation in UML*. Visual Paradigm Guides. Retrieved January 18, 2024, from https://guides.visual-paradigm.com/understanding-sequence-diagram-notation-in-uml/

Wikipedia. (n.d.). *ESP32*. Wikipedia. Retrieved November 20, 2023, from https://en.wikipedia.org/wiki/ESP32

Wikipedia. (n.d.). *Real-time database*. Wikipedia. Retrieved November 20, 2023, from https://en.wikipedia.org/wiki/Real-time_database

Wikipedia. (2023, October 3). *diagrams.net*. Wikipedia. Retrieved January 20, 2024, from https://en.wikipedia.org/wiki/Diagrams.net

A. O. Kotb, Y. -C. Shen, X. Zhu and Y. Huang, "iParker—A New Smart Car-Parking System Based on Dynamic Resource Allocation and Pricing," in IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 9, pp. 2637-2647, Sept. 2016, doi: 10.1109/TITS.2016.2531636