FOSS IMPLEMENTATION OF A CAMPUS SHUTTLE TRACKING MANAGEMENT SYSTEM.

ADANIKE OLUWASEUN TIMOTHY

(15CK02892)

A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND INFORMATION ENGINEERING, COLLEGE OF ENGINEERING IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE BACHELOR OF ENGINEERING DECREE (ELECTRICAL AND ELECTRONICS ENGINEERING)

SUPERVISOR: PROF. AAA. ATAYERO

AUGUST 2020

DECLARATION

I hereby declare that I carried out the work reported in this document in the Department of Electrical and Information Engineering, Covenant University, under the supervision of Prof. AAA. Atayero. I also solemnly declare that to the best of my knowledge; no part of this report has been submitted here or elsewhere in a previous application for the award of a degree. All sources of knowledge used have been duly acknowledged.

ADANIKE OLUWASEUN TIMOTHY
15CK02892

CERTIFICATION

This is to certify that the project titled “Foss Implementation Of A Campus Shuttle Tracking Management System” by Adanike Oluwaseun Timothy, meets the requirements and regulations governing the award of the Bachelor of Engineering, B.Eng. (Electrical and Electronics Engineering) degree of Covenant University and is approved for its contribution to knowledge and literary presentation.

Student:	Sign: ______________________	__________________
	Name: Adanike Oluwaseun Timothy	Date
Supervisor:	Sign: [illegible]	_29th August 2020_
	Name: Prof. AAA. Atayero	Date
Internal Examiner:	Sign: ______________________	__________________
	Name: ______________________	Date
Head of Department:	Sign: ______________________	__________________
	Name: Prof. Anthony U. Adoghe	Date

DEDICATION

I dedicate this project to the Almighty, the reason for my existence for his grace and guidance throughout my undergraduate studies.

I also dedicate this project to my brother Dr. Segun M. Adanike.

ACKNOWLEDGEMENT

I want to appreciate God for guidance and strength through the duration of this project and my undergraduate years in Covenant.

I sincerely appreciate my parents Mr T. T. Adanike and Mrs H.B. Adanike, for their continuous prayer and support. I also appreciate my aunt and uncle, Mr B.O. Adeyemi and Mrs E.O. Adeyemi for their continuous prayer and support.

I deeply appreciate my supervisor Prof. AAA. Atayero, the 5th substantive vice chancellor of Covenant, for his continuous support and selfless mentorship.

I also want to acknowledge my colleagues at Olive Media - Covenant University Technical Crew for their encouragement during this project.

ABSTRACT

Commuters overworld have expressed a level of frustration due to the lack of real-time information about buses and this often time affects their daily routine, hence the need to provide real-time information about this buses' location. Moreover, in campuses, this affects the attendance, psychology of students in the classrooms and the punctuality of campus workers. One of the ways of providing real-time information to commuters is through the use of a Global Positioning System (GPS) and a Map service. However, the prosperity of such system depends on the availability, strength of the internet on both the driver's and rider's side for optimum performance. The project developed a mobile application where users can gain access to real-time information on the location of shuttles on campuses for effective shuttle management using a Covenant University as a sample space. Users can request pickups within the campus and also provided messaging feature as a medium of communication between the drivers and the administrators. The system was built with Flutter tool for mobile application for cross-platform, Firebase for backend, database, authentication services, and the google map platform APIs for providing map services which include: direction routing, geolocation services. The project is built using the agile approach of software development. The agile approach gives room for additions and updates depending on the client and campus of deployment. With this, the level of frustration experienced by campus users would be on the decrease. Hence attaining of the Sustainable Development Goals (SDGs) goal 4: Quality Education, Goal 9: Industry, Innovation and Infrastructure and goal 11: Sustainable Cities and Communities

Keywords: Real-time information, campus, shuttle management, software development, mobile application, cross-platform, flutter, firebase.

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

FOSS	–	Free and Open Source Software
LOSS	–	Libre Open Source Software
IoT	–	Internet of Things
IT	–	Information Technology
ITS	–	Intelligent Transport System
S.D.G.	–	Sustainable Development Goals
API	–	Application Programmable Interface
SBU	–	Strategic Business Unit
GPS	–	Global Positioning System
P.G.	–	Postgraduate
U.G.	–	Undergraduate
IEEE	–	Institute of Electrical and Electronics Engineers
OSS	–	Opened Source Software
CSS	–	Closed Source Software
WORA	–	Written Once, Run Anywhere
CAN	–	Control Area Network
UI.	–	User Interface
VANET	–	Vehicular Ad hoc Network
SDK	–	Software Development Kit
PC	–	Personal Computer
EULA	–	End User License Agreements
HTTP	–	Hypertext Transfer Protocol
HTML	–	Hypertext Markup Language
OS	–	Operating System

VM	–	Virtual Machine
CLI	–	Command Line Interface
CICD	–	Continuous Integration, Continuous Delivery

Table of Contents

CHAPTER ONE
INTRODUCTION

1.1 Background Statement

In recent times, IoT has contributed immensely to people's daily life. Its impart ranges across all areas of human endeavour such as health care, smart grid, communication, education, transportation. According to IEEE ITS society, ITS is a system that adopts "synergistic technologies and system engineering concepts to advance and expand transportation mode of all kinds." ITS is one of the areas where IoT is immensely applied. ITS covers a wide range of applications ranging from management systems such as navigation, traffic signal control systems, "vehicle to vehicle" communication network (VANET), augmented artificial co- drivers [1]. One of the attributes of smart communities is the Intelligent Transport System (ITS). ITS is based on data collection, analysis, and using the data gathered for traffic management. ITS is one of the characterising factors of smart cities. In smart communities, ITS (Intelligent Transportation System) has been used to solve the problems underlying transportation such as long-standing queues waiting for a bus to arrive, traffic congestion due to accidents, the security of school children[2-4]. I. J. Garcia Zuazola et al. adopted the use of ITS for transportation of medicines; where real-time data could be fed to the pharmacy and stores where they need [5]. Vehicle tracking which includes predicting locations, the motion state of the vehicle and other related material such as trajectory, shape, size, the sum of cars in the same image series, has been made possible through ITS[6]. Over the four decades, mobile devices have transitioned from being an asset of luxuriousness to an asset of necessity[7]. In recent years, mobile devices such as smartphones have become very crucial to our daily lives[8, 9]. In the Ericsson mobility report, smartphone subscriptions in the world are expected to rise to about 6.1 billion this year 2020[10]. Hence, the innovation and implementations of mobile software that could benefit humanity would keep uprising to harness the relevance of mobile devices. GPS Tracking is a surveillance mechanism that specifies the location of an item or an object remotely with the help of artificial satellites [11]. In recent times, Cost-efficient GPS equipped devices (smart devices, i.e. smartphones and tablets) are used for tracking and navigation. GPS tracking has benefited area such as; real-time Vehicle tracking(users can manage their time more efficient as the system gives the live location of vehicles/buses), transport official accountability(the effectiveness and conduct of drivers can be more accurately noted and management would know the locations of the vehicles at every instantaneous time), vehicular management/maintenance(the due service timing for the car can be accurately observed by recording route and distance travelled).

The expected result of this project is to develop a platform where higher education students can track shuttle services and order shuttle(cars) from almost any way on their various campuses. This platform would help plan their time more efficiently as the position of the bus could be known in line with a scheduled time. The proposed system is line with Sustainable Development Goals (SDGs) 9 – Industry, Innovation and Infrastructure and 11 – Sustainable Cities and Communities.

A survey was carried out among students of top universities in Nigeria. From the study, about 250 of every 300 students suffer from the frustration of long-waiting shuttle queue(Figure 1). Out of these 300, about 260 of them expressed the importance of knowing where the shuttle is in real-time as it could help them to plan their time.

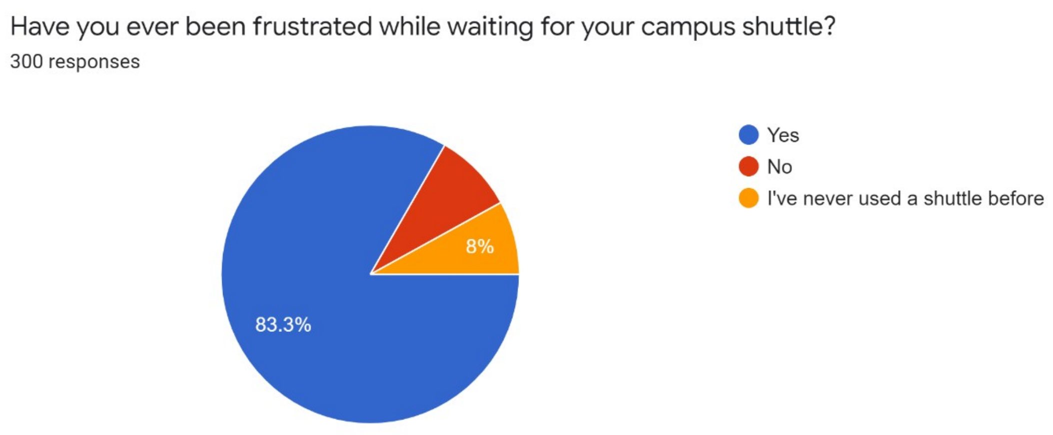


Figure 1.1: Reaction chart while waiting for a shuttle

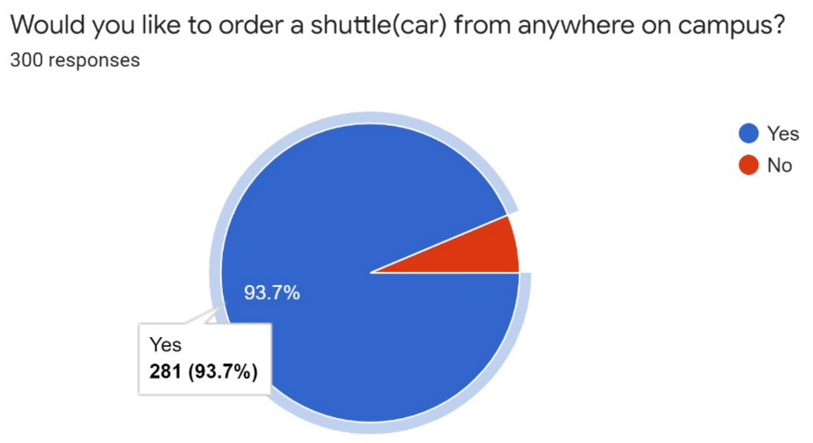


Figure 1.2: Pick Up service chat

Using Covenant University as a sample space, Covenant is located in Ota, Ogun state between the coordinates (6.2°N, 3.0°E) and (7.8°N, 5.0°E) [12]. The survey showed that about 40% of Covenant University students (undergraduate and postgraduate) use the shuttle system provided by the school for transportation within the university campus. About 83% (Figure 1.1) of the student population suffers from the frustration of long-waiting shuttle queue. About 94% of the students complained about the need to be able to request a pickup service on the campus (Figure 1.2).

ITS is one of the characteristics of smart cities hence the need to adopt this system on our campuses. There is a need to harness the efficiency of ITS on our campus transport system. Moreover, from the survey among the students, added to their frustration is their inability to track these shuttles as they spend time waiting for these shuttles. Campus transport problem has a ripple effect on how productive students are, as it affects their time planning, punctuality and stress level in the classroom. In this project, a system is proposed that could enable students to track these bus locations and plan themselves following the real-time information provided on the mobile app and also a system that could help assist the postgraduate student in ordering car-shuttle from anywhere on the campus.

1.2 Problem Statement

Using Covenant as a case study, research was carried out to get feedback from the students about the present state of the shuttle system on the campus. The research showed that more than four-fifth of every 164 Covenant students complain about being frustrated while waiting for a shuttle, about 30 per cent habitually use the shuttle services. I planned on building an application that would enable students to track shuttle location real-time, thereby generating an automated schedule of the shuttle based on the data derived from google maps API. The application would allow students to order shuttle – car – from anywhere on the campus. It also embodies an interface where transport managers could communicate with the shuttle in the case of a particular service. The proposed system required the collaboration of the SBU – Strategic Business Unit (transport department) – of the University for it to be implemented. The SBU would assist in ensuring that their drivers have smartphones that could be used as the GPS tracker in tracking the bus location. The SBU (transport) would also assist with providing information that would help in linking the drivers to the type of shuttle they drive – (car, wagon, buses) because different shuttle would have various colour indicators on the map. I would be using flutter a (FOSS)Free and Open-Source Software created by Google for creating android and iOS applications.

1.3 Justification of the Project

Influenced by the exponential explosion of GPS enabled smart devices, a large pool of data has been dung on transportation [1, 13, 14]. The density of traffic can be determined to a very great through the presence of the human traffic present to time. Smartphones and mobile devices have contributed immensely to the acquisition of transport data[15]. With the aid of the embedded GPS, accelerometer and gyroscope sensors in smartphones information such as the rotation vector, speed, and acceleration, the transportation mode of a user could also be known using machine learning model [16].

A survey was carried amidst random students of famous Nigerian universities, below is a chart that shows the frequency at which students use their campus shuttle, more than 85 per cent of the students indicated they are of the shuttle system on their various campuses. Some commented on how the shuttle destabilises their daily schedules as they are prone to coming late to class but the delay they expressed at the bus-stops.

1.4 Aim

To create a real-time tracking system and an online pickup request system. Also, to explore the use of Flutter – a Free and Open Source Software - and Google Maps APIs to provide innovative solutions for smart campus shuttle services.

1.5 Objectives

The objective of this project is built to:

- build a platform where campus users can track the shuttle through the implementation of Google Maps API with flutter tools for mobile application.
- build a shuttle(car) pickup request application using flutter development tools for creating a mobile app.
- create a messaging platform between the drivers and the administrators.

1.6 Methodology

With the use of Google Maps Platform SDKs and APIs, the mobile application designed with Flutter tools would map out the geographical location of the campuses with two different ends for the users. Namely;

i. An interface where the postgraduate students could order shuttle(cabs) from anywhere on the campus and both undergraduate and postgraduate students could track the location of shuttles on campus, drivers;

ii. The system would enable students to track these bus locations and plan themselves by the real-time information provided on their mobile devices. The proposed system would permit the users(students) to track the drivers' phones while they on duty. This system would help students in planning their time as at when they need to be at the bus stops and on the other hand help in increasing the accountability of the transport system.

Moreover, the mobile application would provide medium communication between the drivers and the administrators. The administrators would be allowed to send messages to the drivers in case of a special pickup or meeting.

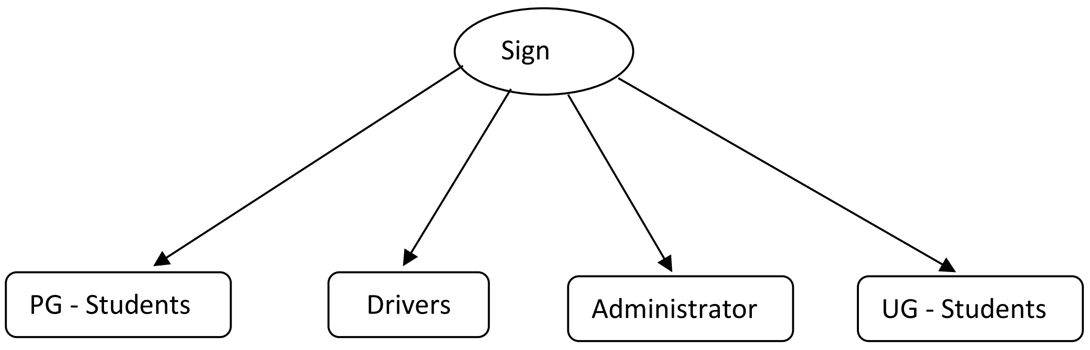


Figure 1.3: Access level of users to the Sign Up pages

The diagram (Figure 1.3) above indicates the level of access that would be given to different users, namely: PG – students, U.G. – students, shuttle drivers, transport administrator in accessing the signup page. All users of the mobile application would be required to submit the information that would be used in granting them access to the other UIs upon logging into this application. This information would be stored in firebase.

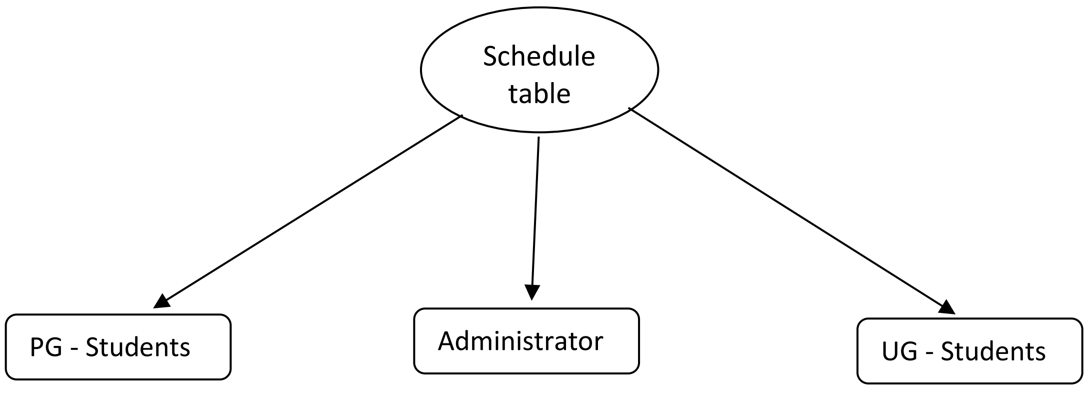


Figure 1.4: Access level of users to the Estimation page

The diagram (Figure 1.4) above indicates the level of access that would be given to different users, namely: PG – students, UG – students and transport administrator in accessing the scheduling table/ Estimation table. Based on the information submitted in Figure 1.4. only the users listed above would be granted access to the UI.

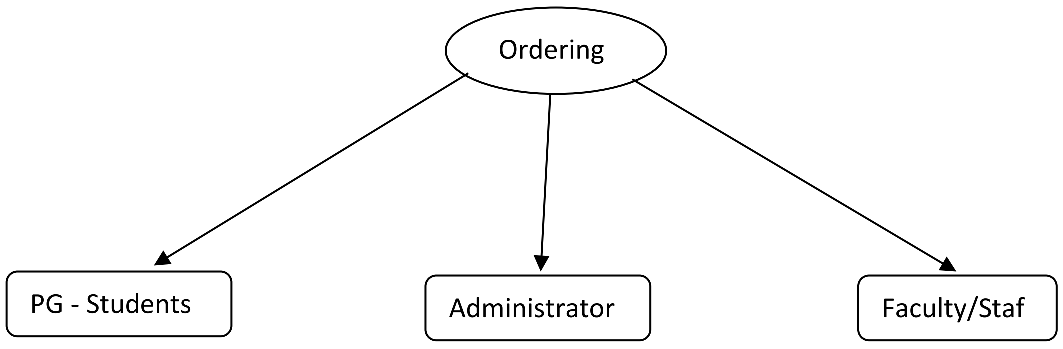


Figure 1.5: Access level of users to the Ordering UI

The diagram (Figure 1.5) above indicates the level of access to the ordering page, namely: PG – students, faculty/staff and administrators. The screen would enable them to request pickups from any location on Campus.

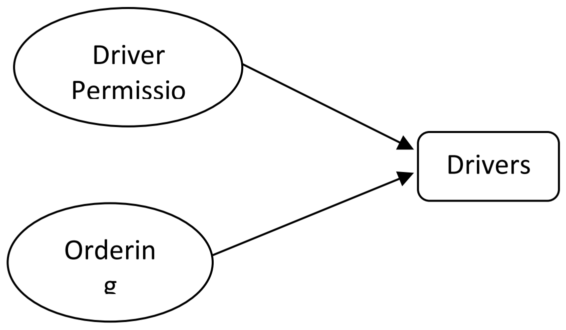


Figure 1.6: Access level of users to the Ordering and Driver Permission UI

The diagram (Figure 1.6) above indicates the level of access to the driver permission page and ordering receipt UI that would be given to the drivers only. The consent of the drivers are gotten to use their mobile devices in tracking shuttle locations in the driver permission UI, and the drivers can also see and accept placed orders on the ordering receive screen.

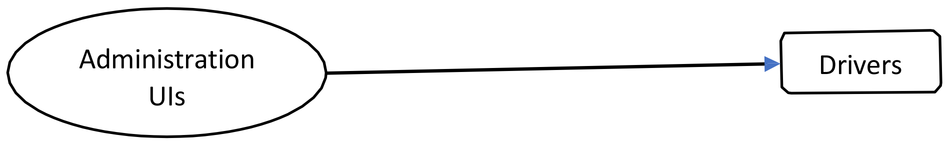


Figure 1.7: Access level of users to the Administration UIs

Figure 1.7. indicates the level of access to the administration UIs. Only the administrators are permitted to access the administration UIs where they can send messages to drivers and update driver profiles.

In the course of this project, the Agile approach of software development would be used. The agile approach of software development is an iterative approach where the software developed is delivered to the clients at incremental[17, 18]. Because of its single-phase, flow, the agile approach is used to give room for changes in the requirement as requested by the client/sponsor. The reason for selecting this approach of software development is that unlike plan-driven approaches, the functionality of these increments is decided during the development phase and not planned. Moreover, the agile approach makes the development process flexible as it keeps in term with trends and technologies. It also helps in minimising the overall risk of the project because it follows step by step procedures and as a step ends, the development is measured to finish to project moved to the next step. The proposed mobile application would connect to the GPS application inside the user's mobile phone to find the location of the user (rider or driver). This location would be reflected on google maps and the functionality to show the application's data on the map. The functionality (such as speed, heading, rotation, time) provided by the GPS would be rooted in the application for tracking purposes. Since this app would work on data, the data would be stored on a database (firebase for this project).

1.7 Scope of the study

The scope of this project includes:

- Exploring the use of flutter in building the user interface and backend of a tracking mobile application.
- To explore the use of Google Maps Platform in getting the real-time location of buses using the drivers' smartphone for tracking the shuttle's location.

1.8 Limitations of the study

ITS is a branch of IoT (Internet of Things) that relies on Internet availability for its implementation. The project work depends on three major factors, namely; the availability of internet connection, the possession of smart devices by the users and the coverage of the GPS on the campuses.

The strength of an internet connection is a relevant requirement for the use of the mobile application. Availability of internet connection either in the form of a wired, wireless or cellular network is required. The network is needed to create a communication line amongst the application on the user end to the database then to the location of the driver using GPS, and it also needed by the driver to inform the users about the shuttle's current location with the help of the smartphone's embedded GPS. Moreover, smartphones run a shorter battery performance time when the embedded GPS is continually turned on[19].

The smartphones are the interface between users (drivers and students), and also it houses the embedded GPS that is needed to track the shuttle on the driver's side and to give an estimated time on the side of the users. Moreover, since the mobile application depends on Google Maps, the app is limited with the amount of data.

As such, the coordinates of the bus stops and landmark building would be inputted to the database.

The system interface controls the mobile app to the GPS navigation system inside the mobile phone. Since there are various systems and numerous GPS producers, the interface will undoubtedly not be the equivalent for all of them. Additionally, there might be a distinction between what navigation features each of them gives.

1.9 Report organisation

This report is organised into five chapters. Below are a summary and content of each chapter:

- Chapter 1 delivers a summary of the project. It consists of the different sub- sections namely; background study, problem statement, the significance of the project, aims, objectives, methodology, the scope of the study, limitations of the study, and the report organisation.
- Chapter 2 provides reviews on related literature works, academic and technical, previously carried out on vehicular tracking and the use of GPS embedded in smartphones.
- Chapter 3 provides information on the system design and methodology used to fulfil the aim and achieve the objectives of the project. Details about the code third party API and online database management are included in this chapter.
- Chapter 4 entails the results of the implementation and testing for this project. It contains discussions on the results of the project.
- Chapter 5 includes a recap of the entire project work, achievements, recommendations, challenges faced during this work.

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

Access to real-time transportation has proven to have contributed positively to the overall structure and management of transportation systems. A survey was taken among students of Nigerian campuses, and the result shows that the structure of the transport system has an indirect impact on the performance of students in the classrooms.

According to Ojo et al., one of the reasons for Campus shuttle is the location of campus users' houses hostels, halls of residence or and lecture rooms as the case is for the University campuses[20]. Zhang et al. analysed how travellers to a recently utilised real-time bus arrival information system for the University of Maryland(College Park) 's shuttle service utilising two fixed-effects models and five random-effects-ordered probit models. It was discovered that the utilisation of real-time information essentially expanded the rider's sentiments of security about riding the bus in the night and their time planning[21]. Moreover, Vanderschuren et al. listed some effects of the accessibility of real-time information such as; reduced waiting time, reduced uncertainty, adjusted travel behaviour, higher customer satisfaction and an improved system image[22]. Hence the importance of providing real-time shuttle information on campuses can not be overemphasised.

2.2 Software

According to IEEE, software refers to all or part of the programs, procedures, rules, and associated documentation of an information processing system and it is likewise computer programs, computer programs, related documentation and information identifying with the operation of a computer system[23]. Software is a collection of instructions that users use to work computers and execute specific instructions. They allow users to communicate with a PC, its hardware, or perform specific assignments. Software is a conventional term used to refer to

applications, scripts and projects that run on a computer. Computer software is broadly classified into two major categories:

- Application software: they are published for users to perform specific tasks and handle certain kinds of problems that can not be handled by the software that controls the computer itself[23]. i.e. they are designed distinctly for the user's specific need. It can be subdivided into general-purpose (i.e. they are used for lots of different tasks, e.g. word processors) and specific purpose application(i.e. they perform a single specific task, e.g. calculator, web browsers).
- System software: this refers to software that is intended to simplify the operation and maintenance of a computer system and the computer's associated programs[23]. They are primarily used to control the system's hardware, thereby serving as the interface between the hardware and the user's application(application program). System software can further are grouped into; Operating system, device drivers, firmware, language translators, utilities.

2.2.1 **FOSS**

Free and Open Source Software (FOSS) also known as Libre Open Source Software (LOSS) refers to software dispersed under a license that allows redeployment of the source code, the alteration, and usage with almost no restriction[24, 25]. Free and Open-source software is a term used for software that is free and open-sourced. Open source describes the concept of distribution of already finished products and the documents involved in making these products[26]. In software engineering, OSS (Open Sourced Software) is a software that has the source code accessible to the public for use, redistribution, amendment or modification to suit personal preference or for redistribution without restriction, i.e., it is license-free. OSS gives room for flexibility, reliability, stability, security, transparency with amounts to a better quality

of software because this software can be adjusted to suit a different purpose. Open-source software is not a cost centred functionality software. However, some features and technical support may cost. According to Atayero et al., for a software to be classified as Open Source, it must possess these qualities:

- No intentional secrets
- All patents must be under a royalty-free license, and
- Availability[26]

Application of OSS ranges from communications, automation, web development, application development.

The source code is the root part of the project that programmers can modify to add/modify/edit the functionality of the software. It also refers to a list of human-readable and human-comprehensible instructions written in a syntax that, when run, turns into machine code "(object code)" - which is the computer-readable instructions consisting of "0s" and "1s". It is the part that controls the interface the computer users use. The accessibility to this source code is what determines the type of software. Another advantage of FOSS is that it gives room for upcoming training programmers. Examples of Free and Open Source Software includes Linux kernel[27], Samba, MySQL, BIND, Apache, Flutter

2.2.2 Proprietary/Closed-source software

Proprietary or Closed Source software is the software where the software publisher withholds intellectual property rights. The difference between the Open Sourced and Proprietary Software is that in Open Sourced software, the intellectual property right is not withheld while in Proprietary software, it is withheld. A company or an individual usually owns it, and they place some level of restriction on the software's use, and the source code is kept from the public.

A sub-type of Proprietary software is the freeware. Freeware is free to use, but not open-source. Examples include Adobe's Acrobat Reader, Internet Explorer browser, and Google Chrome. Freeware is still copyrighted although it is free in its usage – which makes a proprietary software – and may contain license agreement, which limits its usage and distribution among users must consent to gain a level of access to the software. The software publishers list the license agreement in the End User License Agreements (EULAs), which are usually lengthy. Shareware refers to software that is accessible to the users on a trial basis, and for continual usage, the user would pay for it. Compared to freeware, shareware regularly has limited functionality, and the functionality may just be utilised temporarily before payment instalment is being required and registration from the user. When the user pays for a shareware program, the program is completely functional, and the time limit restriction is removed.

Significant propriety producers include Adobe, Borland, I.B.M., and Microsoft. The significant differences between opened and closed sourced software include; source code availability and accessibility, the price policy, security, quality of support and the software's usability.

Other classification of software includes:

- Liteware: is a type of shareware with certain capacities restricted until the client fully buy the software.

- Public domain software: this refers to software that can be downloaded for free without restrictions.

2.2.3 Comparison between OSS and CSS

S/N	Open Sourced Software	Closed Sourced Software
1.	It does not have an expense related to the core functionality, even though it may have costs for additional features, support, or included functionality [24].	It is a paid software and costs varies on the complexity of the software. Although some software publishers give trials to convince the users to
2.	Open-source software publishers provide an ability to change the source code without any restrictions.	Only the software publishers have freedom with the source code. i.e. the users can not view the sourced code nor change it, and this
3.	Due to its availability, the source code is prone to hackers.	CSS source code is not readily accessible; therefore, it is less prone
4.	Due to its availability, the source code can be viewed, shared and modified by any programming community, i.e. anyone can fix, modify[28], upgrade and test the source code. i.e. it is flexible	CSS can be fixed only by the software publisher. In case of a bug, the user would have to contact the publisher for rectification.

5.	OSS publishers write documentation for programmers; hence a level of programming technicality is needed to use the software.	CSS's documentation is usually well-written and contains detailed instructions that required little or no level of programming technicality to

The table shows the comparison between open sourced and closed sourced software

Table 2.1: Comparison between OSS and CSS

2.3. Mobile Application

This refers to application software that works on mobile devices like smartphones and tablets. Mobile application development is a cross-disciplinary field that integrates zones as software development, web programming, human-machine interaction, IT-security, artificial intelligence, and machine learning.[29].

The amount of smartphones' users is increasing exponentially, resulting from the availability of numerous mobile applications that serve the users in various aspect of their daily life[30]. For this reason, apps have been developed, and more mobile application is developed in fields such as education[31], tourism, environment, tourism[32, 33], environment, governmental services and entertainment[30]. Software publishers produce mobile applications on mobile device platforms such as iOS, Android, and windows phone.

Types of Mobile Application

There are three types of mobile applications[7, 8], namely:

1. Native apps
2. Web apps
3. Hybrid apps:

2.3.1 Native applications

Publishers designed for specific platforms[7], i.e. they generally developed by platform- specific languages, e.g. NET for windows phone, C++ for windows and blackberry Java/Kotlin for Android, Objective-C or Swift for iOS. The users of these applications required these applications from the platform stores, e.g. Apple's AppStore, Google Play for android[34]. They gain direct access to the hardware and local resources, thereby enabling them to attain optimal performance and satisfy user experiences by invoking the application programming

interfaces (APIs) provided by the platform. To develop native applications, the bases for building the software differs. For example, to build an android application the developer must understand; the intent(components that function to accept and respond to deliver reports), the service(the components that run at the background) and the content provider(components that make a set of data applications that is reused for other applications)[35]. They are also referred to as Platform Application[7]. Examples include Camera+ for iOS devices and KeePassDroid for Android devices; other examples include Instagram, skype.

Benefits of Native applications include:

i. Device specifications: because they are platform/device specs, they are fully integrated into the specific platform devices, i.e. they tend to provide a recognisable home look and feel for the application platform. They also allow developers to increase full access to a wide range of device features.
ii. Security: Native applications are more secured. Developing native application engenders a great way to guarantee user's data protection because, for example, web applications rely on different browsers and their underlying technologies.

iii. Performance: Native applications are known to be faster and more responsive compared to Web applications.When the performance of native applications and Web apps were seen from the viewpoint of Web services, as far as HTTP(s) requests, traffic, round-trip time, and energy utilisation native apps performed better than Web apps.[34].

Limitations of native application

i. Portability and flexibility: Native applications are firmly coupled with their specific platforms; hence, resulting in poor portability[8], i.e. it has one codebase. Android applications will not run on iOS devices and vice versa, so the application publisher would have to write syntax in different codebases for each platform they choose to build in.
ii. Cost: Software publishers have to publish different versions for diverse platforms, and these resolves to high cost in development, deployment and maintenance of the applications[8]. Native app development requires more time of development with cumulates to a developmental cost. With different codes for different platforms, developers take extra time and money to develop a native application.

2.3.2 Web applications

These are applications that are accessed through a web browser of a mobile device over a network[7], e.g. Internet. The main difference between web apps and websites is that additional functionality is added coupled with its' interactiveness. Web applications are also developed to work within browser-based runtimes such as Gecko and WebKit[34]. They are executed with platform-specific web technologies such as HTML, JS (JavaScript), CSS(Cascading Style Sheets). In Web apps, web code can access limited local resources as a result of limitations in the browser and security considerations. Consequently, lowers their performance [8]. One of the advantages of the web application is the Cross-Platform compatibility, - (one codebase), WORA - feature[34, 36]. With the advances of HTML5, the user experiences on Web applications are comparable to that of native applications. As a result of the restriction of Web technologies, typical Web apps cannot take full benefit of the users device hardware functionalities resulting in poor performance compared to native applications in practice. Example of Web Application is online shopping applications.

Benefits of web application

- Cost: it cuts the additional costs of emerging separate applications for each mobile OS platform, thereby widening the target market[7]. Moreover, users have a stable platform of accessing the system, i.e. the browser.
- Accessibility: compared to other types of mobile applications, web applications are accessible anytime, anywhere, and via any PC with an Internet connection. Hence it fosters remote works.
- Installation and maintenance: the users can access it from the server through their device, once a new version or an improvement of the previous version is connected on the host server, thereby eliminating the need to upgrade the devices of every potential user. Hence installation and maintenance become easy.

Limitations of web application

- One of the limitations of web applications is that they do not have certain structures that are offered on native apps that provide an improved experience to the users.

- Users additionally need to depend on the availability and reliability of mobile networks/the internet to get to data through the applications [7].
- While building web applications, the developer has to factor in the different types and versions of browsers.

2.3.3 Hybrid apps

Mobile application publishers intend to maximise their investment yields by making their mobile applications integrable on various mobile platforms. In the hybrid application development process, the absence of approval and proper sanitisation for data communicated   through the scaffold interfaces may prompt security issues, because the data could begin unsafe third parties.

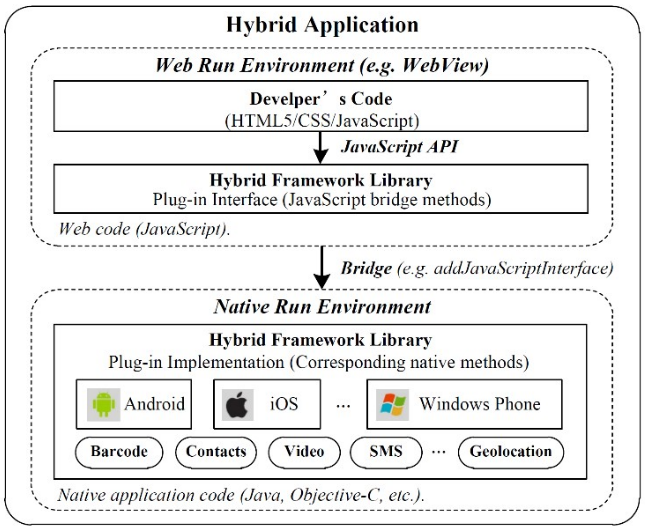


Figure 2.1: The architecture of Hybrid applications

The architecture of hybrid applications can be gone back to 2009 is the bridge among native and web applications. The diagram above indicates the architecture of hybrid applications, where the lower layer depicts where the code is stacked by the native components and run in the native runtime environment. The code in the higher layer is delivered by an inbuilt web part such as WebView provided by the mobile platform. Moreover, platforms such as Xamarin and Flutter can build native views instead of WebViews to give the users the native-like feeling. The Web component runs a unique communication mechanism called bridge communication for interoperation between Web code and native code. This communication mechanism also helps to enable web code to access the hardware resources and use the mobile platform's APIs. Tools and frameworks used hybrid applications includes; Apache Cordova, ManifoldJS, React Native[34], Flutter, Xamarin.

Benefits of Hybrid application

- Lesser development time: Hybrid mobile applications are more comfortable and quicker to compared to native application because the developed is required to write a single code for the different platforms in contrast native applications that is platform specified.
- Cost: Hybrid frameworks permit software publishers and developers to build a solitary version, write and maintain singular code bases for various platforms and this has a positive financial implication.
- Multi-stage support: the performance of hybrid application across various platforms relies on the system used to build the app. However, hybrid applications have no restriction on their development and can support almost all platforms.
- Performance: Hybrid apps offer rapid and performance simply like native

applications as opposed to web applications, and this is a direct result of the no or little reliance on network communication.

Limitations of Hybrid application

- Wang et al. categorised two aspects of risk relating to hybrid applications; code injection attacks and privacy leaks[8].
- Limited features: one of the significant disadvantages of hybrid applications is that it suffers from potential limitations in terms of OS and hardware integration.
- The native application offers a better look and feels to users, thereby enhancing their users' experience.
- Due to its cross-platform enhancement, debugging consumes more time compared to native applications.

The below shows the summary of web, native, and hybrid applications;

	Native apps	Web Applications	Hybrid Applications
Development Skills	Objective-C/Swift for iOS, Java/Kotlin for Android, .NET(C#) for windows phone	HTML, CSS, Javascript, JS frameworks	HTML, CSS, Javascript, Cordova/PhoneGap, Flutter,

Performance	Native codes have access to a good range of device functionality and the visual elements, and content structure is stored in the device memory, which makes them ready for immediate use.	The performance of the web application is directly related to the browser and a network connection.	The hybrid application utilises a native app wrapper that is utilised for communicating with the native device stage and the WebView.
Distribution	They can be downloaded from; Apple Appstore, Google Play, Windows App Store, Amazon App Store,	It is accessible through the web	They can be downloaded from; Apple Appstore, Windows App Store, Amazon App Store,
Accessibility	They have access to broad access to device features such as camera, geolocation. i.e. they have access to native APIs.	It has access to limited device APIs; therefore, it can only access some of the device's API	It has access to a high number features. i.e. they have access to a good number of native APIs

Portability	The codes written are platform specified, i.e. they cannot be reused on other platforms.	It utilises the WORA(Written Once, Run Anywhere) feature depending on the browser's performance, i.e. it can be run on multiple platforms.	One codebase can be used to build applications for different platforms. i.e. it can be run on multiple platforms.
Uses	It is used for applications where performance, graphics and overall user experience is most important.	Applications that performance requirements is not paramount but necessary, but need full device access.	Applications performance requirements are very low, and there is little or no need to access device hardware functionality.

Table 2.2: Summary of Web, Native, and Hybrid Applications

2.4. Cross-Platform Development.

The mobile application is usually developed with tools designed for heterogeneous platform- specified development [37]. Cross-platform development tools allow the usage of one programming language that supports multiple platforms (iOS, Android, Blackberry) as opposed to native development tools that are restricted to a single platform, often the mobile platform's SDK[38].

2.4.1 Approaches to Cross-Platforms

Cross-Platforms can be categorised into approaches such as[39]:

i. Cross-compiler

This methodology is utilised to create native applications for a few platforms. These applications are limited in the utilisation of specific features like camera, local notifications with a lower user interface. Cross-compiler works by isolating the built environment from the target environment and enough decoupling a source from its target. In mobile development, the framework gives a platform-autonomous API (application programming interface) using a mainstream programming language (like Dart, JavaScript, Ruby, or Java). The code is taken care of with the cross-compiler that converts code into platform-specific native applications targeted at the different platforms that the mobile application would run on. Advantages of this are that it incorporates the performance(because the application would be running natively on the gadget), improved client experience(it has full native access to the extent of explicit gadget functionalities like the gadget camera, sensors like GPS, accelerometer). One significant disadvantage of this is the complexity since cross-compilers are hard to compose, and consistency is needed with different mobile platforms and operating systems [40].

ii. Virtual machine (VM):

This approach helps to access and reuse more resources in cross-platform applications. Since VMs are more comfortable to sustain and more adaptable to expand when new features are added to the device and should have been upheld by the API hence, one the advantages of VM over cross-compilation is portability. There are a couple of disadvantages to this methodology. The application development relies upon the VM API (Application Programming Interface), not the device-specific platform, which expands the limitations in development.

Moreover, another disadvantage is the poor performance caused by the middleware connecting the code and the expected platform. A Virtual Machine is the product implementation of a machine (for example, a mobile device) that implements programs like a real physical machine. The Virtual Machine framework provides the API and also the runtime environment inside which the application would run on[40].

iii. Web-based (Pure Web):

These applications are navigated through the users' mobile browser. Web tools such as HTML5, CSS(Cascading Style Sheets) and JavaScript including: embedded canvas, web sockets, SQL databases, local storage, and animations are used for building these applications by working on the with the compatibility format of different mobile browsers, and the user can navigate this type of applications via mobile's browsers. This approach is minimal, especially when the developer needs a level of access to the device's resources [40].

iv. Hybrid (Hybrid Web)

This approach is also developed using HTML5, CSS and JavaScript. However, the difference between hybrid and web apps is that hybrid apps are executed into web browser container, and it then uses the device's web browser engine to access the device resources. i.e., the hybrid web app is based inside a thin covering native application which serves as a bridge to the device's working system and services. The web application is stored locally on the device upon installation, hence removing the requirement for internet connection, hence improving its speed and responsiveness. The disadvantage of this methodology is the poor performance and non- native user interface [40].

2.4.2 The Requirement for Cross-Platforms

M.Latif et al. proposed some desirable requirements of any cross-platform technology since the main object of cross-platforms are to publish mobile applications for multiple platforms taking into consideration the need for producing and maintaining high performing mobile application [41]s.

- Application Scalability and maintainability
- Access to devices features
- Resources consumption
- Security
- Development environment

2.4.3 Merits of Cross-Platform Development

One of the merits of cross-platform development is the reusability of codes, i.e. a single code base can be used for the development of a mobile application for different platforms. Moreover, the approach is cost-effective as it erases the need for building a platform-specific mobile application. Another advantage of this approach is performance since the app works like a

regular app on the user's ecosystem; and have full native access to a range of device-specific capabilities like integrated camera, sensors and so on.

2.4.4 The drawback to Cross-Platform Development

I.T. Mercado et al. used the Natural Language Process (NLP) models to classify the reviews of some users on about 50 mobile applications. Their work was carried out on both Android and iOS mobile operating system. It was observed that users complain more about hybrid applications than the native's application in the area of the usability, performance, reliability and security [42]

One of the pioneers of the cross-platform base mobile application was Facebook by leveraging on web technologies(HTML, CSS and JavaScript)[43] through the Hybrid approach. In 2012, they concluded that their cross-platform mobile application did not meet a certain level of the intended requirement because of the app's characteristic unique complexity. In addition to this, Facebook abandoned the app because of reported issues of scalability and unmet expectations of users' experience. Hence they continued with the Native application development. Facebook abandoning the project as one of the pioneers have raised questions on whether the approach could generate reliable applications[37, 44, 45].

2.5. Campus Shuttle

A shuttle is any form of transport that moves regular between two places, i.e. it usually moves in a predefined path or route. As explained in the concept of a university, a campus involves land, buildings of a college, including student residential. Therefore, campus shuttle is any form of transport that moves students, faculties, staffs and visitors from places within the campus.

In most campuses in Nigeria, campus shuttles are responsible for transporting students, staffs and sometimes visitors within the campus. Although there might be some exceptions where some campus shuttle transport riders out the campus. In Covenant, the shuttle services are responsible for transportation within the Campus and Canaanland. Covenant Shuttle system managed by the SBU(Strategic Business Unit) of the school, and Covenant University is a Christian mission private university in Nigeria founded by the Living Faith Commission in October 2002. The University is located at the Mission's international headquarter Canaanland, Ota, Ogun State Nigeria, is a fully residential community for staff and students. Covenant Shuttle vehicles are responsible for transporting students, staffs, faculties, mission members, visitors routing within Canaanland. Another example is the University of Lagos a premier federal university in Akoka, Lagos State Nigeria run a bus shuttle that is responsible for transportation to and fro the campus to notable bus terminals around the school while the car shuttle is responsible for movement within the campus.

2.5.1 Importance of shuttle transport on campus

Campus transportation planning is one of the crucial things being considered inside a campus master plan for some colleges campuses. A campus transportation system includes the development and activity of various traffic modes as a whole, including motor vehicles, shuttles, pedestrians, and bicycles [46]. Additionally, Campus transportation planning is a process that includes cautious considerations and analysis of various modes of transportation elements and planning objectives with the campus. Campus transportation planning likewise requires collection and analyses of an alternate assortment of existing and anticipated transportation data, data, for example, traffic counts, campus populace, and infrastructure.

Özgür et al. pointed out that the mode of transportation has effects in the daily routines of college environments, campus-workers, student's behaviour and attitudes[47]. A survey was taken using the google form tools amidst students of selected schools in Nigeria, and this selection was based on performance with Covenant being the first in the country. These schools include Covenant University, University of Ibadan, University of Ilorin, Babcock University, Obafemi Awolowo University, University of Lagos and Federal University of Akure. The survey showed that about 80% of the respondents have experience some level of frustration while awaiting shuttles at the terminals. Ineffective transportation on campus could serve as a source of stress, and this could cause deleterious mental health effects[48].

In a survey carried out by Cattaneo et al., safety, well-being, and sustainability are measured as preferences that may impact their choice of transport mode within university students[49]. Student transportation also affects the student's overall health and well-being. It can also ripple on the student's attendance, punctuality and her overall psychology in the lecture halls. In conclusion, student transportation has a significant effect on the quality of a student's education[50].

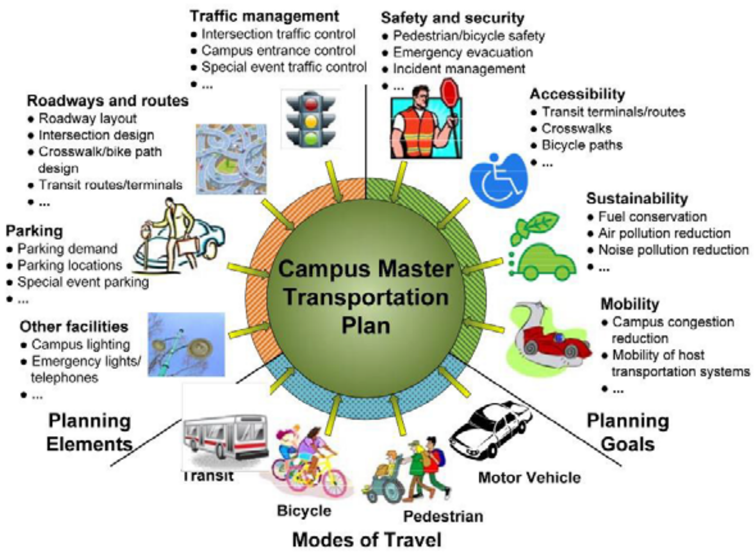


Figure 2.2: Campus Master Transportation Plan

Hence, the need to foster adequate transportation on the Nigerian university campus in other to foster the Sustainable Developmental Goals 9; building resilient campus transport infrastructure, promoting an inclusive and sustainable environment that would foster innovation and Sustainable Developmental Goals 11. Thereby, making our Nigerian campuses inclusive, safe, resilient and sustainable for foster learning, research, creativity, impacting research on the Nigerian University Campuses.

2.6. Intelligent Transport System.

Intelligent Transport Systems (ITS) is the use of communication and information technologies with the transportation infrastructure to enhance safety, improve mobility and efficiency [51, 52]. ITS is to make the transportation system more reliable, more efficient, secure, and safer using information, communications and control technologies to enhance the engaging quality of the public transport system. Also, ITS helps to address the rising congestion, which builds travel times and increased cost.

Intelligent Transportation Systems (ITS) has assisted with improving road efficiency, road safety, and diminishing the level of uncertainty in the transportation sector. It incorporates utilising cutting-edge technologies that are applied to all types of transportation to improve throughput, safety yet utilising advanced sensors and communication devices [53].

- GPS and GNSS (Global Positioning System)

2.7. Cloud computing

Cloud computing is the on-demand delivery, and sharing of a virtualised pool of IT resources over the internet [54]. Cloud computing can be simplified as the provision of online IT services. It offers services ranging from massive storage, management, real-time querying of databases, sharing of resources, and reliable infrastructure to accomplish large-scale computing[55]. Ibikunle L et al. stated that Cloud computing ranged from activities, for example, the utilisation of social networking sites to interpersonal computing and inferred that cloud computing is concerned about getting access to online software applications, data storage and processing power. Through cloud computing, capacity can increment dynamically without putting resources into new infrastructure[56]. Bala et al. pointed out five essential qualities of cloud computing, namely; wide-ranging network access, on-demand capabilities, swift elasticity, resource pooling and restrained service[57]. The Cloud can be seen as a collection of computer hardware networked together that works in sync to provide diverse aspects of computing as online services. The virtual collection of computing resources[58]. The cloud users have access to control it remotely via web interfaces.

2.7.1 Layers of Cloud Computing

Layers of cloud computing accessible over the internet [54, 59-62]

- Infrastructure as a service(Iaas)
- Platform as a service(PaaS)
- Software as a service(SaaS)
- Data as a service(DaaS)

Infrastructure as a Service(IaaS)

Computer infrastructures are delivered as service. It provides users with visual infrastructure such as server[57, 63]. It is a single-tenant cloud layer where the vendor's assets are only familiar with contracted clients as a service for their payment as they use. The primary purpose of the infrastructure as a Service model is to make available to users an option of renting resources such as networking resources, computing resources and storage resources[61]. Examples include DigitalOcean, Amazon Web Services (AWS), Cisco Metapod, Microsoft Azure, Google Compute Engine (GCE).

Platform as a service(PaaS)

Here, development environment services that can facilitate and run in-house built applications to deliver to the user, [57]. In PaaS, the provider's servers house the set of software and development tool. Examples include; Google App Engine, Microsoft Azure.

Software as a service(SaaS)

This model assists in giving the user access to effectively developed applications that are running in the cloud through the internet [57]. Software as a service refers to a business model in the software industry that offers Internet-based software application programs to clients through the network channels(e.g. the internet)[64]. SaaS is characterised as a software application delivery model that permits software publishers to deploys and host software applications in a multi-tenant - cloud-platform for its clients to work the application over the internet as services[65]. Examples include BigCommerce, Google Apps, Salesforce, Dropbox, ZenDesk, DocuSign, Slack.

Data as a service(DaaS)

In this model, data is readily available over a Cloud-based platform[57]. In DaaS, data is provided on demand to the user regardless of separation between the provider and consumer. The main aim of DaaS is to increase the user's accessibility to the data centre through the internet. Examples include Xignite(a company that makes financial data accessible to their users), D&B Hoovers(a company that makes business data accessible to their customers).

2.7.2 Cloud Storage

According to M. Akter et al., Cloud storage is an intangible entity, which gives users/clients capacity to store, recover, share, and synchronise documents on a relative core [66]. Cloud storage is a cloud computing model that includes storing/stashing data on the internet through a cloud computing provider in a remote location who oversees and works data storage as a service and accessible through the internet. Cloud storage is one of the services offered by cloud computing [60, 67]. Examples of cloud storage options include; Amazon drive, Google Drive, Microsoft Onedrive, Apple iCloud, Dropbox.

Cloud storage is available into five clouds; Personal, Public, Private, Hybrid[54, 62] and Community Cloud[56, 59].

i. Personal cloud: is a subdivision of the public cloud storage (mobile cloud storage). Personal cloud is responsible for keeping individual data in the cloud, making it accessible to the individual via the internet anywhere. It provides the individual with data synchronisation and sharing abilities across devices. An example is the Apple iCloud, Google Drive, Amazon’s Simple Storage Service, Windows SkyDrive.

ii. Public cloud: here, the enterprise and the data storage provider are different. i.e. there is the enterprise does not have a data centre so, all data are stored with the cloud storage provider. i.e. users can connect through other companies(enterprises) over the internet to cloud storage that’s maintained by a cloud provider[56, 59]. Examples are the Google App Engine, ESDS eNlight Cloud, Amazon Elastic Compute Cloud (EC2), Blue Cloud by IBM, Sun Cloud, and Microsoft Azure Services Platform.

iii. Private cloud: in Private cloud storage, the organisation and cloud storage provider are coordinated into the organisation's data place. For example, the enterprise is also the data centre [59, 68]; a type of cloud computing that just a single association/company, that ensures that the association is isolated from different organisations. Examples of private cloud providers include; Hewlett Packard Enterprise(HSE), VMware, Dell, Oracle, Cisco.

iv. Hybrid cloud: it is the cross path between public and private cloud storage where some essential, sensitive and personal data are kept in the enterprise(company) 's private cloud while other public data is stored in the cloud storage provider.. i.e. enterprise has the choice to select the kind of cloud storage (either public/private cloud storage) they want to use depending on the sensitivity and level of importance of the data. Hybrid cloud storage is flexible compared to public and private cloud services[59].

v. Community cloud: A community cloud is a multi-tenant platform that allows infrastructure to be shared by several companies same platform usually because they have similar needs and concerns. The cloud may be managed internally or through a third party [56, 57, 59].

Types of cloud data storage

i. Object storage: Also known as object-based storage, tries to solve the problem of scalability. It is a data storage model for handling extensive unstructured data; this type of data cannot be readily organised into a traditional database with row and column. It is a strategy for managing and manipulating data storage as a distinct unit known as objects. It enables the creation of self-guided, shared and safe storage for storage networks. [69]. Object storage adds comprehensive metadata to the file, in this way dispensing with the tiered file structure utilised in file storage, stores the file into a flat address space, called a storage pool. This metadata is the distinctive feature of object storage since it gives an in-depth analysis of the utilisation and function of data in the address space(storage pool). Examples include Amazon S3, Rackspace Cloud Files, Azure Blob Storage, Google cloud storage services. Benefits include Storing/managing unstructured data, Scalability, Customisable metadata, Affordability, Cloud compatibility.

ii. File storage: File storage is a storage methodology that is used in organising and storing data on

    - An internal hard drive,
    - Direct Attached Storage(DAS),
    - External hard disk or Network-Attached Storage (NAS) - for file sharing[70]

    In this storage model, data is stored in files, file are stored and organised in folders, and the folders are arranged under a hierarchy of directories and subdirectories. It is used for; Local file sharing, Centralised file collaboration, Archiving/storage, Backup/disaster recovery. Examples; Amazon Elastic File System (EFS), Amazon FSx for Windows File Server

iii. Block storage: Storage Area Networks(SANs) or cloud-based storage environments are used to store data files. It also provides fixed raw storage capacity. It is either locally or attached to a network, and it is usually formatted with a file system like FAT32, NTFS, EXT3 and EXT4. Block storage provides a traditional storage device like a hard drive over the network. Examples include AWS Elastic Block Storage (EBS), Rackspace Cloud Block Storage, Azure Premium Storage, Google Persistent Disks.

2.8. Review of related works

Seong-Woo Kim et al. designed an autonomous campus mobility service using a driverless taxi at Seoul National University [71], they harness the use of the GPS embedded on the mobile devices to track the location of the users and displaying it on the on-screen map. The system queues order through a central server and assigns the order request to a driverless taxi. The driverless taxi is equipped with 3D lidar that is used for map building and updating, localisation, vehicle/pedestrian detection. The driverless taxi is equipped with GPS to show the instantaneous location of the taxi to the users. The steering, acceleration and deceleration of the taxi is control by a CAN. One of the limitations of this approach is the impact of weather. The potential impact of weather on autonomous driving should be considered because of the performance disturbance of wireless communication and sensor measurements.

Moreover, there is a limitation to the ride-sharing feature of regular campus shuttle, thereby leading to a reduction in the ratio of the campus shuttle to the student/user body. Hence, there is need for a sharable campus shuttle like the case of an uber pool[72] and human-driven campus shuttle to navigate the shuttle in case of an unfavourable weather condition. Other limitation includes a possible glitch in the system, i.e. security and safety concerns, and reduced job opportunities for drivers.

Automated vehicles are beneficial in reducing crashes, increasing mobility, green and sustainable compared to conventional vehicles. There is a difficulty regarding the assurance and affirmation of the social, safety and health implications of Automated vehicles and how other road users relate with these vehicles[73].

Shafri et al. proposed a realtime campus tracking application for buses on campus, a mobile application that helps campus users know the present location of the bus in real-time [17]. The application shows approximation time of arrival and the number of people inside the bus. In this work, two devices were embedded inside the bus; GPS (Tracker device) and counter device. The data gotten from these devices are stored into firebase; cloud database. Moreover, avenues were also provided for students to query the database. PhoneGap was used as the platform to develop the application. The GPS Tracker device used Arduino, and the counter device used Raspberry PI to transmit data. The limitation to this approach is that since smart-devices are cost-efficient GPS enabled device[71] the approach taken by Shafri et al., would have been more cost-efficient if the GPS embedded inside the smartphone was harnessed. The drivers could be provided with smart devices that would be used to track the location of the vehicles. The smart device could also serve as the linking channel for storing information in the cloud.

Pengfei Zhou et al. proposed a system of forecasting bus arrival based on mobile phone participatory sensing. He pointed out the effects culminating from the inability of city travellers to not being able to track bus locations. They discovered that this also caused a rise in the number of vehicles hence contributing to the greenhouse effect[74].

Nalawade et Akshay proposed a bus tracking system that used Raspberry Pi to primarily track buses on Google map through a web application and an android application. The system aims to forecast the bus arrival time for students and utilise the various sensors to show extra data about the bus for security purposes, modules like GSM/GPRS module was joined for internet connection. The area is store in

MySQL database through a webserver and refreshed to show on the map[75].

Salman Almishari et likewise proposed a tracking system dependent on GPS for real-time tracking systems. The system is into three parts, namely the tracking unit, cloud, and android application. Tracking unit is situated in the vehicle and sends the necessary data like the temperature and real-time area of the vehicle to the cloud. The system gives a real-time area on the map by identifying the latitude and longitude of the vehicle[76]. The limitation of this work is cost-effectiveness. The tracking unit could be replaced by the mobile devices of the drivers since modern devices have an inbuilt GPS.

A system was proposed and implemented by Prashant A. Shinde and Mane that would increase the security of school transports through real-time monitoring. The system is made out of a Raspberry Pi; Embedded Linux board and android application, SIM900A Module that houses the GPS GPRS GSM. The GPS gives the instantaneous location of the vehicle; GPRS sends the tracking information got to the server and the GSM for sending an alert message to the allocated vehicle's proprietor number. The information from the site is occasionally recovered and check with the coordinates updated in the database and the instance of a miscorrelation, the system sends an alert the proprietor concerning the miscorrelation[76].

CHAPTER THREE
METHODOLOGY

3.1. Introduction

In this chapter, the methodology for the implementation of a campus shuttle tracking management system is discussed. The development techniques and tools for the system are also discussed in this chapter. The accessibility and arrangement of the users' screen are discussed here.

3.2. Choice of Development tools

3.2.1 Dart

Google developed dart and, it first appeared on the 10th of October 2011 and approved as a standard by ECMA (ECMA-408). It is an open-source general-purpose programming language. Dart is an object-oriented language with a syntax structure similar to C. Dart is a combination of the best features of major languages with some language specifics.

Classes

Classes usually start with uppercase, i.e. they are usually written in upper camelcase. Methods(functions) begin with lower case, i.e. they are usually written in lower camelcase can be seen as the object blueprint. Inside the class, we can find their properties(variables declared inside the class), constructors(used for initialising and setting the values of our object), and functions(reusable code that is used to perform a single, related action).

Extend and mixin

Mixin is a class can share methods and attributes with multiple classes but logically with less connection. For example a “Person” class with a mixin “Agility”. The Agility mixin is like an adjective used for qualifying living things. Extend is a concept of inheritance which is part of the feature of Object-Oriented Programming is a syntax where a new class “A” has the properties and functions of the extended class “B” and is at liberty to override some the functions or methods using the @override keyword. For example, a “Person” class extending a “Mammal” class; there is a closely related as a person is a subset of mammal. Moreover, a class can extend only one class while a class can be used with more than one mixins, and all their properties would be added together. A mixin could be seen as a utility function provider.

Variables

Variables store references. String firstName = 'Seun';

The variable called firstName above contains a reference to a String object with a value of ‘Seun’.

- Final and const: A variable set to be final can be set only once; a const variable is a compile-time constant.

Built-in types in Dart

The Dart language supports the following types:

- Numbers(int and double)
- Strings: is a sequence of UTF(Unicode Transformation Format)-16 code units, and it can either be created with a single or double-quotes.
- Booleans: this has only two objects(true and false), and they are both compile-time constant
- List: also known as an array is ordered a group of objects

- Sets: This is an unordered grouping of unique items
- Maps: A map can refer to as an object that links keys and values. These keys and values can be a variable of any type of object.
- Runes: these are Unicode characters in a string expressed as an integer.
- Symbols: this signifies an operator or identifier stated in a Dart program

3.2.2 Flutter

Flutter helps to build for more than one platform while maintaining quality across each platform such as android, iOS and the web with one codebase. Hence flutter uses a programming language that is supported on these platforms with a fast development experience hence Dart as the programming language for a flutter. Dart is a client-optimised language for fast mobile applications on cross-platforms. Dart eradicates slow compilation and debugging cycles through hot reload and this injects updated source code into a running application and rebuilds the UI, updates almost immediately. Dart production compilers compile to ARM, x64 machine code for mobile desktop and backend or optimised JavaScript for the web.

Other tools for developing cross-platform application are Xamarin and React Native.

In 2011, Xamarin, presently owned by Microsoft, came up with a platform for hybrid mobile apps through its signature product, ‘Xamarin SDK with C#’. Xamarin contributed to the revolution of hybrid mobile applications, the ease in having one code base for diverse platforms. Xamarin's Mono execution component additionally reacts with Java or Objective-C runtime directly and utilises the majority of the native code there. Even though Xamarin architecture looks strong, it does not have great support for the Kotlin or the Swift runtime, which are official runtimes for creating android and iOS apps. The Flutter engine has the more significant part of the native components in the framework or structure itself.

Moreover, Xamarin is available for free with limitations and flutter is open-sourced hence developed have reasonable control over the code. Also, flutter apps have proven to a much better performance than Xamarin apps.

React Native was started internally by Facebook and was later opened sourced in 2015. React Native link the best parts of native development with React, a JavaScript library for building user interfaces. It also utilises the JavaScript bridge to build up communication with native modules, which results in poor performance. However, the Flutter engine has a large portion of the native parts in the framework itself. Thus, it eliminates the need to bridge communication with the native parts. Other advantages of flutter over react native incorporates is that flutter offers more useful documentation and CLI support for arrangement and configuration, flutter is richer in development APIs and U.I. Components and React Native, on the other hand, places more dependency on third-party libraries. Moreso, the React Native community, has no official support for integration and U.I. Level testing, while flutter has excellent documentation and a rich set of testing features. Lastly, Flutters utilises the CICD services through its secure use of CLI tools while React Native does not provide any official instructions for CICD. Hence the use of flutter as the framework for developing the proposed system.

Flutter has app-specified libraries that rely on the user interface elements, usually referred to as Widgets. In flutter, Cupertino class widgets are used for building iOS look and feel user interfaces. The Material design class houses the widget that can be used for the android look and feel, and some for iOS also.

A Widget(local) state affects only a widget on its own and does not affect other widgets in the application. An example is a loading spinner, form input.

The provider package and pattern work by adding a provider pattern through a provider package to a project. It contains a global container/data store that is attached to a widget in the application for effective management and all child in the widget can listen to the data provider, and only the part that listens to the data provider is rerun.

A State

A-State is information that can be analysed synchronously when the widget is built and can change during the lifetime of the widget. A state is a data which affects the U.I. (User Interface) and can change over time. The U.I. is a function of the state(data). There is App-Wide State and Widget(local) State. The App-wide State affects the entire application or significant parts of the app.

An example is user Authentication. Everything in flutter is a widget, and a widget can either be stateful or stateless. A stateless widget never changes while a stateful widget is dynamic depending on the data it receives. The State object manages a widget's state, and the state consists of values that can change, i.e. values the widget depends on. When the widget's state changes(the values that can change), the state object calls setState(), thereby redrawing the stateful widget.

3.2.3 Firebase

Firebase is a mobile application development platform birthed by Google that supports in build, develop, and advance application. Firebase offers one of the layers of cloud computing, which is Backend-as-a-service. The firebase console is displayed in the figure below.

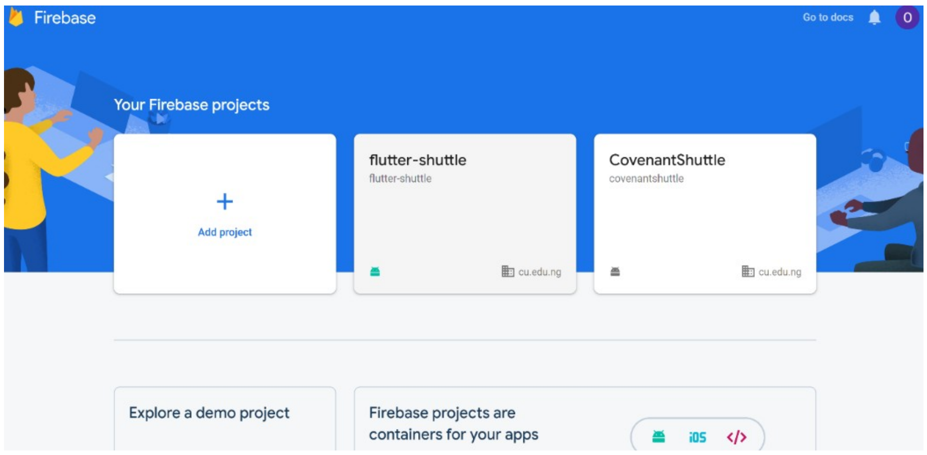
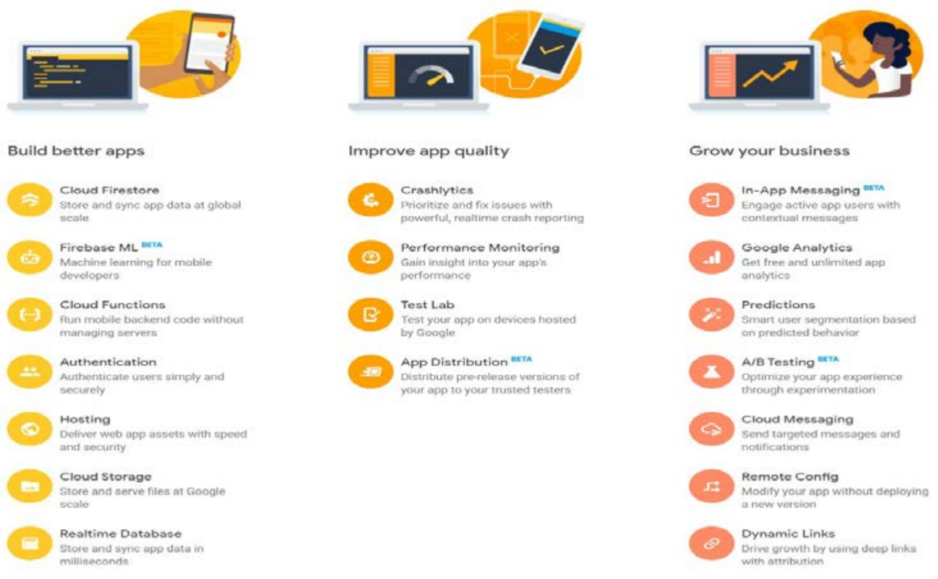


Figure 3.1: Firebase console interface

Firebase is a real-time database, file storage, provides user authentication services, and it can be used to host services for static files and data, the Figure 3.2 below shows a complete firebase suite.


Build better apps

Cloud Firestore
Store and sync app data at global scale

Firebase ML BETA
Machine learning for mobile developers

Cloud Functions
Run mobile backend code without managing servers

Authentication
Authenticate users simply and securely

Hosting
Deliver web app assets with speed and security

Cloud Storage
Store and serve files at Google scale

Realtime Database
Store and sync app data in milliseconds

Improve app quality

Crashlytics
Prioritize and fix issues with powerful, realtime crash reporting

Performance Monitoring
Gain insight into your app's performance

Test Lab
Test your app on devices hosted by Google

App Distribution BETA
Distribute pre-release versions of your app to your trusted testers

Grow your business

In-App Messaging BETA
Engage active app users with contextual messages

Google Analytics
Get free and unlimited app analytics

Predictions
Smart user segmentation based on predicted behavior

A/B Testing BETA
Optimize your app experience through experimentation

Cloud Messaging
Send targeted messages and notifications

Remote Config
Modify your app without deploying a new version

Dynamic Links
Drive growth by using deep links with attribution

Figure 3.2: The Firebase suit

Traditionally, most databases require making HTTP calls to get and synchronize data to gadgets, and they only give data only when queried for it. An app associated firebase is not interfacing through normal HTTP, but instead, it is interfacing through a WebSocket. WebSockets offer a persistent connection between a client and server that the two players can use to exchange data whenever in real-time. The client sets up a WebSocket connection through a procedure known as the WebSocket handshake. WebSocket API presents an advanced technology that makes it feasible for a two-way interaction between the client's program and a server. With this API, the app can converse with a server and get event-driven responses without surveying the server for an answer. WebSockets is faster than HTTP. The apps do not need to make individual WebSocket calls since one attachment connection is the length. The entirety of the data syncs automatically through that single WebSocket relying upon the quality of the network available. The Figure 3.3. below shows a graphical explanation,

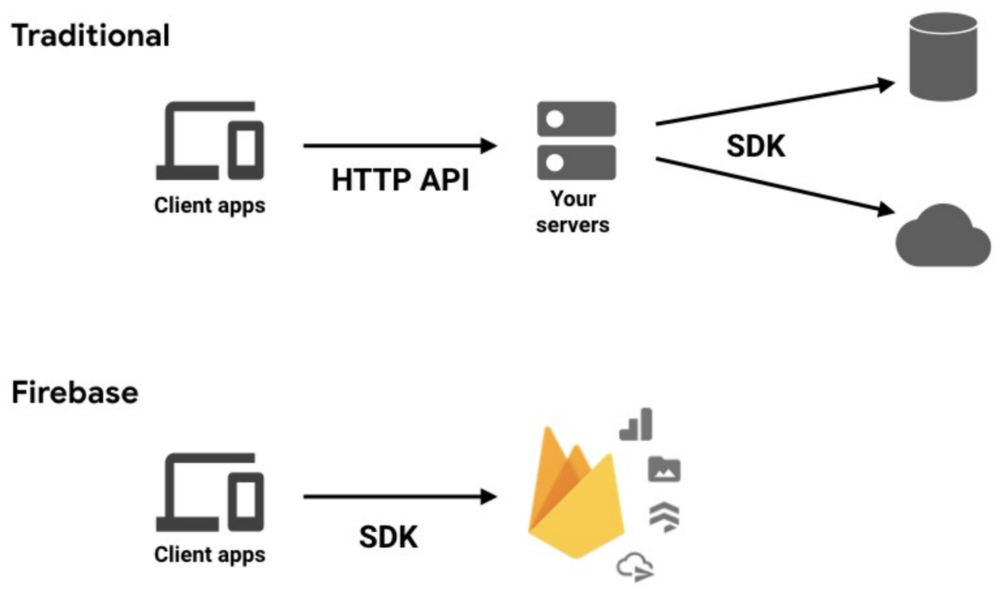


Figure 3.3: Comparison of Traditional databases requests with Firebase

One of the advantages of Firebase application is that it remains responsive even when offline because the Firebase Realtime Database SDK stores data to disk. Once connectivity is restored, the client device receives any changes it missed, synchronising it with the current database server state.

For the implementation of this project, Firebase would be used for the following services.:

- Authentication provides: this is used to sign up, sign in and sign out users
- Cloud firestore: this is used to store real-time changing geopoints and real-time user data that some parts of the application would be listening to providing a real-time, cloud-hosted NoSQL database.

3.2.4 Google Maps Platform

The Google Maps Platform consists of a set of APIs and SDKs that are used for providing real- world Mapping system to users, and it is managed from the Google Cloud console.

The google maps platform is grouped into three, and each is sub-grouped with different APIs and SDKs

1. Maps: this helps to build and customised static and dynamic maps, 360 degree and Street View imagery bring real-world experience to the users

- Maps SDK for Android – this adds a google map to an android application
- Maps SDK for iOS – this adds a google map to an iOS application
- Maps JavaScript for API – this adds a responsive/ interactive map to a website, and it allows customisation with the developers content and imagery
- Maps Embed API – this creates a responsive/interactive map, or a Street View Panorama to a site through an HTTP request.
- Maps Static API – this adds an embeddable and simple map image with minimal

code to a website

- Street View Static API – this embeds real-world images with 360 degrees panoramas.
- Maps URL – this launches Google Maps and creates actions like directions using a cross-platform URL scheme.

2. Routes: this serves to give users the best way between two locations with real-time – traffic updates and high – quality directions.

- Directions API – this provides directions for walking, biking, driving and transit between several locations.
- Distance Matrix API – this serves as the calculator for travel times and distances for multiple destinations.
- Roads API – helps to control the precise route a vehicle or a moving body can travel on the road.

3. Places: these set of APIs helps users to discover the world with over 100 million places with rich location data enabling them to find places with addresses, phone numbers and real-time signals.

- Places SDK for Android – this adds details of millions of places to an Android application, it also allows autocomplete feature from the user's queries and convert these addresses to geographic coordinates and vice versa.
- Places SDK for iOS – this adds details of millions of places to an iOS application, it also has autocompleted feature from the user's queries and convert these addresses to geographic coordinates and vice versa.
- Places Library, Maps JavaScript API– this adds details of millions of places to a website, it also has autocompleted feature from the user's queries and convert these addresses to geographic coordinates and vice versa.

- Places API – this provides up-to-date information of millions of locations using HTTP requests
- Geocoding API – the converts geographic coordinates to addresses and vice versa.
- Geolocation API – this return the location of a device using the location data from cell towers and WiFi nodes without relying on the device G.P.S.
- Time zone API – this fetches the time zone for a specific latitude and longitude coordinate.

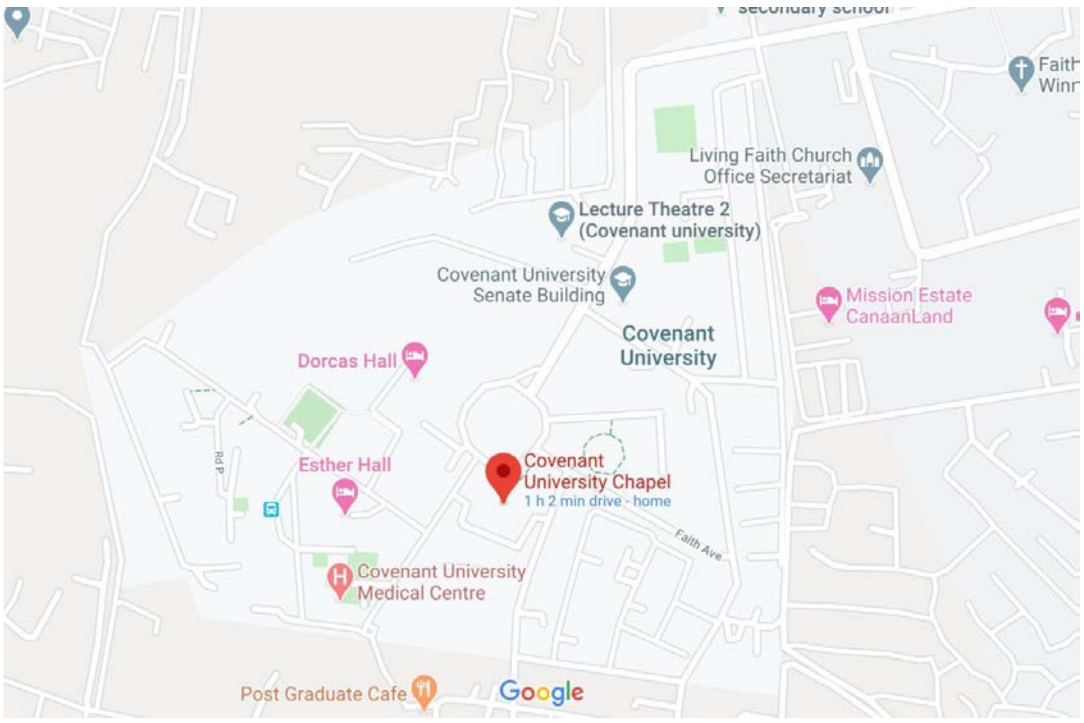


Figure 3.4: An Overview Of Covenant University On Google Map

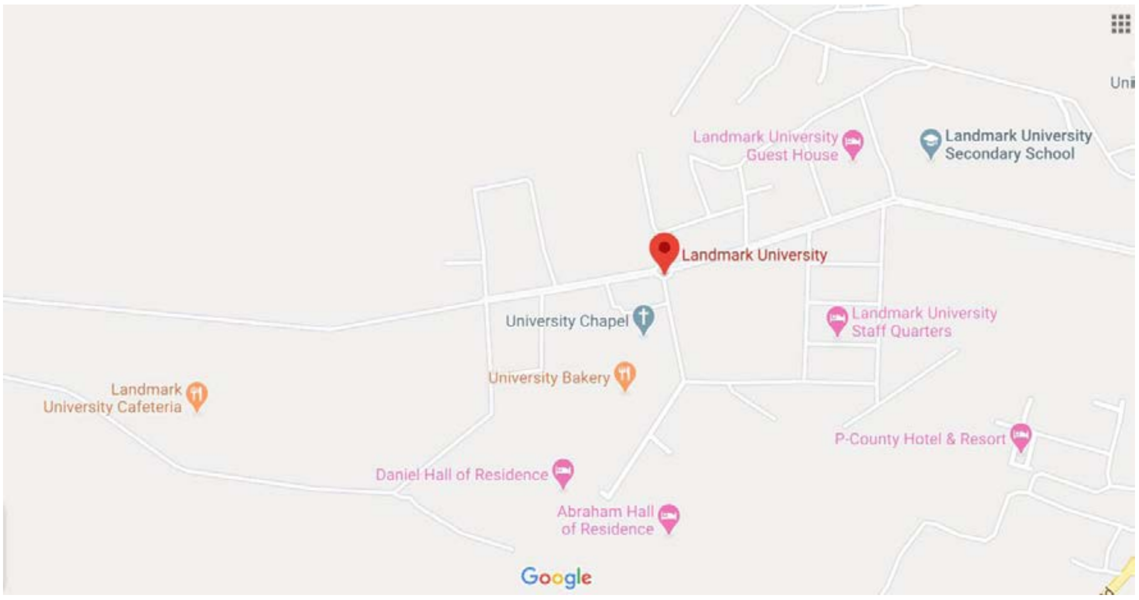

Landmark University Guest House
Landmark University Secondary School
Landmark University
University Chapel
Landmark University Staff Quarters
University Bakery
Landmark University Cafeteria
P-County Hotel & Resort
Daniel Hall of Residence
Abraham Hall of Residence
Google

Figure 3.5: An Overview Of Landmark University On Google Map

The following services from the google maps platform would be used in the course of this project.

- Maps SDK for Android
- Maps SDK for iOS
- Directions API
- Places SDK for Android
- Places SDK for Android
- Places API
- Geolocation API

3.3. Project requirements

Here the details about the functional and quality requirements of the system are stated. It contains an explanation of the inputs into and outputs from the system. It also gives a description of the users, hardware, software, and a description of the UIs contained in the application.

User Interfaces

A new user of the mobile application would see a login page upon opening the application. If the user is not registered, the user would be required to do so from the login page on the register button.

Upon registration, the category of user would be required from the users; categories ranging from drivers, faculty/staff, students and administrator. Every user would have a profile page where they can edit any of their submitted information. If the user is a driver, the user would be required to submit an ID that must be generated by the administrator before the registration

then the driver would be requested to select the type of shuttle he' she drives and also to give the mobile application to use the device's inbuilt GPS to track the driver's location. When the driver permits the application, the current position of the driver would be indicated on the google maps in the application.

All riders(i.e. students, faculty, staff) would have access to the estimation table where an estimated time for each driver to get to a particular bus stop would be shown in accordance to the type of shuttle selected by the driver. Moreso, all riders excluding the undergraduate students would be allowed to request a pickup, once a request is placed the database is updated. The car shuttle drivers would be alerted with the request, and immediately a driver accepts their request, the would-be notify and given an option to track the driver as the driver approaches their location.

The administrator would be able to monitor the number of pick up request made by each driver, their closing and starting time. The administrator also would be given room to communicate with these drivers. Upon signing up, the administrator would be required to enter a series of admin authentication keys that would be delivered to the client for managing the application. These keys would be available to whomsoever the client deems fit to be an administrator. The administrator is also responsible for generating IDs for the registration of each driver. The IDs would serve as the driver's authentication.

Hardware interfaces

The mobile application does not specifically require any designated hardware, i.e. it does not have any direct hardware interfaces. The inbuilt GPS manages the GPS activities in these devices. The underlying operating system manages the mobile application on the mobile phone and firebase server. The accelerometer in the driver's phone is utilised to provide the database with the current speed, heading directions of the drivers.

Software interfaces

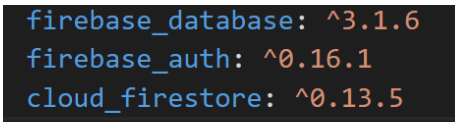
The application communicates with the inbuilt GPS application in fetching geographical information about drivers, users and also the graphical representation of it with the database in order to sync the information amidst riders.

The mobile app also communicates with the inbuilt accelerometer in order to get the current speed of the drivers and syncing it with the database for the timing calculations.

3.4. Implementation

The proposed system would consist of nine screens namely;

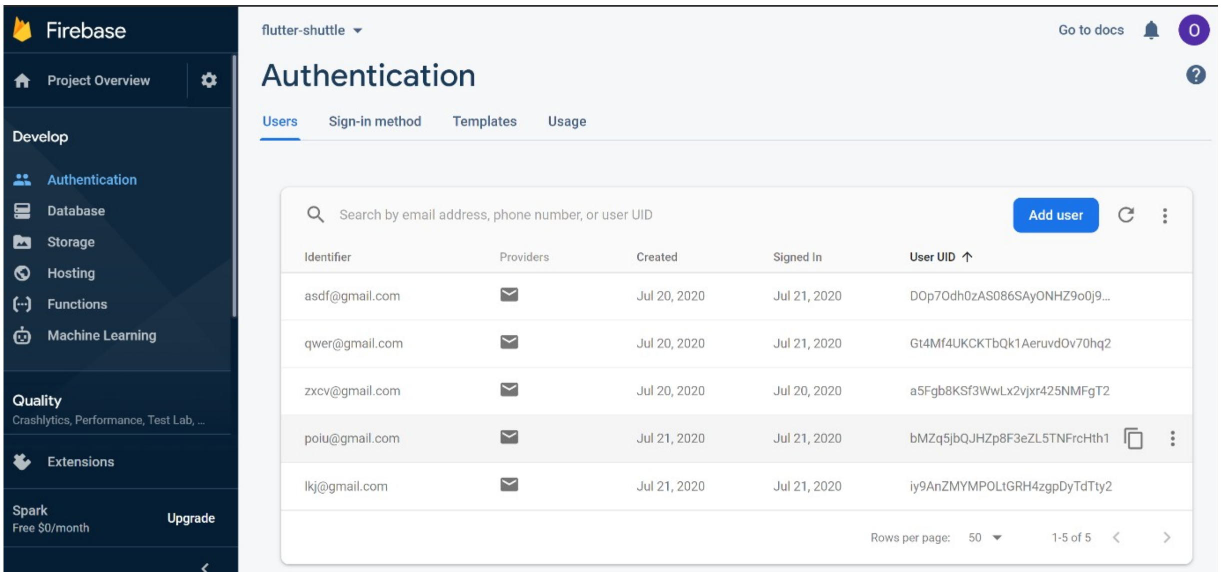
- the Login screen: this is the home screen where existing/ registered users request for permission and are granted permission into the system. A google flutter package (firebase_auth) is responsible for this authentication process. The login screen text form input takes in two inputs, namely the username and the user's password. Although the signIn approach of this package requires two arguments, namely, the email address and the user's password. The database is queried to get the email linked to the username before calling the signIn approach of the (firebase_auth). Moreover, querying the database is made possible through a flutter package known as cloud_firestore and firebase_database. The email and password are verified with the registered email and password on the authentication page, as shown in Figure 3.6.



```
firebase_database: ^3.1.6
firebase_auth: ^0.16.1
cloud_firestore: ^0.13.5
```

Figure 3.6: Firebase packages

- The Sign Up pages: the sign up comprises of two screens. Aim of the signup pages is to collate data that would be needed in running and managing the screens, and states of the application. The data collated are then stored in the cloud firestore database with the help of the packages in Figure 3.6. The register email and password are therefore on the authentication page overview.



Authentication

Identifier	Providers	Created	Signed In	User UID ↑
asdf@gmail.com	✉	Jul 20, 2020	Jul 21, 2020	DOp7Odh0zAS086SAyONHZ9o0j9...
qwer@gmail.com	✉	Jul 20, 2020	Jul 21, 2020	Gt4Mf4UKCKTbQk1AeruvdOv70hq2
zxcv@gmail.com	✉	Jul 20, 2020	Jul 20, 2020	a5Fgb8KSf3WwLx2vjxr425NMFgT2
poiu@gmail.com	✉	Jul 21, 2020	Jul 21, 2020	bMZq5jbQJHZp8F3eZL5TNFrcHth1
lkj@gmail.com	✉	Jul 21, 2020	Jul 21, 2020	iy9AnZMYMPOLtGRH4zgpDyTdTty2

Figure 3. 7: Flutter Authentication page

- DriverPermissionPage: this is the welcome screen for the drivers. The locations of the drivers are gotten from the widgets in this screen. Their locations are therefore uploaded to cloud firestore to provide synchronisation to all users connected to the application. The location is extracted from the device's G.P.S. via the location package. The device's location can also be gotten from the cell towers and WiFi through the geolocator package.

- DriverMapScreen: this is the active screen accessible only to the drivers where there are real-time updates about the driver's location on a map.

- UsersMapScreen: this is an active real-time screen where all the available drivers' location is shown with the mapped out an area of the campus. For the implementation of this project, Covenant University would be used. This screen is accessible by all users, excluding the drivers.

- EstimatedTimeScreen: this screen shows the estimated time derived from the calculation of the drivers' distance between two coordinates provided by the locations API's location data. The driver's time is, however, calculated with the use of distance- time formula. Moreover, this screen comprises of four widgets because of the four types of shuttle operating in Covenant. This screen is accessible by all users, excluding the drivers.

- OrderingRequestScreen: this screen is made accessible to only the staff, faculty and postgraduate students. On this screen, orders are made from one destination to another on campus to drivers driving car shuttle.
- PendingOrdersScreen: this is the screen where the car shuttle drivers receive and accept orders from the OrderingRequestScreen.
- UpdateProfile: this screen is accessible to all users to update or edit or change their registered profile from the database.
- PreviewOrderScreen: this screen is shown to drivers before and upon accepting a pickup request placed by a user.
- TrackDriverScreen: this screen is shown to the user when a driver accepts their pick up request.

Use Case Diagrams

Use case diagrams are employed to collect the requirements of a system, including internal and external impacts, generally design requirements. The user case diagrams in Figures 3.8, 3.9, 3.10 and 3.11 shows the number of screens and the level of access of each user.

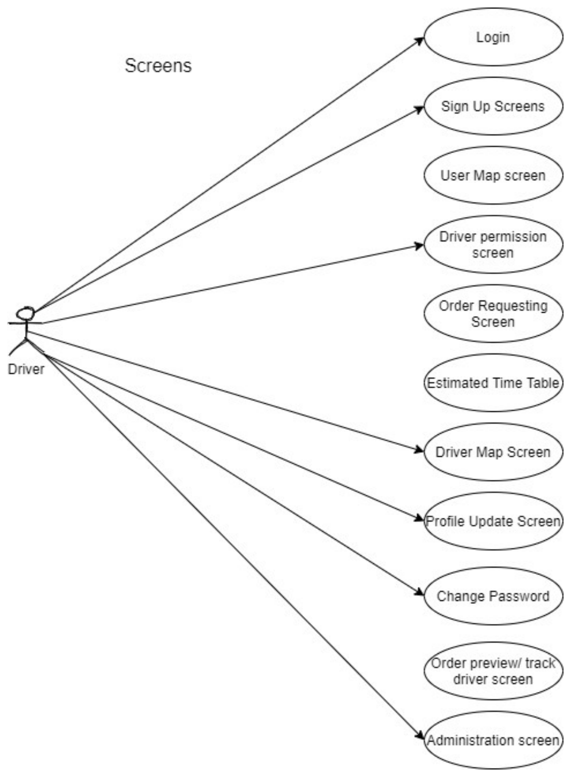


Figure 3.8: Use case diagram for the driver's screen system

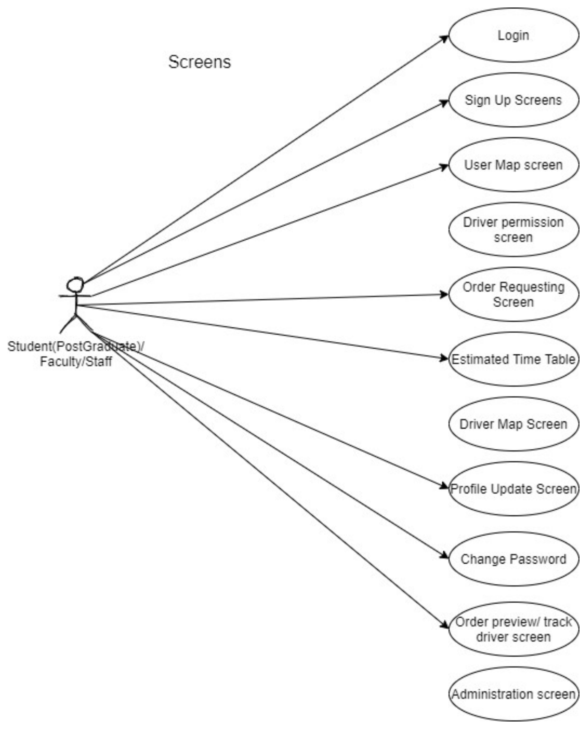


Figure 3.9: Use case diagram for the student (PG, Faculty/Staff) screen system.

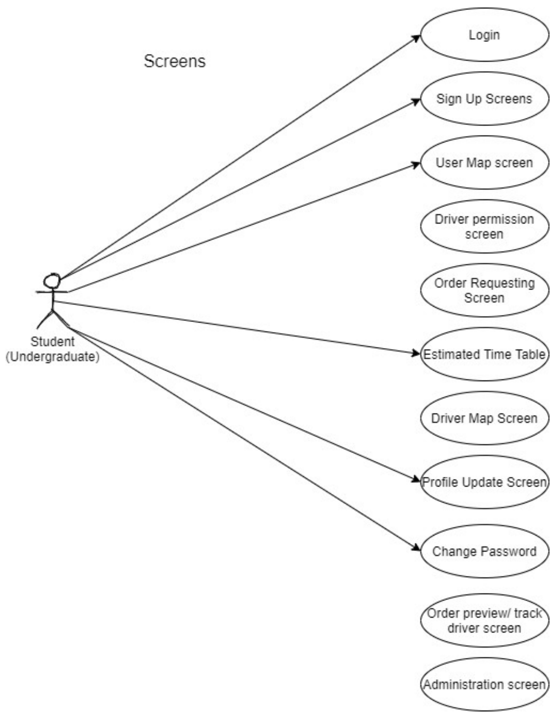


Figure 3.10: Use case diagram for the student Undergraduate screen system.

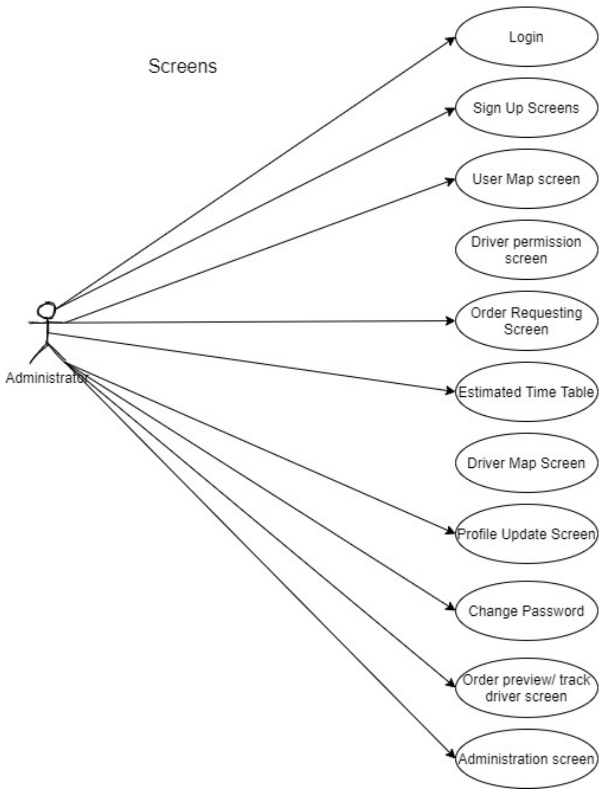


Figure 3.11: Use case diagram for the student Undergraduate screen system.

Class diagrams

A class diagram is a static diagram because it describes the static perspective on an application. A class diagram is used for building blocks of executable codes for the application and also describing and documenting different aspects of the system. The class diagrams show the attributes and methods in a class and also the relationship between classes in a project.

The class diagrams in Figures 12, 13, 14 and 15 show the attributes and functions in all the classes used to construct this application.

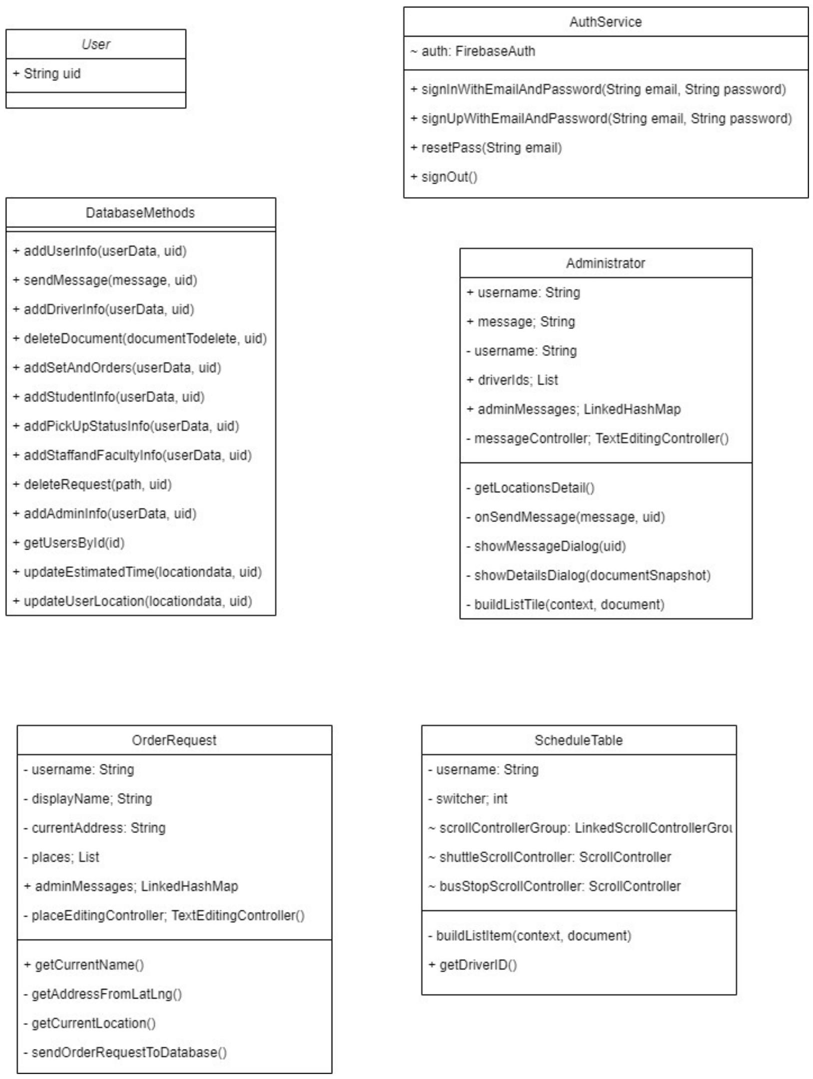

User
+ String uid
AuthService
~ auth: FirebaseAuth
+ signInWithEmailAndPassword(String email, String password)
+ signUpWithEmailAndPassword(String email, String password)
+ resetPass(String email)
+ signOut()
DatabaseMethods
+ addUserInfo(userData, uid)
+ sendMessage(message, uid)
+ addDriverInfo(userData, uid)
+ deleteDocument(documentTodelete, uid)
+ addSetAndOrders(userData, uid)
+ addStudentInfo(userData, uid)
+ addPickUpStatusInfo(userData, uid)
+ addStaffandFacultyInfo(userData, uid)
+ deleteRequest(path, uid)
+ addAdminInfo(userData, uid)
+ getUsersById(id)
+ updateEstimatedTime(locationdata, uid)
+ updateUserLocation(locationdata, uid)
Administrator
+ username: String
+ message; String
- username: String
+ driverIds; List
+ adminMessages; LinkedHashMap
- messageController; TextEditingController()
- getLocationsDetail()
- onSendMessage(message, uid)
- showMessageDialog(uid)
- showDetailsDialog(documentSnapshot)
- buildListTile(context, document)
OrderRequest
- username: String
- displayName; String
- currentAddress: String
- places; List
+ adminMessages; LinkedHashMap
- placeEditingController; TextEditingController()
+ getCurrentName()
- getAddressFromLatLng()
- getCurrentLocation()
- sendOrderRequestToDatabase()
ScheduleTable
- username: String
- switcher; int
~ scrollControllerGroup: LinkedScrollControllerGrou
~ shuttleScrollController: ScrollController
~ busStopScrollController: ScrollController
- buildListItem(context, document)
+ getDriverID()

Figure 3.12: Class diagrams

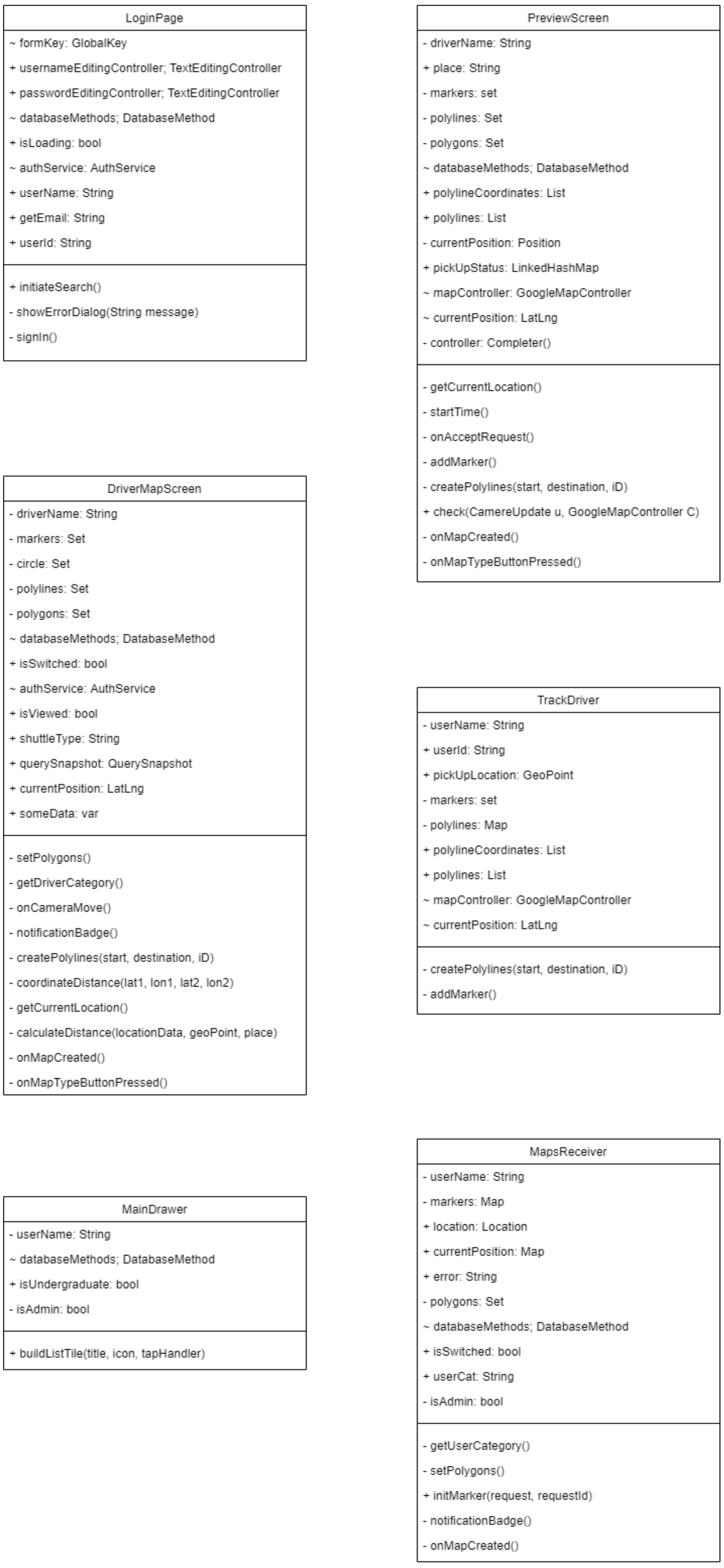

LoginPage
~ formKey: GlobalKey
+ usernameEditingController; TextEditingController
+ passwordEditingController; TextEditingController
~ databaseMethods; DatabaseMethod
+ isLoading: bool
~ authService: AuthService
+ userName: String
+ getEmail: String
+ userId: String
+ initiateSearch()
- showErrorDialog(String message)
- signIn()
PreviewScreen
- driverName: String
+ place: String
- markers: set
- polylines: Set
- polygons: Set
~ databaseMethods; DatabaseMethod
+ polylineCoordinates: List
+ polylines: List
- currentPosition: Position
+ pickUpStatus: LinkedHashMap
~ mapController: GoogleMapController
~ currentPosition: LatLng
- controller: Completer()
- getCurrentLocation()
- startTime()
- onAcceptRequest()
- addMarker()
- createPolylines(start, destination, iD)
+ check(CamereUpdate u, GoogleMapController C)
- onMapCreated()
- onMapTypeButtonPressed()
DriverMapScreen
- driverName: String
- markers: Set
- circle: Set
- polylines: Set
- polygons: Set
~ databaseMethods; DatabaseMethod
+ isSwitched: bool
~ authService: AuthService
+ isViewed: bool
+ shuttleType: String
+ querySnapshot: QuerySnapshot
+ currentPosition: LatLng
+ someData: var
- setPolygons()
- getDriverCategory()
- onCameraMove()
- notificationBadge()
- createPolylines(start, destination, iD)
- coordinateDistance(lat1, lon1, lat2, lon2)
- getCurrentLocation()
- calculateDistance(locationData, geoPoint, place)
- onMapCreated()
- onMapTypeButtonPressed()
TrackDriver
- userName: String
+ userId: String
+ pickUpLocation: GeoPoint
- markers: set
- polylines: Map
+ polylineCoordinates: List
+ polylines: List
~ mapController: GoogleMapController
~ currentPosition: LatLng
- createPolylines(start, destination, iD)
- addMarker()
MainDrawer
- userName: String
~ databaseMethods; DatabaseMethod
+ isUndergraduate: bool
- isAdmin: bool
+ buildListTile(title, icon, tapHandler)
MapsReceiver
- userName: String
- markers: Map
+ location: Location
+ currentPosition: Map
+ error: String
- polygons: Set
~ databaseMethods; DatabaseMethod
+ isSwitched: bool
+ userCat: String
- isAdmin: bool
- getUserCategory()
- setPolygons()
+ initMarker(request, requestId)
- notificationBadge()
- onMapCreated()

Figure 3.13: Class diagrams

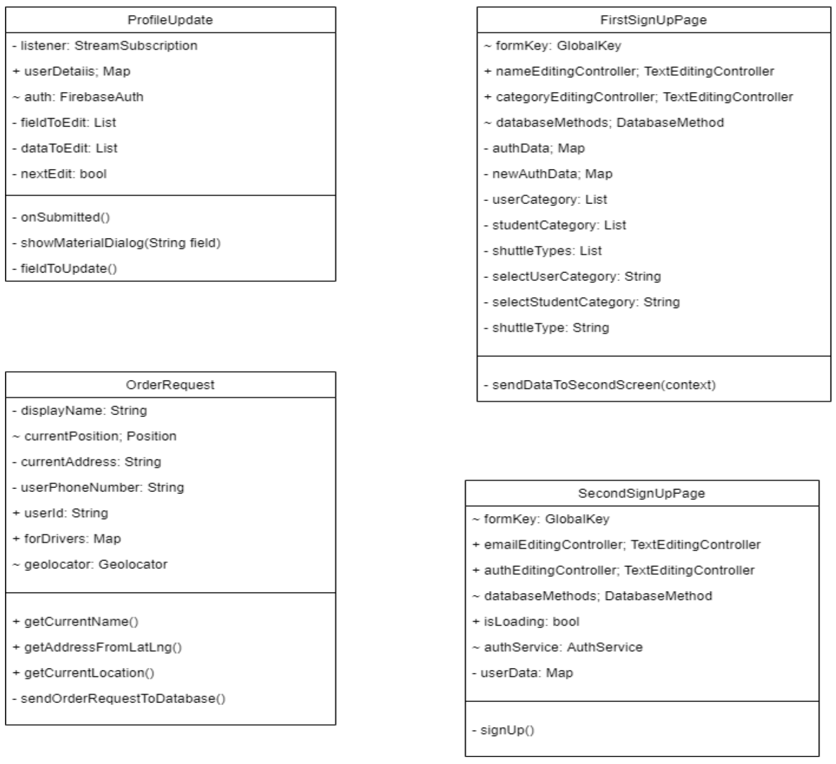


Figure 3.14: Class diagrams

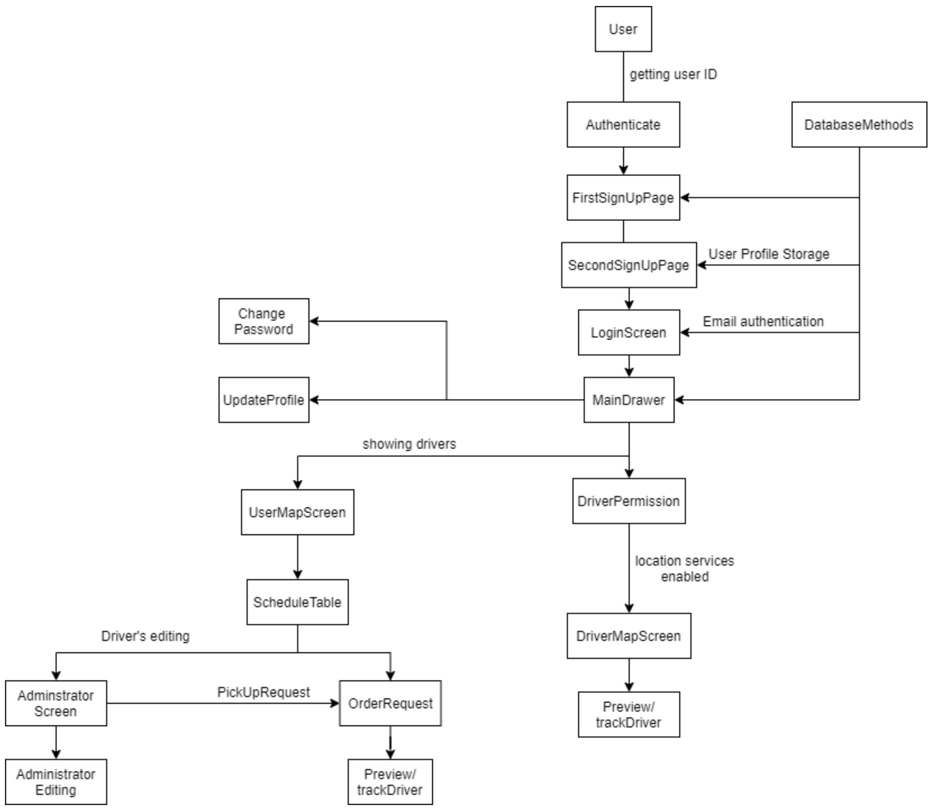


Figure 3.15: Class relationship and level

Flowchart

A flowchart is a graphical description of steps in sequential order and is usually employed in representing workflows or processes. A flowchart represents this workflow with shapes that depict a specific process and operation. The flowchart of the proposed system is shown below.

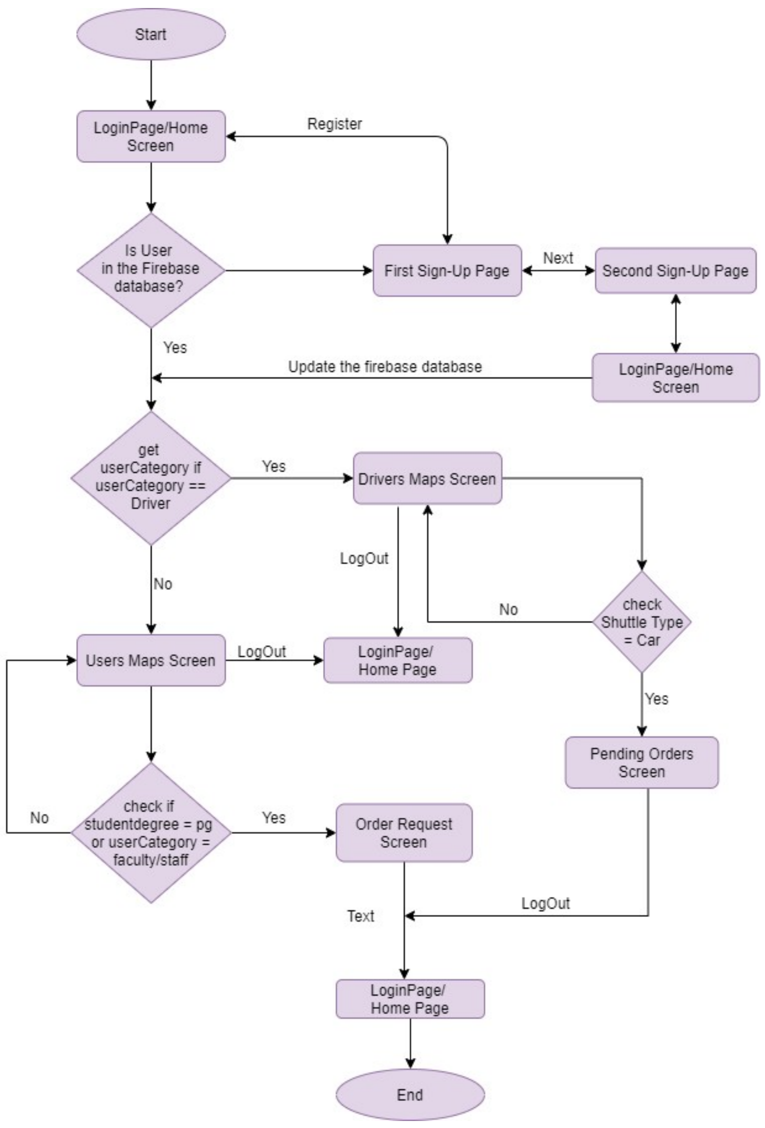


Figure 3.16: The flowchart of the proposed system

CHAPTER FOUR
IMPLEMENTATION AND TESTING

4.1 Introduction

This mobile application would enable Covenant campus user track and request shuttle pickup within the campus. Hence this chapter shows the detail implementation of the project.

4.2 Scope

As the time of writing this report, the name of this application has not yet been finalized, and for reference sake and this report, the mobile application would be referred to as FosShuttle. FosShuttle would provide a real-time location to campus users about the location of shuttle drivers without campuses. The mobile application leverages on the inbuilt GPS and accelerator in the user's devices for providing real-time information. All system information is maintained in the database firebase.

On loading the application, the user is required to sign up, and the accessibility of the screen on the application is dependent on the information provided by the users.

The mobile application has basic functionality;

- Real-time shuttle tracking and driver's estimation to the bus stop on campus,
- Pick up request,
- The messaging medium between the administrators and driver

For this project, Covenant university campus would be used. Below is the geographical mapping of the perimeter mapping of Covenant.

[(6.672595ºN, 3.151276ºE), (6.667549ºN, 3.153160ºE), (6.667208ºN, 3.155820ºE), (6.665162ºN, 3.155605ºE), (6.663712ºN, 3.161869ºE), (6.664480ºN, 3.166031ºE), (6.665247ºN, 3.166014ºE), (6.666099ºN, 3.164973ºE), (6.667346ºN, 3.163118ºE), (6.670984ºN, 3.163353ºE), (6.678610ºN, 3.162515ºE), (6.678418ºN, 3.160263ºE), (6.675093ºN, 3.160177ºE), (6.674624ºN, 3.157898ºE)].

Below is the geographical locations in latitude and longitude respectively of each terminal on Covenant campus:

- Hall of Residence Entrance (6.669351ºN, 3.156985ºE),
- Lecture Theatre (6.673729ºN, 3.159949ºE),
- CST (6.673383ºN, 3.159761ºE),
- Library (6.670124ºN, 3.157516ºE),
- Round About (6.672025ºN, 3.158854ºE),
- Post graduate Residence (6.666922ºN, 3.156356ºE),
- Hall of Residence Bus stop (6.669389ºN, 3.157147ºE),
- CDS/CBSS (6.671425ºN, 3.159114ºE),
- CDS/CBSS/Senate (6.673062ºN, 3.159773ºE),
- Cafeteria Two (6.672080ºN, 3.162251ºE),
- ALDC (6.672218ºN, 3.162643ºE),
- Department of Mechanical Engineering (6.673252ºN, 3.162510ºE),
- Department of Civil Engineering (6.674504ºN, 3.162361ºE),
- Department of Electrical and Electronics Engineering (6.675688ºN, 3.162238ºE),
- Senate Building (6.673388ºN, 3.159958ºE),
- Covenant Gate (6.677669ºN, 3.162594ºE),

4.3 Mobile application interfaces

The UI of the mobile application, their functionalities and their level of accessibility is discussed.

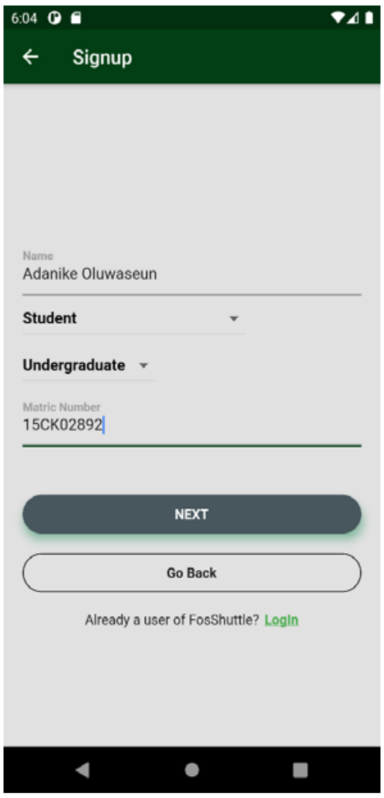


Figure 4.1: The First Sign Up Page

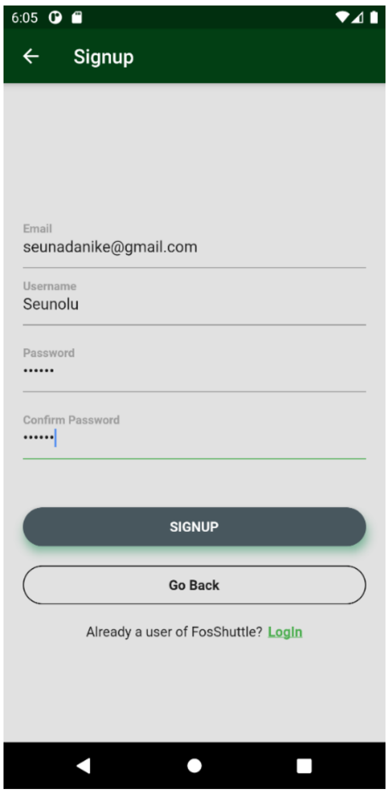
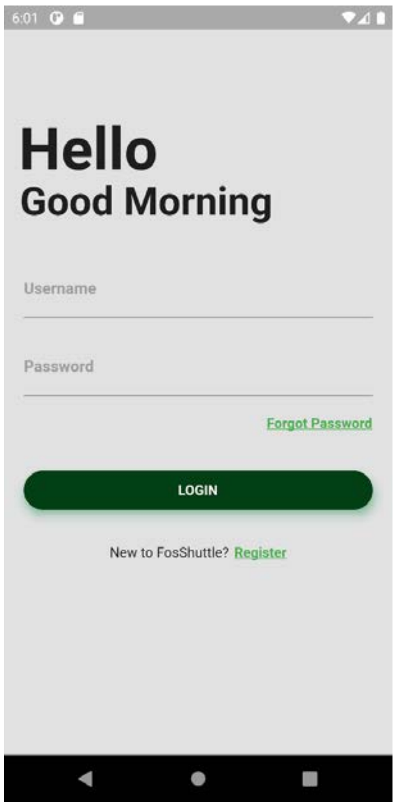


Figure 4.2: The Second Sign Up Page

Figure 4.1 is the first sign up page a new user would be required to fill to ascertain the type of user(drivers/riders, i.e. students, faculty/staff). The second sign up UI(Figure 4.2)) takes in the authentication information need for a subsequent sign-in of the users. Here the firebase authentication package was used to sign users up.



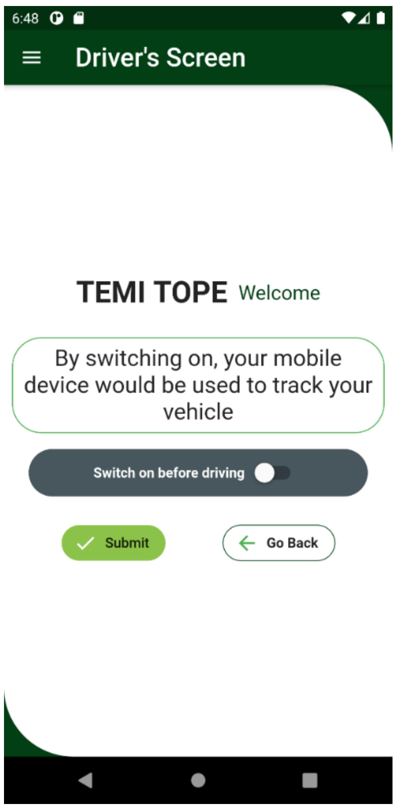


Figure 4.3: The Login Page

Figure 4.4: The driver permission

Figure 4.3 is the Login/SignIn page. It is the first page a user would see upon the opening of the application. The salutation on the login page is time-dependent. The firebase authentication services are also used here. Although the firebase authentication requires an email and password sign-in, FosShuttle would query the firebase database to get the email link to the user's username and hence for authentication. The login page is built in sync with the timer of the device, hence the salutation on the login page. Figure 4.4 is the driver's permission request screen. This is where the driver would give FosShuttle permission to use the GPS, and the accelerometer embedded his/her mobile device.

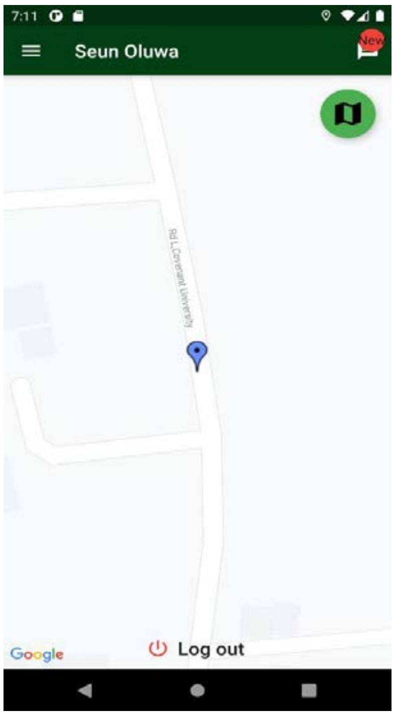


Figure 4.5: Driver Location UI

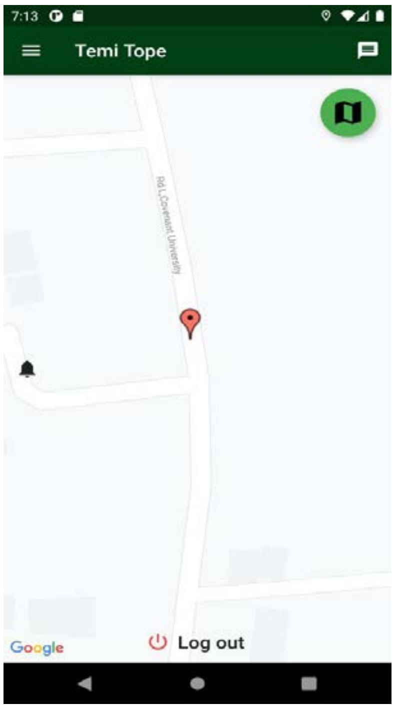
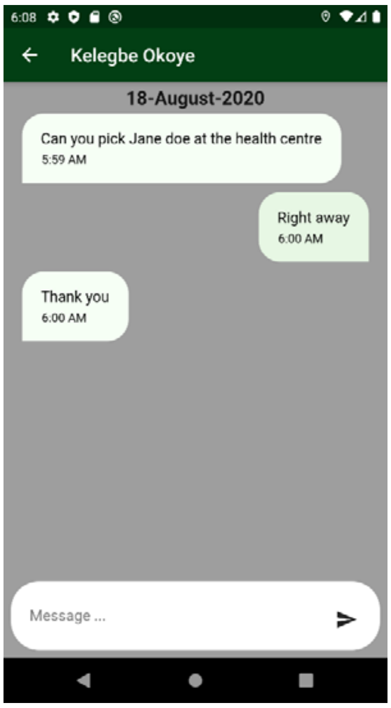


Figure 4.6: Driver Location UI(Notifier)



18-August-2020

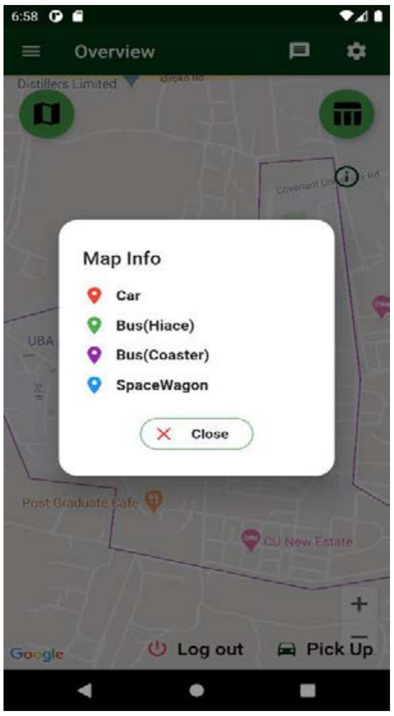
Can you pick Jane doe at the health centre
5:59 AM

Right away
6:00 AM

Thank you
6:00 AM

Message ...

Figure 4.7: Driver Messages Dialog



Map Info

- Car
- Bus(Hiace)
- Bus(Coaster)
- SpaceWagon

Close

Figure 4.8: Shuttle color indicator

Figure 4.5 shows the first screen the driver would see upon signing in into the application. Here the driver's permission (Figure 4.4) is requested before proceeding to Figure 4.5 and 4.6 where the current location of the driver is made known in real-time to all users and administrator of FosShuttle within the campus. The access level of figure 4.5 and 4.6 is the type of shuttle the driver drives. Figure 4.6 is the first screen only car shuttle drivers would see upon granting the permission request. The notifier notifies the car drives to incase of pickup request. Figure 4.7 shows the messages from the administrators once the drivers tap the message notification Icon. Figure 4.7 is the chat screen for drivers to see admin messages, and Figure 4.8 shows the user's colour guide in identifying the different types of shuttles bases on their marker colours.

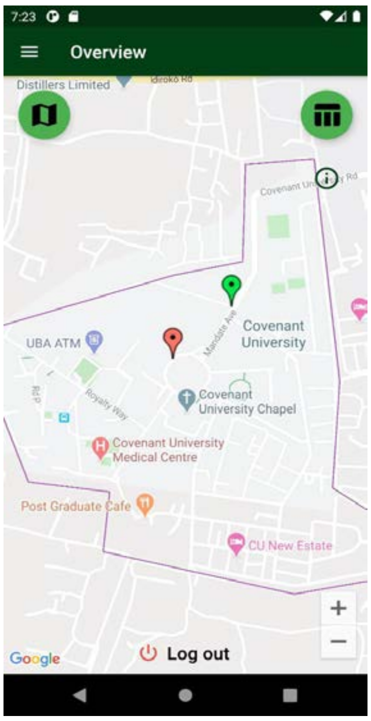


Figure 4.9: User's screen

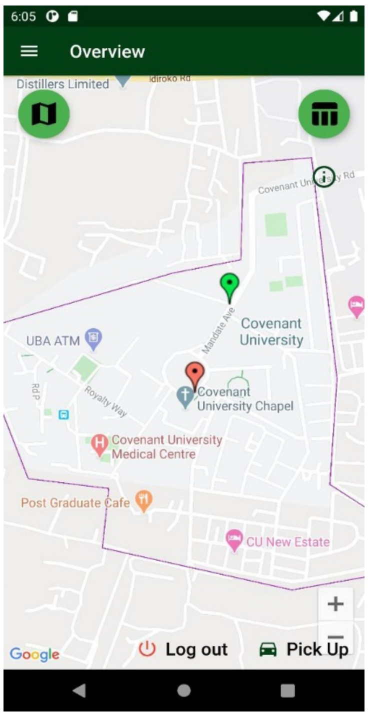
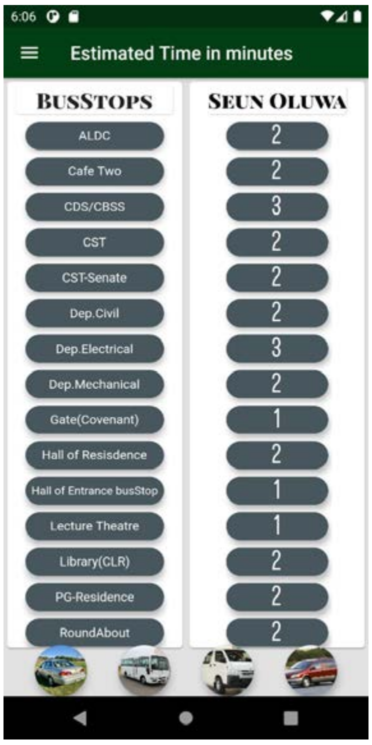


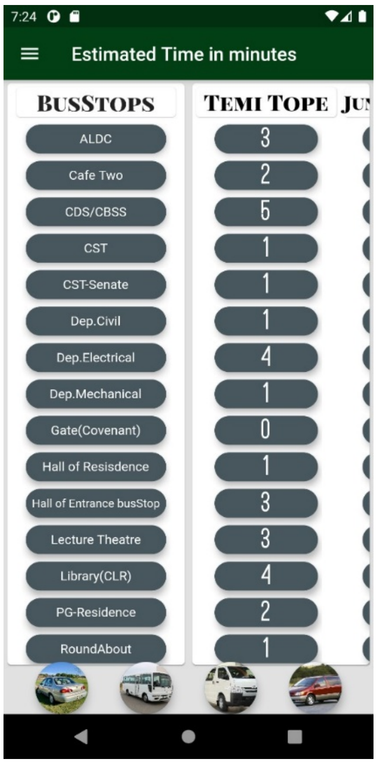
Figure 4.10: User's screen with request

Figure 4.9 and 4.10 is the User's screen showing the real-time location of shuttles on the map. Shuttles are indicated with red marks. Once a driver clicks the submit button in Figure 4.4, the current location of the driver is sync with that of the users through the database, and current location of the drivers is shown on the user's screen. Figure 4.10 is a restricted screen for undergraduate users. Figure 4.10 gives room for the users to place a request to the drivers of car shuttles as shown with the notifier in Figure 4.6.


6:06

Estimated Time in minutes

BusStops	Seun Oluwa
ALDC	2
Cafe Two	2
CDS/CBSS	3
CST	2
CST-Senate	2
Dep.Civil	2
Dep.Electrical	3
Dep.Mechanical	2
Gate(Covenant)	1
Hall of Resisdence	2
Hall of Entrance busStop	1
Lecture Theatre	1
Library(CLR)	2
PG-Residence	2
RoundAbout	2

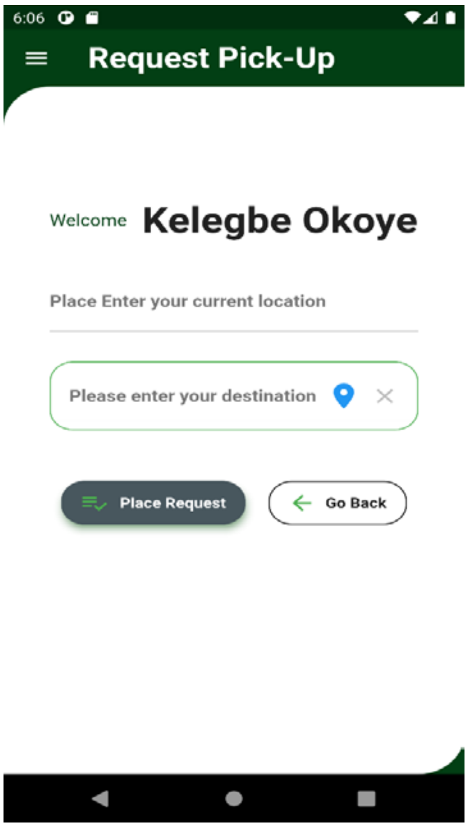

7:24

Estimated Time in minutes

BusStops	Temi Tope	Ju
ALDC	3	
Cafe Two	2	
CDS/CBSS	5	
CST	1	
CST-Senate	1	
Dep.Civil	1	
Dep.Electrical	4	
Dep.Mechanical	1	
Gate(Covenant)	0	
Hall of Resisdence	1	
Hall of Entrance busStop	3	
Lecture Theatre	3	
Library(CLR)	4	
PG-Residence	2	
RoundAbout	1	

Figure 4.11: Estimation Table Figure 4.12: Estimation Table

Figures 4.11 and 4.12 shows the estimated time a driver is expected to get to the listed bus stop. The screen contains, the driver's name, the type of shuttle he/she is driving and the proposed time he or she would get to the in-line listed scrollable bus stop. In a row, with the list, the names of the bus stop are the estimated time for the type of shuttle name above the driver's name. The users can also switch between the types of shuttles using the icon buttons. Each icon button represents a type of shuttle.


6:06
Request Pick-Up
Welcome Kelegbe Okoye
Place Enter your current location
Please enter your destination
Place Request
Go Back

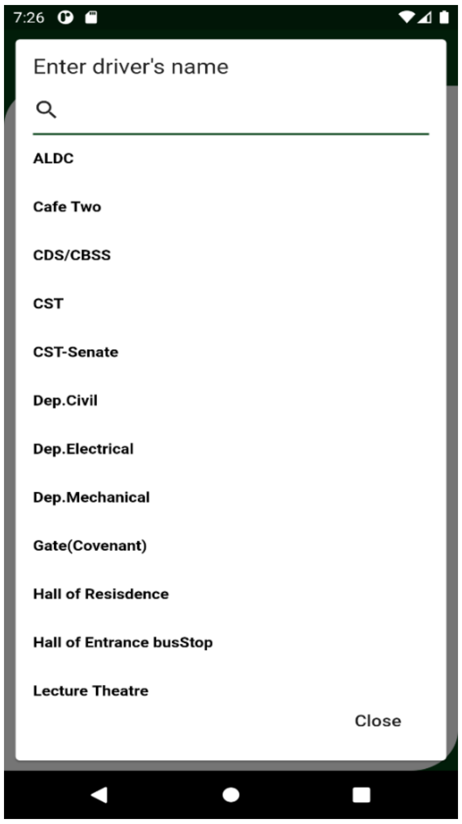


Figure 4.13: Request Page

Figure 4.14: Request Page(locations)

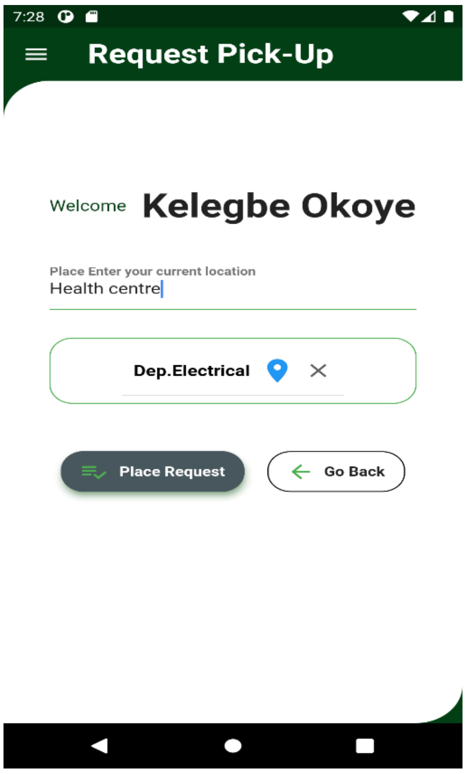
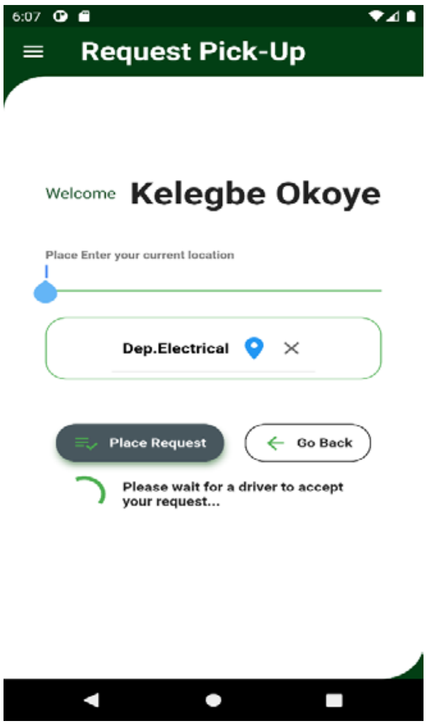


Figure 4.15: Request Page(Placing)



6:07
Request Pick-Up
Welcome
Kelegbe Okoye
Place Enter your current location
Dep.Electrical
Place Request
Go Back
Please wait for a driver to accept your request...

Figure 4.16: On placing request page

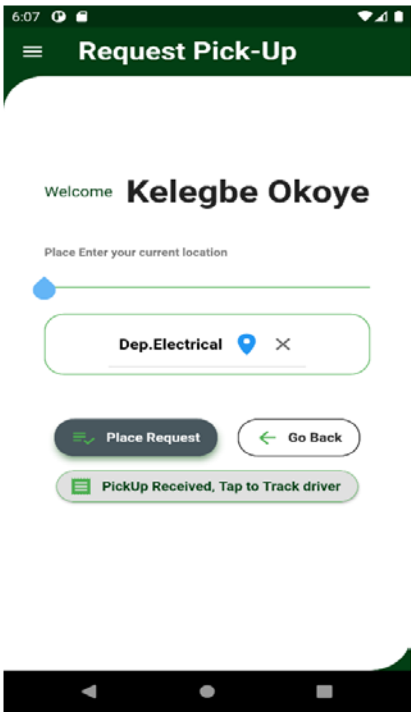
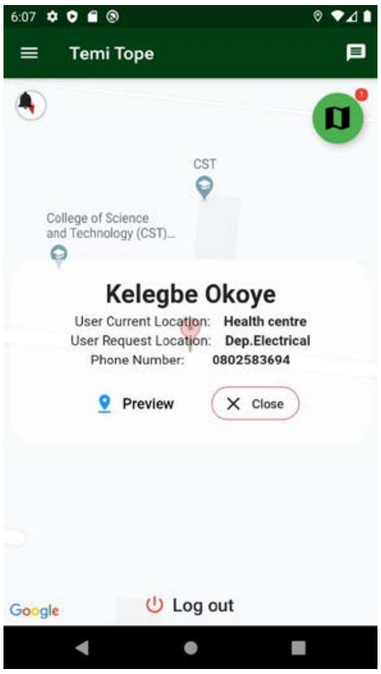


Figure 4.17: On request accept page

Figures 4.13, 4.14, 4.15, 4.16 and 4.17 shows the order screen. The user is required to enter the name of the current location through the field provided in Figure 4.13, while the current location of the use would be gotten from the GPS application in the device. Figure 4.14 shows the list of available bus stops on campus where the user is required to select where he/she is going, as shown in Figure 4.15. Once the user places the request, the database is updated, and the notifier notifies the car drivers about the pending orders. Once the user places his/her request, the user is asked to wait for a driver to accept the request (Figure 4.16). Once a driver accepts the request, the user is permitted to track this driver as the driver approaches the pickup location in (Figure 4.17).


6:07
Temi Tope
CST
College of Science and Technology (CST)...
Kelegbe Okoye
User Current Location: Health centre
User Request Location: Dep.Electrical
Phone Number: 0802583694
Preview
Close
Google
Log out

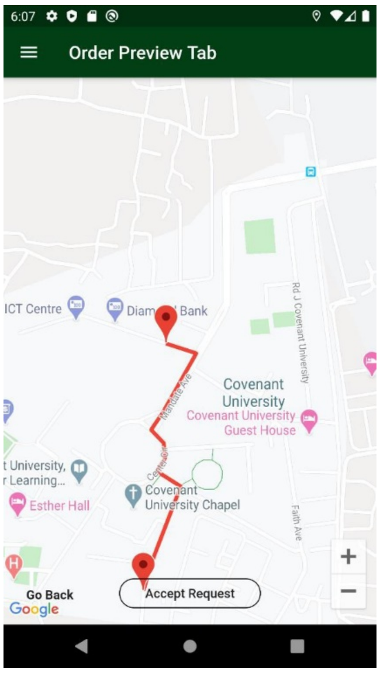


Figure 4.18: Driver's screen showing pick up requests Figure 4 19: Order Preview

Figures 4.18 and 4.19 shows what happens when the car shuttle driver clicks the notifier. A dialogue is displayed with the rider's name, current location, the proposed location and the user's phone number. Once the driver clicks preview this order in Figure 4.18, driver would be taken to Figure 4.19 where the distance between the driver and rider is shown to the driver. Once the driver accepts this request, the map screen would be shared between the rider and driver to know each other's location.

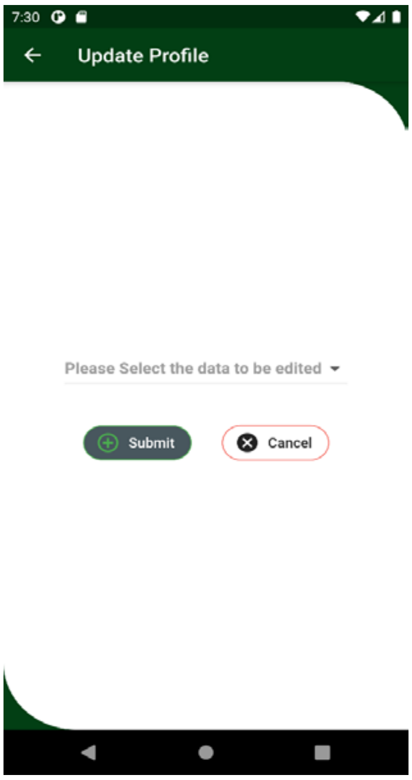


Figure 4.20: Update Profile screen

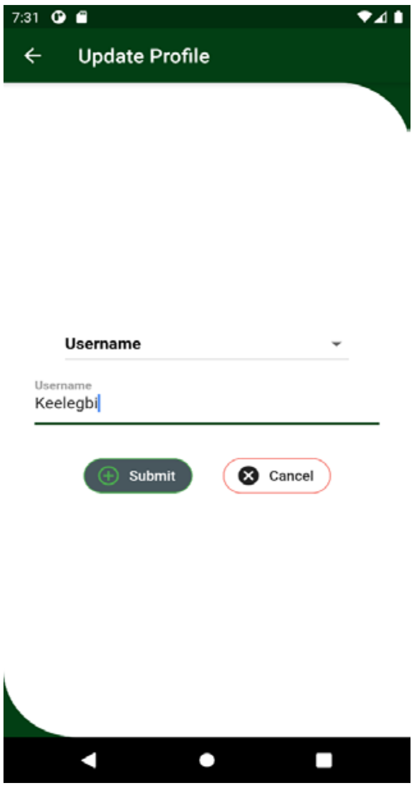
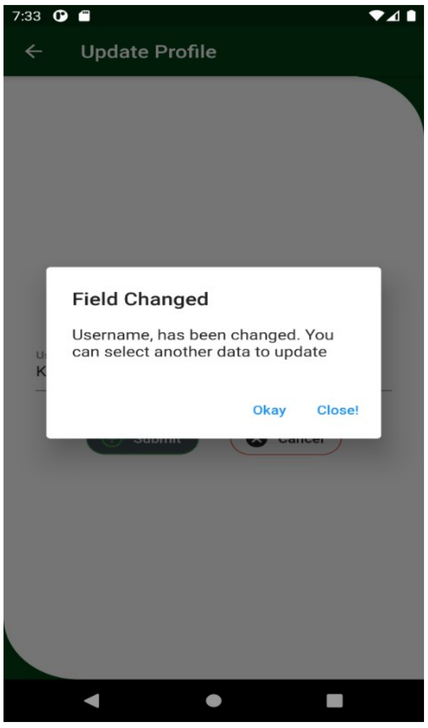


Figure 4. 21: Update Profile Screen



Field Changed

Username, has been changed. You can select another data to update

Figure 4.22: Update Profile Screen

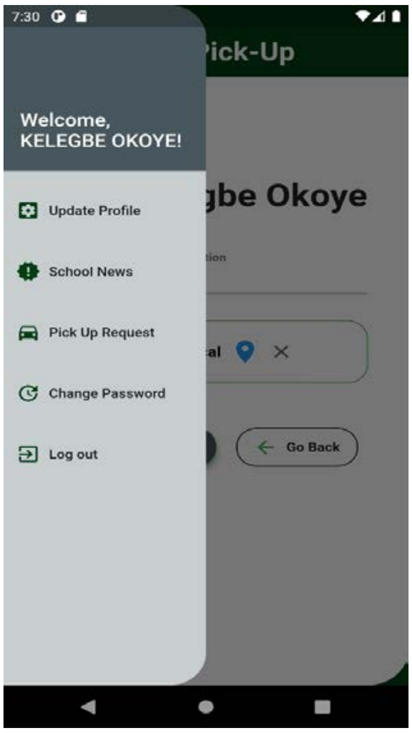


Figure 4.23: App drawer

Figures 4.20, 4.21 and 4.22 shows the user's update UIs where users can update their data.

Figure 4.23 shows the app drawer UI for quick route and navigation between the users' permitted UIs.

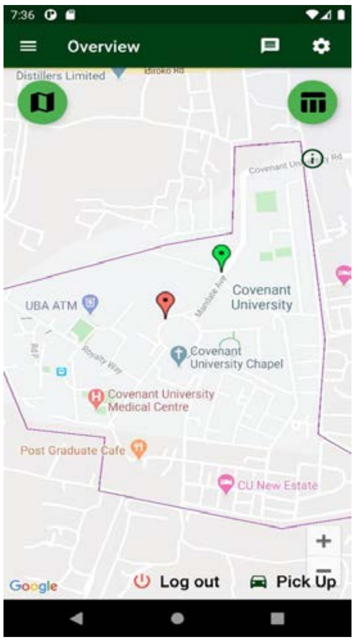
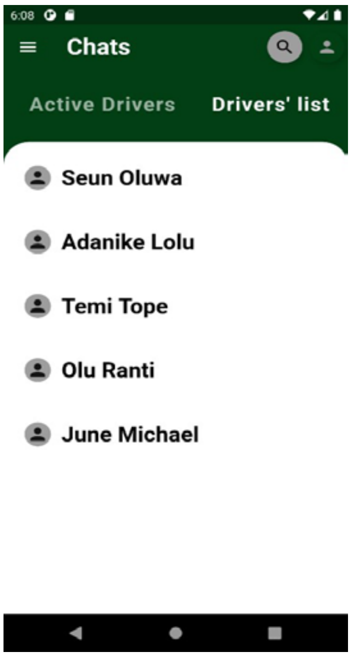
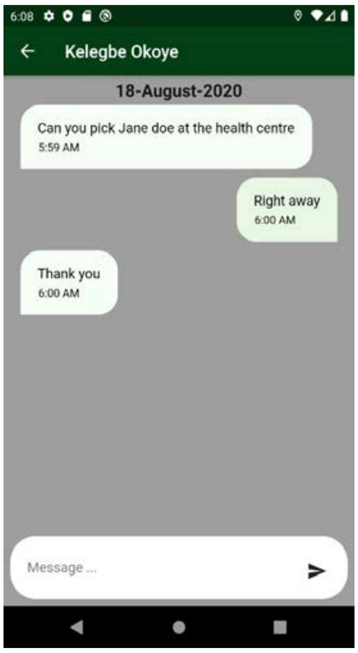


Figure 4.24: Admin Map Screen



Seun Oluwa
Adanike Lolu
Temi Tope
Olu Ranti
June Michael

Figure 4.25: Admin active drivers overview UI



18-August-2020

Can you pick Jane doe at the health centre
5:59 AM

Right away
6:00 AM

Thank you
6:00 AM

Figure 4.26: Admin Messaging UI

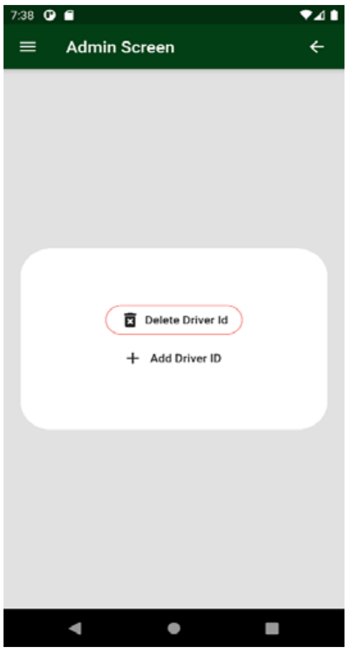
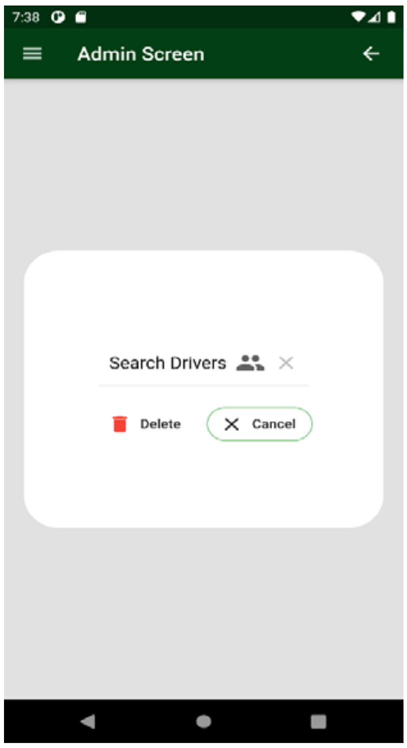


Figure 4.27: Admin-drivers update UI



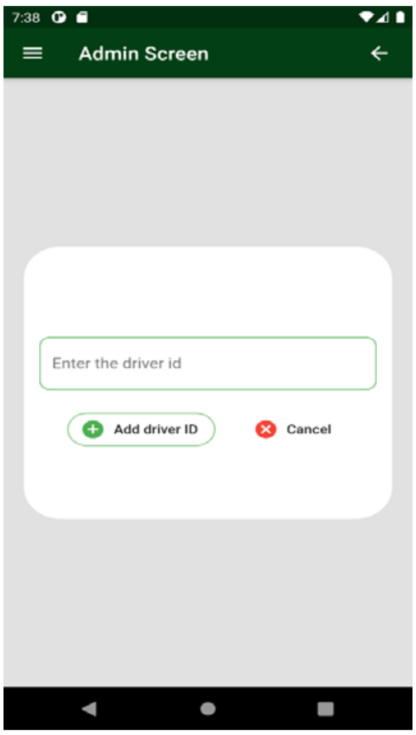
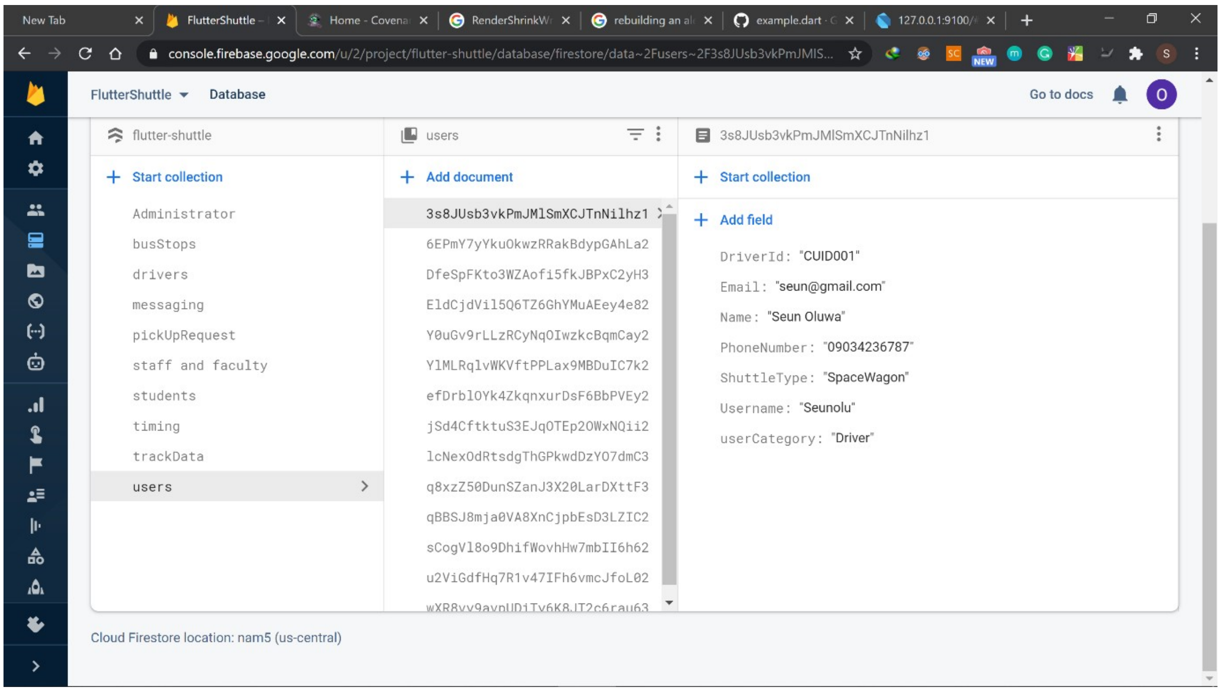


Figure 4.28: Admin Delete Driver

Figure 4.29: Admin Add Driver

Figures 4.24, 4.25, 4.26, 4.27, 4.28 and 4.29 show the administrator user interfaces. From Figure 4.24, the administrator can navigate to other admin UIs (Chat UI and driver update UI). Figure 4.25 shows the chats UI here the administrator can see his/her previous messages with the drivers, the list of active drivers, i.e. the admin can send message to any of these drivers, and also the list of all drivers in the case of a general message. Figure 4.26 shows the chat UI with the name of the recipient driver as the app bar name. Figure 4.27 is the editing UI for the administrator; here, the administrator can delete and add the IDs of drivers. For a driver to sign up, the driver's ID must first be created else the sign up will not be granted. Figure 4.26 is the message UI through which the administrator can communicate to the drivers individually on tapping their names in Figure 4.25. Figures 4.28 and 4.29 shows the add and delete widget depending on the operation the administrator taps in Figure 4.27. Figure 4.29 is the interface for which the Admin must generate an ID for the driver before the driver signs in


console.firebase.google.com/u/2/project/flutter-shuttle/database/firestore/data~2Fusers~2F3s8JUsb3vkPmJMlS...

FlutterShuttle ▾ Database

Go to docs

flutter-shuttle

+ Start collection

```
Administrator
busStops
drivers
messaging
pickUpRequest
staff and faculty
students
timing
trackData
users
```

users

+ Add document

```
3s8JUsb3vkPmJMlSmXCJTnNilhz1
6EPmY7yYkuOkwzRRakBdypGAhLa2
DfeSpFKto3WZAofi5fkJBPxC2yH3
EldCjdVil5Q6TZ6GhYMuAEey4e82
Y0uGv9rLLzRCyNqOIwzkcBqmCay2
YlMLRqlvWKVftPPLax9MBDuIC7k2
efDrblOYk4ZkqnxurDsF6BbPVEy2
jSd4CftktuS3EJqOTEp2OWxNQii2
lcNexOdRtsdgThGPkwdDzYO7dmC3
q8xzZ50DunSZanJ3X20LarDXttF3
qBBSJ8mja0VA8XnCjpbEsD3LZIC2
sCogVl8o9DhifWovhHw7mbII6h62
u2ViGdfHq7R1v47IFh6vmcJfoL02
wXR8vy9avpUDiTy6K8JT2c6rau63
```

3s8JUsb3vkPmJMlSmXCJTnNilhz1

+ Start collection

+ Add field

```
DriverId: "CUID001"
Email: "seun@gmail.com"
Name: "Seun Oluwa"
PhoneNumber: "09034236787"
ShuttleType: "SpaceWagon"
Username: "Seunolu"
userCategory: "Driver"
```

Cloud Firestore location: nam5 (us-central)

Figure 4.30: Firebase Interface

Figure 4.30 shows the firebase backend of FosShuttle, where all information relating to the application is fetched and set.

4.4 Conclusion

The purpose of the chapter was to present the software implementation process of a Free and Open Sourced Software of a campus shuttle management system. The chapter also discusses the different stacks of the screen and their level of accessibility.

CHAPTER FIVE
CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This project has provided a mobile application for campus shuttle management system. The system would grant campus users' access to real-time information about the shuttle locations. This information would help them plan their daily routine more effective and would also contribute to the punctuality of students for lectures. The project is developed based on the agile approach of software development as such; there is room for additional components and functionality.

The paper was conducted as a technical report for the implementation of a campus shuttle management system using a free and open software using Covenant University as a cases study.

Although diverse rider sharing applications such as Uber, Taxify, Lyft, hitch, Ridescout and so on exists, they are not designed for the walls of a university campus. Moreover, the relationship between the riders and drivers of this application is limit. Once a driver is booked for pick up other users can have access to the information of the driver again. The system provides access to the real-time location of all drivers of the campus shuttle system.

This system comprises of two main parts, namely, the users (drivers and users – students, faculty/staff) and administrator. The user's part consists of the synchronisation of location data between riders and drivers and requesting pickups. The administrator is responsible for the management of the shuttle drivers. The administrator can also send messages to the drivers individually about special pickups and messages.

5.2 Challenges Encountered

Some of the challenges I encountered in the course of this project include:

i. Getting real-time coordinates

ii. Firebase security

iii. Testing the mobile application on a real-time device

5.3 Limitations

Some of the factors that limit this project include

- Insufficient location data: the availability of coordinates of bus stops is minimal. As such coordinates for each bus stops had to be encoded in the firebase database for the mobile application to listen to.
- Internet availability: the functionality of the mobile application is proportional to the strength of the internet available.
- The ratio of campus users to drivers: the proportion of shuttles to campuses can be challenging due to the high demand rate from the campus users. This can put a stress shuttle system.

5.4 Recommendations

Since this project work was developed using the agile approach of software development, I would recommend that the mobile application should be made more interactive not only between the drivers and riders but also among riders. Users should be able to share information with this application since it is built for campus users. Furthermore, just like other ride-sharing applications, I would recommend payment structure be implemented in paying for services.

Appendix

https://github.com/SeunAdanike/FosShuttle

References

[1] D. Cerotti, S. Distefano, G. Merlino, and A. Puliafito, "A Crowd-Cooperative Approach for Intelligent Transportation Systems," IEEE Transactions on Intelligent Transportation Systems, vol. 18, pp. 1529-1539, 2017.

[2] S. H. Sutar, R. Koul, and R. Suryavanshi, "Integration of Smart Phone and IOT for development of smart public transportation system," in 2016 International Conference on Internet of Things and Applications (IOTA), 2016, pp. 73-78.

[3] S. H. Sutar, R. Koul, and R. Suryavanshi, "Integration of Smart Phone and IOT for development of smart public transportation system," in 2016 International Conference on Internet of Things and Applications, IOTA 2016, 2016, pp. 73-78.

[4] S. Abbas, H. Mohammed, L. Almalki, M. Hassan, and M. Meccawy, "A safety tracking and sensoring system for school buses in Saudi Arabia," Periodicals of Engineering and Natural Sciences, vol. 7, pp. 500-508, 2019.

[5] I. J. G. Zuazola, A. Moreno, H. Landaluce, I. Angulo, A. Perallos, U. Hernández-Jayo, et al., "Telematics system for the intelligent transport and distribution of medicines," IET Intelligent Transport Systems, vol. 7, pp. 131-137, 2013.

[6] S. M. Darwish, "Empowering vehicle tracking in a cluttered environment with adaptive cellular automata suitable to intelligent transportation systems," IET Intelligent Transport Systems, vol. 11, pp. 84-91, 2017.

[7] R. Jabangwe, H. Edison, and A. N. Duc, "Software engineering process models for mobile app development: A systematic literature review," Journal of Systems and Software, vol. 145, pp. 98-111, 2018/11/01/ 2018.

[8] J. Bai, W. Wang, Y. Qin, S. Zhang, J. Wang, and Y. Pan, "BridgeTaint: A Bi-Directional Dynamic Taint Tracking Approach for JavaScript Bridges in Android

Hybrid Applications," IEEE Transactions on Information Forensics and Security, vol. 14, pp. 677-692, 2019.

[9] F. Al-Turjman, "5G-enabled devices and smart-spaces in social-IoT: An overview,"Future Generation Computer Systems, vol. 92, pp. 732-744, 2019/03/01/ 2019. [10] D. B. Silva, M. M. Eler, V. H. S. Durelli, and A. T. Endo, "Characterizing mobileapps from a source and test code viewpoint," Information and Software Technology, vol. 101, pp. 32-50, 2018/09/01/ 2018.

[11] S. S. Sagar Tete, Diksha Likhar, Reshma Badalu, "ANDROID APP: VEHICLE TRACKING SYSTEM," International Research Journal of Engineering and Technology (IRJET), vol. Volume: 05, Feb-2018.

[12] M. E. E. a. I. E. O. Marvel L. Akinyemi, "Natural Approach towards Mitigating Noise Pollution: A Case Study of Covenant University," International Journal of Scientific & Engineering Research, vol. Volume 6, p. 3, May-2015.

[13] G. Andrienko, N. Andrienko, W. Chen, R. Maciejewski, and Y. Zhao, "Visual analytics of mobility and transportation: State of the art and further research directions," IEEE Transactions on Intelligent Transportation Systems, vol. 18, pp. 2232-2249, 2017.

[14] M. G. Demissie, S. Phithakkitnukoon, T. Sukhvibul, F. Antunes, R. Gomes, and C. Bento, "Inferring Passenger Travel Demand to Improve Urban Mobility in Developing Countries Using Cell Phone Data: A Case Study of Senegal," IEEE Transactions on Intelligent Transportation Systems, vol. 17, pp. 2466-2478, 2016.

[15] Z. Cai, F. Ren, J. Chen, and Z. Ding, "Vector-Based Trajectory Storage and Query for Intelligent Transport System," IEEE Transactions on Intelligent Transportation Systems, vol. 19, pp. 1508-1519, 2018.

[16] H. I. Ashqar, M. H. Almannaa, M. Elhenawy, H. A. Rakha, and L. House, "Smartphone transportation mode recognition using a hierarchical machine learning classifier and pooled features from time and frequency domains," IEEE Transactions on Intelligent Transportation Systems, vol. 20, pp. 244-252, 2019.

[17] S. Sharif, M. Suhaimi, N. Jamal, I. Riadz, I. Amran, and D. Jawawi, Real-Time Campus University Bus Tracking Mobile Application, 2018.

[18] L. Gherardi, D. Hunziker, and G. Mohanarajah, "A Software Product Line Approach for Configuring Cloud Robotics Applications," in 2014 IEEE 7th International Conference on Cloud Computing, 2014, pp. 745-752.

[19] T. Hadwen, V. Smallbon, Q. Zhang, and M. D. Souza, "Energy efficient LoRa GPS tracker for dementia patients," in 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2017, pp. 771-774.

[20] T. Ojo, R. Amoaako-Sakyi, and W. Agyeman, "Students' satisfaction of campus shuttling services: a Qualbus approach," Management Research and Practice, 01/01 2015.

[21] F. Zhang, Q. Shen, and K. J. Clifton, "Examination of Traveler Responses to Real-Time Information about Bus Arrivals using Panel Data," Transportation Research Record, vol. 2082, pp. 107-115, 2008/01/01 2008.

[22] M. Vanderschuren and D. d. Vries, "Advanced public transportation information provision: What are the effects on improved customer satisfaction?," in 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), 2013, pp. 499-504.

[23] "ISO/IEC/IEEE International Standard - Systems and software engineering -- Vocabulary," ISO/IEC/IEEE 24765:2010(E), pp. 1-418, 2010.

[24] G. Aceto, V. Persico, and A. Pescapé, "A Survey on Information and Communication Technologies for Industry 4.0: State-of-the-Art, Taxonomies, Perspectives, and Challenges," IEEE Communications Surveys & Tutorials, vol. 21, pp. 3467-3501, 2019.

[25] M. Välimäki and V. Oksanen, "The impact of free and open source licensing on operating system software markets," Telematics and Informatics, vol. 22, pp. 97-110, 2005/02/01/ 2005.

[26] P. A. Atayero, "FOSS IMPLEMENTATION OF AN EDUCATIONAL VIRTUAL OFFICE SUITE," Proceedings of EDULEARN14 Conference 7th-9th July 2014, Barcelona, Spain, 07/01 2014.

[27] J. Gediya, J. Singh, P. Kushwaha, R. Srivastava, and Z. Wang, "7 - Open-Source Software," in Software Engineering for Embedded Systems (Second Edition), R. Oshana and M. Kraeling, Eds., ed: Newnes, 2019, pp. 207-244.

[28] L. Pisha, J. Warchall, T. Zubatiy, S. Hamilton, C. Lee, G. Chockalingam, et al., "A Wearable, Extensible, Open-Source Platform for Hearing Healthcare Research," IEEE Access, vol. 7, pp. 162083-162101, 2019.

[29] G. Aimicheva, Z. Kopeyev, Z. Ordabayeva, N. Tokzhigitova, and S. Akimova, "A spiral model teaching mobile application development in terms of the continuity principle in school and university education," Education and Information Technologies, 2019.

[30] W. S. El-Kassas, B. A. Abdullah, A. H. Yousef, and A. M. Wahba, "Enhanced Code Conversion Approach for the Integrated Cross-Platform Mobile Development (ICPMD)," IEEE Transactions on Software Engineering, vol. 42, pp. 1036-1053, 2016.

[31] T. Kim, B. Kim, and J. Kim, "Development of a lever learning Webapp for an

HTML5-based cross-platform," in Lecture Notes in Electrical Engineering vol. 240 LNEE, ed, 2013, pp. 313-320.

[32] W. El-Kassas, A. Solyman, and M. Farouk, "MTourism multilingual integrated solution: A case study EgyptTravel," in eChallenges Conference, 2015.

[33] P. J. Lin, C. C. Kao, K. H. Lam, and I. C. Tsai, "Design and implementation of a tourism system using mobile augmented reality and GIS technologies," in Lecture Notes in Electrical Engineering vol. 293, ed, 2014, pp. 1093-1099.

[34] Y. Ma, X. Liu, Y. Liu, Y. Liu, and G. Huang, "A Tale of Two Fashions: An Empirical Study on the Performance of Native Apps and Web Apps on Android," IEEE Transactions on Mobile Computing, vol. 17, pp. 990-1003, 2018.

[35] B. Soewito, F. E. Gunawan, and I. P. Rusli, "The use of android smart phones as a tool for absences," Procedia Computer Science, vol. 157, pp. 238-246, 2019/01/01/ 2019.

[36] M. Huynh and P. Ghimire, "Browser app approach: Can it be an answer to the challenges in cross-platform app development?," Journal of Information Technology Education: Innovations in Practice, vol. 16, pp. 47-68, 2017.

[37] A. Biørn-Hansen, T. M. Grønli, G. Ghinea, and S. Alouneh, "An Empirical Study of Cross-Platform Mobile Development in Industry," Wireless Communications and Mobile Computing, vol. 2019, 2019.

[38] S. Gupta, "Addressing the Issues in Mobile Application Development," International Journal of Computer Sciences and Engineering, vol. Volume-2, , 2014.

[39] B. A. M. Hammoudeh S. Alamri, "Software Engineering Challenges in Multi Platform Mobile Application Development," Faculty of Computer Systems and

Software Engineering, Universiti Malaysia Pahang, Pahang, Malaysia.

[40] G. H. G. S. A. DeGani, "Cross-platform mobile development," Tribal: Medical Mobile Development Project: D4, Mobile Learning Environment, March 2011.

[41] M. Latif, Y. Lakhrissi, E. H. Nfaoui, and N. Es-Sbai, Cross platform approach for mobile application development: A survey, 2016.

[42] I. T. Mercado, N. Munaiah, and A. Meneely, "The impact of cross-platform development approaches for mobile applications from the user's perspective," presented at the Proceedings of the International Workshop on App Market Analytics, Seattle, WA, USA, 2016.

[43] H. S. Heitkötter H., Majchrzak T.A. , "Evaluating Cross-Platform Development Approaches for Mobile Applications.," In: Cordeiro J., Krempels KH. (eds) Web Information Systems and Technologies., vol. WEBIST 2012. Lecture Notes in Business Information Processing, vol 140., 2013

[44] I. Mercado, N. Munaiah, and A. Meneely, The impact of cross-platform development approaches for mobile applications from the user's perspective, 2016.

[45] A. Smith, K. de Salas, I. Lewis, and B. Schüz, "Developing Smartphone Apps for Behavioural Studies: The AlcoRisk App Case Study," Journal of Biomedical Informatics, vol. 72, p. 108, 07/01 2017.

[46] J. S. Rafael Aldrete-Sanchez, and Dr. Ruey Long Cheu, P.E., "INTEGRATING THE TRANSPORTATION SYSTEM WITH A UNIVERSITY CAMPUS TRANSPORTATION MASTER PLAN: A CASE STUDY," Texas Transportation Institute The Texas A&M University System College Station, Texas 77843-3135 September 2009.

[47] Ö. Göçer and K. Göçer, "The effects of transportation modes on campus use: A

case study of a suburban campus," Case Studies on Transport Policy, vol. 7, pp. 37-47, 2019/03/01/ 2019.

[48] M. J. Binder, J. Martin, and J. K. Schwind, "Exploring mindfulness in teaching-learning scholarship through a reflective conversation," in Mindfulness in the Academy: Practices and Perspectives from Scholars, ed, 2018, pp. 83-96.

[49] M. Cattaneo, P. Malighetti, C. Morlotti, and S. Paleari, "Students' mobility attitudes and sustainable transport mode choice," International Journal of Sustainability in Higher Education, vol. 19, 07/19 2018.

[50] U. I. S. T. W. Group, "Student Transportation and Educational Access," February 2017

[51] T. Federici, A. M. Braccini, V. Albano, and E. D'Atri, Intelligent Transport Systems: how to Manage a Research in a new Field in IS, 2011.

[52] M. H. Rahman, & Hoque, M. S., "Intelligent Transport Systems and its Prospects in the Contextof Dhaka City," Proceedings of 10th International Conference on Application of AdvancedTechnologies in Transportation (AATT), May 28-30, Athens, Greece. 2008.

[53] N. Khademi, "“Travel Time Cognition: Exploring the Impacts of Travel Information Provision Strategies.” ,," Travel Behaviour and Society, 2019. .

[54] S. amamou, Z. trifa, and M. khmakhem, "Data protection in cloud computing: A Survey of the State-of-Art," Procedia Computer Science, vol. 159, pp. 155-161, 2019/01/01/ 2019.

[55] R. S. Gargees and G. J. Scott, "Dynamically Scalable Distributed Virtual Framework Based on Agents and Pub/Sub Pattern for IoT Media Data," IEEE Internet of Things Journal, vol. 6, pp. 599-613, 2019.

[56] S. Kuyoro, F. Ibikunle, and A. Oludele, "Cloud Computing Security Issues and

Challenges," International Journal of Computer Networks (IJCN), vol. 3, pp. 247- 255, 12/01 2011.

[57] B. M. B. a. S. Prasad, "Challenges and Benefits of Deploying Big Data Analytics in the Cloud for Business Intelligence," International Conference on Knowledge Based and Intelligent Information and Engineering, 6-8 September 2017.

[58] D. Mukhopadhyay, G. Sonawane, P. Gupta, S. Bhavsar, and V. Mittal, "Enhanced Security for Cloud Storage using File Encryption," 03/01 2013.

[59] Z. Kartit and M. El marraki, "Applying Encryption Algorithm to Enhance Data Security in Cloud Storage," Engineering Letters, vol. 23, pp. 277-282, 11/17 2015.

[60] T. Kamalakannan, K. S. Senthil, C. Shanthi, and D. Radhakrishnan, "Study on Cloud Storage and its Issues in Cloud Computing," 06/10 2019.

[61] W. Ahmed and Y. Wu, "Estimation of cloud node acquisition," Tsinghua Science and Technology, vol. 19, pp. 1-12, 2014.

[62] S. Akintoye, A. Bagula, Y. Djemaiel, and N. Bouriga, "A Survey on Storage Techniques in Cloud Computing," International Journal of Computer Applications, vol. 163, pp. 22-30, 04/17 2017.

[63] R. R. Expósito, G. L. Taboada, S. Ramos, J. Touriño, and R. Doallo, "Performance Evaluation of Data-Intensive Computing Applications on a Public IaaS Cloud," The Computer Journal, vol. 59, pp. 287-307, 2016.

[64] D. Chou and A. Chou, "Software as a Service (SaaS) as an outsourcing model: An economic analysis," 01/01 2008.

[65] F. Liu, W. Guo, Z. Q. Zhao, and W. Chou, "SaaS Integration for Software Cloud," in 2010 IEEE 3rd International Conference on Cloud Computing, 2010, pp. 402-409. [66] M. Akter, A. Gani, M. O. Rahman, M. M. Hassan, A.

Almogren, and S. Ahmad,"Performance Analysis of Personal Cloud Storage Services for Mobile Multimedia Health Record Management," IEEE Access, vol. 6, pp. 52625-52638, 2018.

[67] R. Arokia, A. Paul Rajan, and S. Shanmugapriyaa, "Evolution of Cloud Storage as Cloud Computing Infrastructure Service," IOSR, vol. 1, pp. 2278-661, 06/10 2012.

[68] H. Idrissi, K. Ali, and M. El marraki, "FOREMOST SECURITY APPREHENSIONS IN CLOUD COMPUTING," ed, 2014.

[69] M. Factor, K. Meth, D. Naor, O. Rodeh, and J. Satran, Object storage: The future building block for storage systems, 2005.

[70] R. Agrawal and C. Nyamful, "Challenges of big data storage and management,"Global Journal of Information Technology, vol. 6, 03/15 2016.

[71] S. Kim, G. Gwon, W. Hur, D. Hyeon, D. Kim, S. Kim, et al., "Autonomous Campus Mobility Services Using Driverless Taxi," IEEE Transactions on Intelligent Transportation Systems, vol. 18, pp. 3513-3526, 2017.

[72] M. Zhu, X. Liu, F. Tang, M. Qiu, R. Shen, W. Shu, et al., "Public Vehicles for Future Urban Transportation," IEEE Transactions on Intelligent Transportation Systems, vol. 17, pp. 3344-3353, 2016.

[73] S.-A. Kaye, L. Buckley, A. Rakotonirainy, and P. Delhomme, "An adaptive approach for trialling fully automated vehicles in Queensland Australia: A brief report," Transport Policy, vol. 81, pp. 275-281, 2019/09/01/ 2019.

[74] P. Zhou, Y. Zheng, and M. Li, "How Long to Wait? Predicting Bus Arrival Time With Mobile Phone Based Participatory Sensing," IEEE Transactions on Mobile Computing, vol. 13, pp. 1228-1241, 2014.

[75] S. R. Nalawade and S. D. Akshay, "Bus tracking by computing cell tower

information on Raspberry Pi," in 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), 2016, pp.87-90.

[76] S. Almishari, N. Ababtein, P. Dash, and K. Naik, "An energy efficient real-time vehicle tracking system," in 2017 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), 2017, pp. 1-6.