



# **ASHESI UNIVERSITY COLLEGE**

**MOBILE TECHNOLOGY IN TRANSPORTATION: A CASHLESS MEANS  
FOR TRANSPORTATION PAYMENT USING NFC AND QR CODES IN  
GHANA**

**APPLIED PROJECT**

B.Sc. Computer Science

**Leon Ampah**

**2018**

# **ASHESI UNIVERSITY COLLEGE**

## **Mobile Technology in Transportation: A Cashless Means for Transportation Payment Using NFC And QR Codes**

### **APPLIED PROJECT**

Applied Project Submitted to the Department of Computer Science, Ashesi  
University College in partial fulfilment of the requirement for the award of the  
Bachelor of Science Degree in Computer Science

Leon Ampah

April 2018

## DECLARATION

I hereby declare that this Applied Project is the result of my own original work and that no part of

it has been presented for another degree in this university or elsewhere.

Candidate's Signature:

.....

Candidate's Name:

.....

Date: .....

I hereby declare that preparation and presentation of this Applied Project were supervised in accordance with the guidelines on supervision of Applied Project laid down by Ashesi University

College.

Supervisor's Signature:

.....

Supervisor's Name:

.....

Date: .....

## **Acknowledgement**

To my supervisor, PhD. Amanquah, the Computer Science Department of Ashesi University and the Arts and Science Department especially, Mrs. Rebecca Awuah, Mr. Joseph Agyapong Mensah and Mr. Eli Tetteh. I am very grateful for your lives and I am thankful for all the time and patience you gave and knowledge you have all imparted in me throughout my four-year experience here at Ashesi.

## **Abstract**

Over the years, many means of cashless payments for goods and services have been implemented in Ghana. Although these systems claim diversity given the vast number of services that can be paid for on their respective platforms, they lack an avenue for the payment of transportation. This project attempts to fill this gap by developing a means of paying for transportation in a cashless fashion. In this project, technology such as NFC, QR codes and smartphones is employed to implement a cashless payment solution for transportation that will be a handy tool and asset to the average Ghanaian. At the end of this project, a thoroughly tested cross-platform mobile application along with a web-based application was created. The mobile application enables a public transport commuter to pay transport fares in a cashless manner via QR codes and NFC. It also allows a vehicle owner to monitor transactions made by his/her vehicle. The web-based application on the other hand enables a system status administrator to monitor the system and a transport credit retailer to sell transport credits.



## Table of Contents

Chapter 1: Introduction .....	1
1.1 Introduction .....	1
1.2 Motivation .....	2
1.3 Problem Statement .....	2
1.4 Benefits of Proposed System .....	2
1.5 Project Objectives .....	3
1.6 Overview of Chapter .....	4
Chapter 2: Background and Related Work .....	5
2.1 Background .....	5
2.1.1 NFC .....	5
2.1.2 QR Code .....	7
2.3 Relevance of Technologies .....	8
2.4 Chapter Overview .....	9
CHAPTER 3: REQUIREMENTS SPECIFICATION .....	10
3.1 Project Scope .....	10
3.2 Overall Description .....	10
3.2.1 Product Perspective .....	10
3.2.2 User Descriptions and Characteristics .....	11
3.2.3 Operating Environment .....	12
3.3 Use-case Scenarios .....	12
3.3.1 Public Transport User Cases .....	13
3.3.2 Vehicle Owner Use Case .....	13
3.3 Product Features .....	15
3.3.1.1 Principal Features .....	15
3.3.1.2 Supplementary Features .....	16
Chapter 4: Architecture and Design .....	19
4.1 High-level Overview of System .....	19
4.2 REST Service .....	20
4.3 Mobile Application .....	21
4.4 Web Application .....	22
4.5 Overview of Chapter .....	23
Chapter 5: Implementation .....	24

5.1 REST API Implementation .....	24
5.2 Web Application .....	30
5.4 Chapter Overview .....	34
Chapter 6: Testing and Results .....	35
6.1 Component Testing .....	35
6.2 System-Level Testing .....	35
6.3 User Testing .....	36
6.4 Chapter Overview .....	37
Chapter 7: Conclusion and Recommendations .....	38
7.1 Summary .....	38
7.2 Limitations and Challenges.....	38
7.3 Future Work .....	39
7.4 Conclusion .....	40



## Table of Figures

<b>Figure 2-1 NFC interaction styles and operating modes .....</b>	<b>6</b>
<b>Figure 3-0-1 High-level Overview of subsystems and Components of the Xipora Cashless Transportation Payment Platform .....</b>	<b>11</b>
<b>Figure 3-0-2 Use case Diagram for public transport user.....</b>	<b>14</b>
<b>Figure 3-0-3 Use case diagram for vehicle owner .....</b>	<b>15</b>
<b>Figure 3-0-4 Activity diagram for mobile application.....</b>	<b>17</b>
<b>Figure 3-0-5 Activity Diagram for Web application.....</b>	<b>18</b>
<b>Figure 4-0-0-1 High-level Overview of System.....</b>	<b>19</b>
<b>Figure 4-0-2 Architecture of REST API .....</b>	<b>21</b>
<b>Figure 4-0-3 Architecture of Mobile Application .....</b>	<b>22</b>
<b>Figure 4-0-0-4 Architecture of Web Application .....</b>	<b>23</b>
<b>Figure 5-0-1 Configurations in application.properties .....</b>	<b>25</b>
<b>Figure 5-0-2 MongoConfig.java file. ....</b>	<b>25</b>
<b>Figure 5-0-3 Snippet of User Document class.....</b>	<b>26</b>
<b>Figure 5-0-4 Snippet of UserRepository Interface .....</b>	<b>26</b>
<b>Figure 5-0-5 Snippet of UserService class.....</b>	<b>27</b>
<b>Figure 5-0-6 UserForm.java file representing a view .....</b>	<b>28</b>
<b>Figure 5-0-7 AdminLogic.java file snippet .....</b>	<b>29</b>
<b>Figure 5-0-8 AdminController.java file snippet.....</b>	<b>30</b>
<b>Figure 5-0-9 Screenshot of the system status admin’s dashboard.....</b>	<b>31</b>
<b>Figure 5-0-10 Screenshot the transactions breakdown .....</b>	<b>31</b>
<b>Figure 5-0-11 Screenshot of transaction creation form.....</b>	<b>32</b>

<b>Figure 5-0-12 Screenshot of Dashboard page.....</b>	<b>32</b>
<b>Figure 5-0-13 Homepage screenshot .....</b>	<b>33</b>
<b>Figure 5-0-14 Search Page.....</b>	<b>33</b>
<b>Figure 5-1-5 Vehicle owner Search page .....</b>	<b>34</b>
<b>Figure 5-0-16 Vehicle owner Homepage Screenshot .....</b>	<b>34</b>



## **Chapter 1: Introduction**

### **1.1 Introduction**

Ghana over the years has experienced a vast amount of growth in various sectors of its economy such as health, education and finance. One of the main driving forces of this growth is the Technology Industry and the speed at which it is being integrated in the lives of Ghanaians. This is not a surprising fact because all developed countries in one way or another made use of technology to get to where they are today. Some common qualities that persists in these developed countries are cashless payment and cashless society. An example of a country that exhibits the idea of a cashless system is Sweden where 95% of transactions are digital (Gjerding, 2017 ). This then begs the question what is a cashless society? A cashless society can be defined as a community in which payments such as bills and debits are dealt with electronically (or digitally) (Akinola, 2012).

Ghana, in her own way has made various attempts towards a cashless economy. The first notable attempt was the E-Zwich card. Nevertheless, this attempt failed miserably due to issues such as link failure, frequent breakdown of the hardware of the system (the machine) and delayed transactions. (Issahaku, 2012). Although it may be true that due to the popularity of smartphones, mobile and web applications such as Slydepay, ExpressPay, ZeePay etc. Ghana is back on track towards a cashless economy. However, owing to the fact that these cashless mediums lack an avenue for transportation payment the ideal of a cashless economy is beyond our reach.

This project attempts to push the ideal of a cashless economy back within the reach of Ghana by providing a smart medium for the payment of transportation services using mobile

technology. This system will initially focus on making smart payment on a popular mode of transport in Ghana known as ‘tro-tro’.

## **1.2 Motivation**

Due to the increased popularity of smart phones, mobile applications and increased access to the internet among other factors, sectors of the Ghanaian economy such as health, education, finance have experienced considerable amounts of growth. Hence this project draws its motivation from the need to bring growth and prosperity to the transportation sector by making the day to day transportation transactions of the average Ghanaian a seamless and simple experience.

## **1.3 Problem Statement**

In Ghana there are systems that seek to make payments in the transportation easy such as Ayalolo bus card. Regrettably, this system requires that users register and pay for a card in addition to the identification cards and bank cards they already possess. Technologies such as NFC and cameras which are very prevalent in a wide range of smart phones can be used and therefore eliminate the need to pay and acquire an additional card for your wallet or purse. However, due to the fear many Ghanaians would easily not accept change and the lack NFC technologies usage among developers in Ghana, solutions to this problem are yet to come by.

## **1.4 Benefits of Proposed System**

A cashless system designed for the payment of transportation using NFC and QR codes will make payments for transportation extremely easy, fast and seamless. In addition to the convenience it brings, a cashless medium of payment for transportation would also aid in stabilizing transportation costs during transit by making it such that changes in rate would

occur via the platform as opposed to the arbitrary setting of rates by drivers and mates in different vehicles. Another added benefit of having a cashless medium of payment for transportation is that it will go a long way to reduce the cost of between thirty to thirty five million dollars (Ghana Web, 2012) incurred by the government during the printing of bills hence, instead the money would go into different sectors of the economy thus speeding up the development process of the nation. Another benefit of the proposed system is since transactions would be done in a cashless fashion, disputes between mates and passengers concerning change would cease to exist. One other benefit of the proposed system is that it enforces accountability of drivers and mates and hence reduces pilfering since transport owner would be able to view transactions. In addition to the above proposed benefits the system also provides a possibility for data mining transactions which can be used to make very useful insights about the behaviour of the average Ghanaian.

## **1.5 Project Objectives**

The main objective of this project is to build an effective, efficient and secure means of making cashless payments for transportation using NFC and QR codes to make the life of the ordinary Ghanaian simpler. The project objectives are to:

- Design and implement a cross platform mobile application that enables payment for transportation using NFC or QR codes.
- Design and implement a web application interface that can allow a system administrator to manage and monitor activities on the system.
- Design and implement a Representational State Transfer (REST) Application Program Interface (API) that would be the brains for both the mobile application and administrator web application.

## **1.6 Overview of Chapter**

This paper comprises of seven chapters. In this chapter, the reader is presented the project and justification for this project. The next chapter focuses on existing and prior research that has been done in connection to cashless modes of payment for transportation particularly those that use NFC and QR. Next chapter presents a high-level or an abstracted description of the system and the requirements or criteria the proposed system should meet. In Chapter 4 the methodology that would guide the design and implementation of the system is discussed. This chapter would also deliberate on the architecture that will serve as the manual for the implementation of the system. The succeeding chapter sets out to describe the implementation process that was used to develop the system and the tests that were conducted on the system. In Chapter 7 the results of the project are examined and evaluated. Lastly, the shortcomings and limits of the system are displayed, and supplementary work required to advance the system are suggested.

## **Chapter 2: Background and Related Work**

This chapter introduces the concept of cashless transactions and its implementation using Near Field Communication (NFC) and Quick Response (QR). The subsequent section describes key concepts and techniques applied in the development of systems using the above the stated technologies. Afterwards, existing solutions to cashless transportation using NFC and QR codes are discussed. Finally, a high-level description of the proposed system is given to the reader along with the frameworks, platforms and services that could be used to develop the system.

### **2.1 Background**

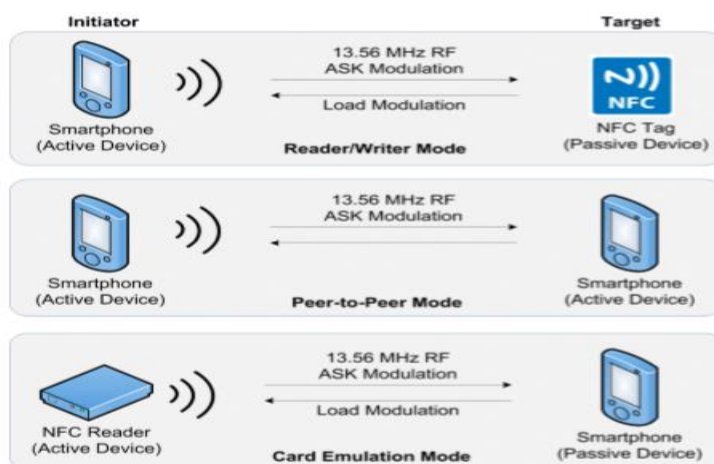
To understand the inner working of the proposed system, it is appropriate to discuss research and projects related to its underlying technologies i.e. Near Field Communication (NFC) and QR codes.

#### **2.1.1 NFC**

NFC which is an acronym for near field communication is a standard for data transfer over extremely short distances which was developed by the joint cooperation of Sony and Philips in late 2002(Schnell 2013). Alternatively, it can also be described as short-range a half-duplex communication (the stations involved can transmit but only one at a time) which was developed with the aim of achieving contactless data transfer and data reception(Coskun, Ozdenizci, and Ok 2015). Data exchange over NFC medium occurs between two devices namely the initiator and the target(Coskun, Ozdenizci, and Ok 2015). During the data exchange process, the initiator device produces a radio frequency field which is within the frequency band of 13.56MHz. When the target device comes within the range of the signal its integrated circuit is activated thereby establishing the communication like between the devices(Coskun, Ozdenizci, and Ok 2015).



After the link is established the devices can then communicate (exchange data) using load modulation, sending messages to each other following the standards specified for NFC(Coskun, Ozdenizci, and Ok 2015). There are two basic types of NFC communication namely active communication and passive communication. In active communication only one device (usually the initiator) generates the radio frequency field while the target does not. Rather the target responds to the generated field by inductive coupling, thus the target does not need a power source to transfer data to the initiator(Mchugh and Yarmey 2012). With active communication on the other hand the both devices emit radio frequency fields, thus they are both initiator and targets simultaneously(Mchugh and Yarmey 2012). This enables the two devices to share data in a peer to peer mode. All initiator devices are usually active during NFC communication whereas targets can either be active or passive and this is dependent on the operating mode. Figure 2-1 shows the diagrammatic descriptions of NFC communication modes just described:



**Figure 2-1 NFC interaction styles and operating modes**

Source: (Coskun, Ozdenizci, and Ok 2015)

NFC is applied in wide variety of industries and sectors of economy (Health, Transportation, etc). One of the popular area of NFC usage is mobile payment. Usually in with this usage, smart phones that wield NFC capability serve as proxies for credit cards(Mchugh and Yarmey 2012). To enable this feature, the owner/user of the smartphone must first allow certain permissions and once they are established, the user is ready to go. Although this does not really add a lot to the payment experience, the peer-to-peer interaction between the smartphone (initiator) and the payment terminal (target) allows to the terminal to get access to other data pertaining to the user's digital purse. This can lead to easy and seamless application of discounts and coupons that the user may have earned during other usages.

Another area where NFC has been found to be extremely useful is the area of access and authentication. For example, the Key and Lock company, Yale have developed Real Living Locks which allows consumers to tap their NFC enabled smartphones to unlock their homes, enable security system and view diagnostic information about the house such as temperature, alarms directly from the home.(Mchugh and Yarmey 2012).

### ***2.1.2 QR Code***

Quick response codes or QR codes for short, is used to refer to an image that consists of black modules on a white background(Durak, Ozkeskin, and Ataizi 2016). They were originally developed in Japan by Denso Wave Corporation which is an affiliate of Toyota automobile company in 1994(Ohigashi Oasay 2011). The name quick response was coined because of the system's multidirectional nature and the capability to decode data rapidly(Ohigashi Oasay 2011). It was originally intended for keeping inventory of automobile parts but was quickly extended beyond that when Japanese marketers realized its vast potential(Ohigashi Oasay 2011). QR codes may seem like fuzzy art but in actuality they house data such as linked URLs, text values

such as locations, name, descriptions etc which are encrypted and embedded in to the black modules on the white background. In order for data exchange and communication to occur, a smartphone with a fitted camera, internet connection, a QR code reader software and the QR code image is all that is required(Ohigashi Oasay 2011). The interaction occurs in the following fashion:

1. The smartphone consumer launches the QR code reader software on their smart phone which also launches the camera.
2. The consumer then focuses the camera on the QR code image.
3. Once the image comes into cameras focus the application decodes the QR code and serves the data encrypted in it (usually linked URLs,)

One of the most popular area of use of QR code technology is usually in libraries. This because it presents the providers of library services and consumers of library products such as aiding in cataloguing books and library resources, quick registration for library services, advertisement of library services etc. One other area where QR code is rapidly gaining ground is in mobile payment. Some examples of mobile applications that are capable include Slydepay, Master Pass.

### **2.3 Relevance of Technologies**

It was highly necessary to gain understanding concerning these technologies because they are the means through which payments in the application are facilitated. This is because information read either using the interaction of smartphone and an NFC target or the interaction between a smartphone camera and a QR code determines the fare during transit. Thus, they are very key to the working of the application.

## **2.4 Chapter Overview**

This chapter explored the origins and relevance of both NFC and QR codes which are both key technologies needed for this project's fruition. The next chapter focuses on the requirements specification for the application in addition of to the intended users of the system.

## **CHAPTER 3: REQUIREMENTS SPECIFICATION**

The purpose of this chapter is to provide a detailed description on the functionalities of the proposed solution which is the Xipora Cashless Transportation Payment Platform. This chapter also describes the intended users of the system, hardware and software requirements.

### **3.1 Project Scope**

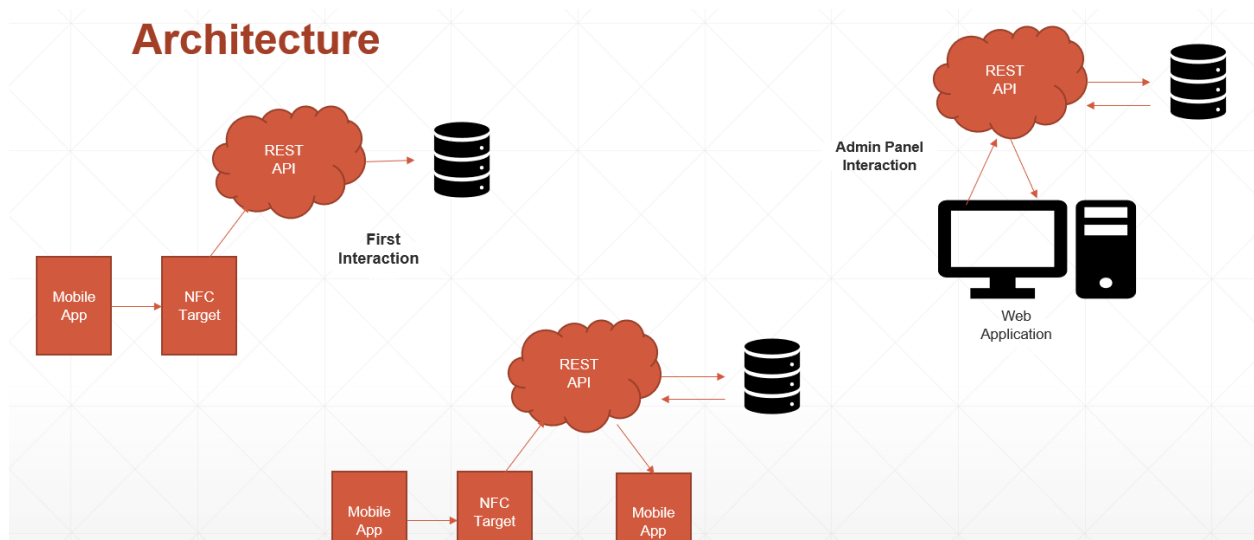
The Xipora Cashless Transportation Payment Platform should comprise of four main components namely the REST API which is used to store and retrieve data concerning the vehicle, transactions and users with which the system monitors, a web application front-end, a mobile application and an NFC target which the mobile application would communicate with for cashless payment. The system should ensure that payment for transportation for the user of public transportation in Ghana is made a seamless and simple experience and also to provide information to vehicle owners concerning the transactions of the vehicle.

### **3.2 Overall Description**

#### ***3.2.1 Product Perspective***

The Xipora Platform can be considered as an efficient and simple means of paying for public transportation. It should serve as a mobile application which will be dynamically fed with data by the cloud but there will also be a web application to enable other user classes such as an administrator and retailer who sells transportation units to partake in the ecosystem. The cloud should be primarily responsible for serving data to both the web and mobile application. This data should include recent transactions (including transactions involving the transport user and a transport credit retailer) and current balance for the public transportation user and recent transactions, volume of transactions by retailer and the transactions done by a particular vehicle which will be relevant for the vehicle owner. The

web application should be accessible to all class of users (transport user, retailer, vehicle owner and system administrator) whereas the mobile application is intended for only then vehicle owner and transport user. Below is the interactions architecture of Xipora Platform which describes how the various client-side application should interact with the back-end and hardware when cashless payments is being in the system.



**Figure 3-0-1 High-level Overview of subsystems and Components of the Xipora Cashless Transportation Payment Platform**

### ***3.2.2 User Descriptions and Characteristics***

The aim of the Xipora Payment Platform is to ensure a seamless and simple means of making payment for public transportation services in a cashless manner using Near Field Communication (NFC) and Quick Response (QR) codes. The main user types for whom the system is intended for are:

- a. User of public transportation.
- b. Vehicle Owners.

Other classes of users of the system includes retailer and system status administrator who primarily exist to serve as facilitators for the user of public transport and the vehicle owners. The retailer's role is to sell transport unit to the users of public transport whereas the system status administrator's function is to monitor the state of the system. The user of public transport should be able to view information on their transactions both with a retailer and interaction with an NFC target in the vehicle. The vehicle owners should have access to view transactions on their related vehicle. The retailers should access to the transactions involving themselves as well as the ability to create transactions. The system status administrator should have access to the status of the system including the number of transactions (including vehicle transactions and retailer transactions) made in total, total number of users (including the segmentation of the various user classes) and total number of vehicles tracked by the system.

### ***3.2.3 Operating Environment***

The Xipora Payment Platform should primarily be a mobile application with data served from a cloud. To supplement the mobile application, the system should have an accompanying web application to enable the retailer and system status administrator to function. Due to large amount of data that is generated because of system usage the data store for the system should have scalability and a fast data access.

### **3.3 Use-case Scenarios**

Below are the described use case scenarios concerning the main classes of users i.e. the public transport user and the vehicle owner.

### ***3.3.1 Public Transport User Cases***

#### **A user with large notes**

An average working-class citizen is on his way to work from his house at 6:45am. Since he does not own a vehicle, he opts for public transportation (“tro-tro”) and stops a bus to his destination. While in the bus the mate asks him to pay a fare of GHS 1.70 and he offers a twenty cedi note. Because the mate does not have enough change to give him, they engage each other in a heated argument.

#### **A user experiencing varying fares to the same location**

A day student attending University of Ghana commutes from his residence located on the Spintex Road to the University premised. Since he lacks a driver’s licence and a private vehicle he has no choice but to opt for public transportation. From Cylinder junction, Spintex Rd. to the Accra Mall which is on route to his final destination costs GHS 1.60 on a regular day. On another occasion on different bus, the cost of commuting from Cylinder junction, Spintex Rd. to the Accra Mall shifts to GHS 1.80. On another separate occasion on a different bus, the cost of commuting on that same route is GHS 1.65. Thus, this frequent change in fare leaves him feeling down every time he commutes.

### ***3.3.2 Vehicle Owner Use Case***

#### **An Owner Confirming Transactions Made**

A “tro-tro” owner meets up with the driver and mate who utilize the vehicle he owns for work. He approaches the two individuals and asks for information concerning the day’s activities. The driver and the mate then go on to inform by word of mouth about the day’s

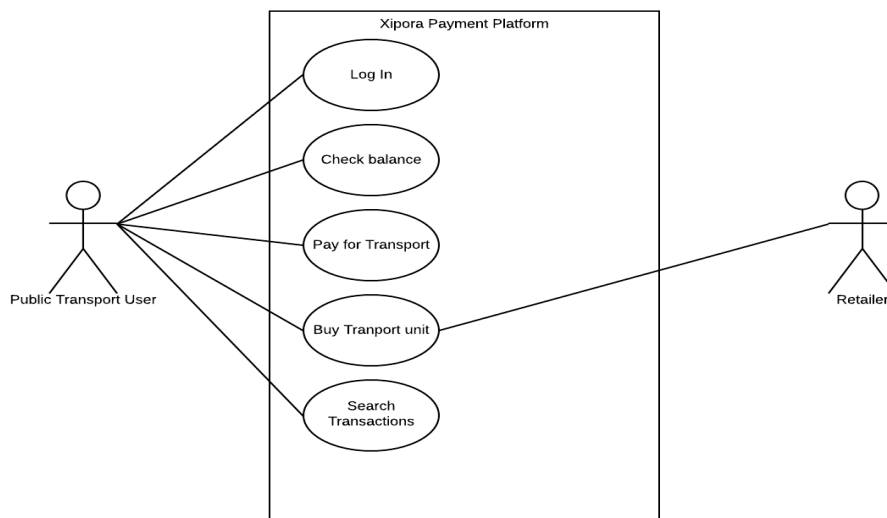


activities. Given the fact that he has no way of confirming their account, he has no choice but to take their word for it.

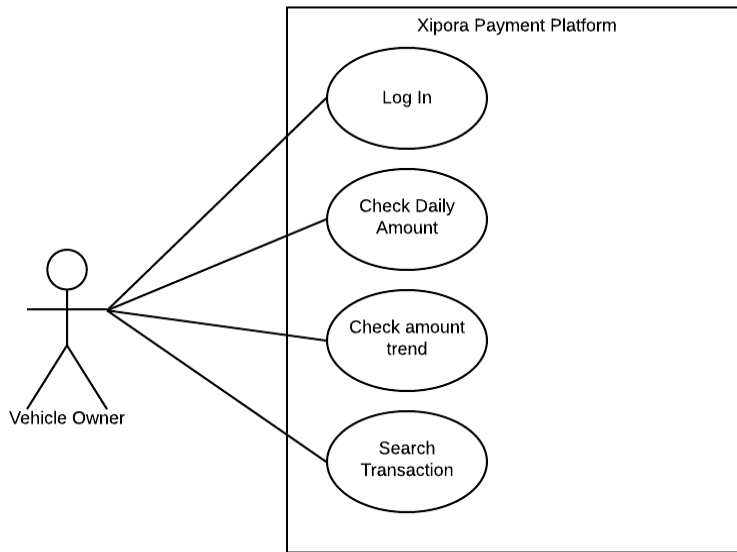
The following are the system requirements generated from the user scenarios:

- The system should be able to facilitate cashless transportation payments.
- The system should be able to use suitable and acceptable metrics such as distance travelled to determine transit fare.
- The system should be able to record transactions and provide visual representation (graphs and charts) concerning transactions and generate some report.

Below figures 3-0-2 and 3-0-3 show the use case diagrams for the public transport user and the vehicle owner.



**Figure 3-0-2 Use case Diagram for public transport user.**



**Figure 3-0-3 Use case diagram for vehicle owner**

### **3.3 Product Features**

This section of the chapter provides an outline and description of the features and functionalities of the Xipora Payment Platform. These functionalities and features are sectioned in two parts namely principal features and supplementary features. As the name implies the principal features are essential for the core operations of the system whereas the supplementary features are meant to add additional functionality and enhance the user experience during the usage of the system hence they will be implemented when principal features are complete.

#### ***3.3.1.1 Principal Features***

1. Sign Up / Registration
  - a. Should allow transport users, transport credit retailers and vehicle owners to register for the platform.

## 2. Log In

- a. Should allow public transport user or a vehicle owner to log into the application and initiate usage.

## 3. Transaction Search

- a. Should allow transport users to search for and list transactions made either with a retailer or during a transit event by date.
- b. Should allow a vehicle owner to search and list all transactions made by vehicle on a date.
- c. Should allow a transport credit retailer to search for transactions by date, user identity or by both.

## 4. Transaction Creation

- a. Should enable the creation of a transit transaction by the public transport user when NFC enabled smartphone interacts with NFC target or when user scans QR code.
- b. Should enable the creation of a transport credit transaction.

### ***3.3.1.2 Supplementary Features***

## 4. Mobile Money Integration

- a. Should public transport users to pay for transit using their mobile money (including Airtel money, Tigo cash, Vodafone cash) and other fintech accounts such as Slydepay, ExpressPay, etc.

## 5. SMS Notification

- a. Should enable vehicle owners to be alerted by SMS notification when their vehicle engages in a transaction.

With respect to the intentions of the system Figure 3-0-4 and 3-0-5 show the activity for both the mobile application and web application.

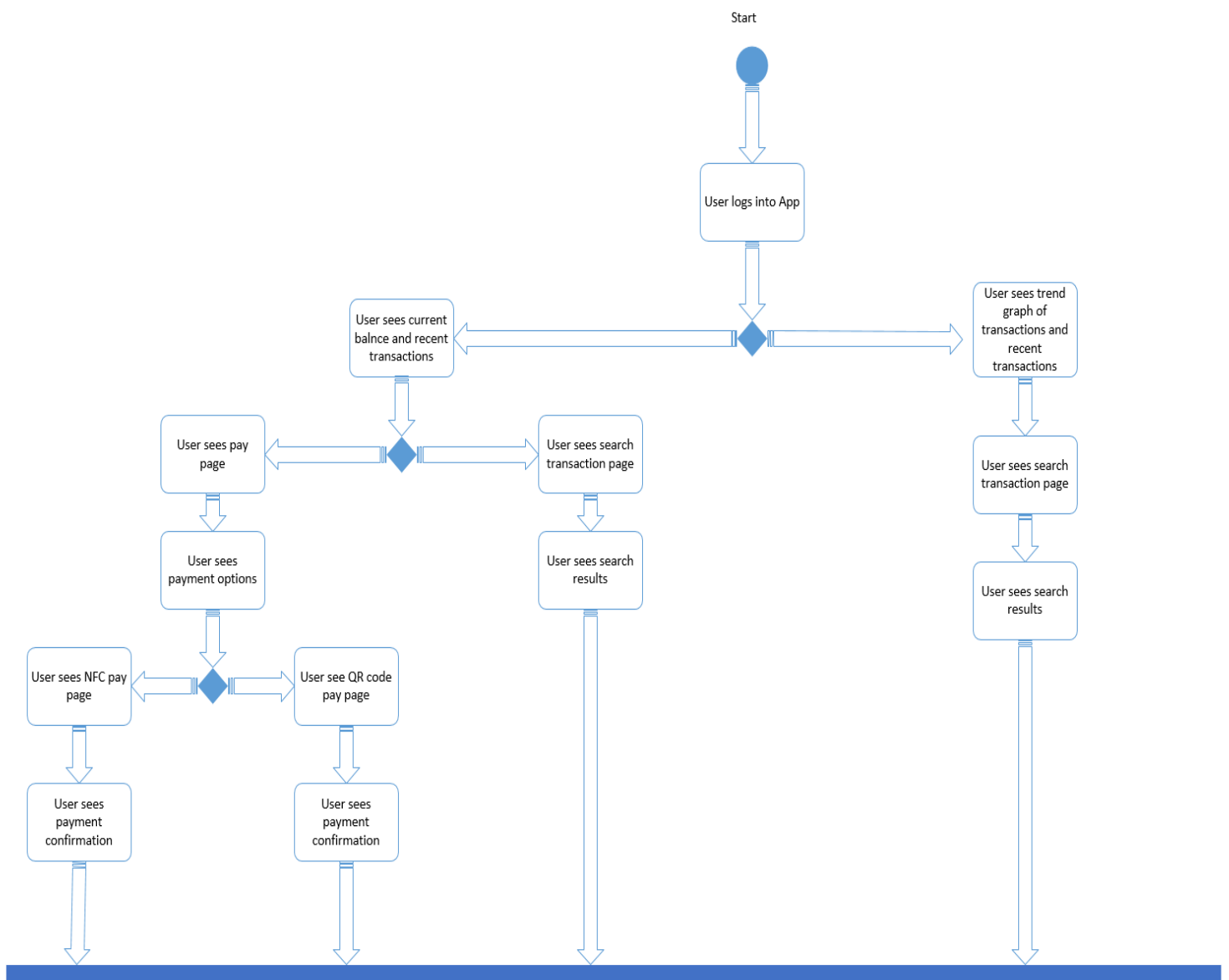


Figure 3-0-4 Activity diagram for mobile application

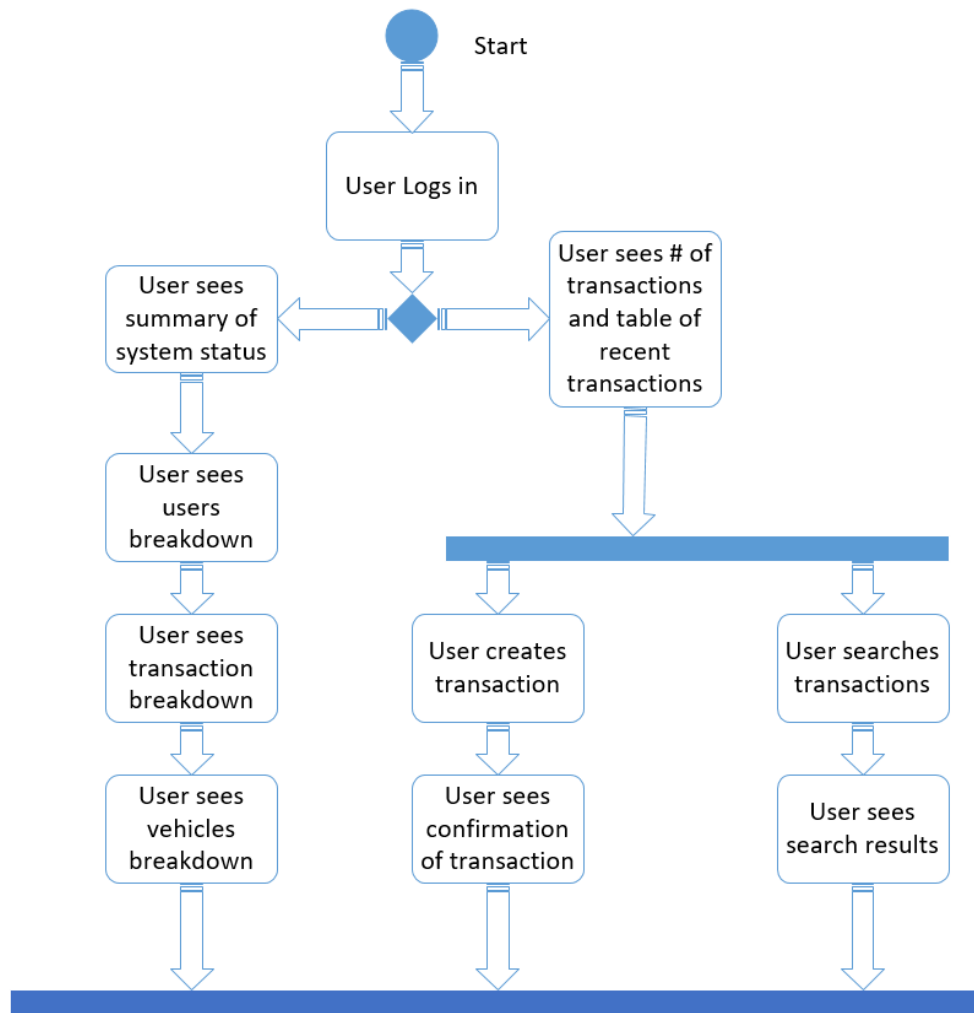


Figure 3-0-5 Activity Diagram for Web application.

## Chapter 4: Architecture and Design

This chapter's primary aim is to give the architecture and design of the Xipora Cashless Payment Platform.

### 4.1 High-level Overview of System.

The Xipora Payment Platform's development is based on REST Architecture. REST which stands for Representational State Transfer describes an architectural style based on HTTP which was first introduced by Roy Fielding in 2002 (Sun, 2009). Some characteristics of REST systems include the separation between clients and servers (meaning changes in one does not affect operation of the other) which allows independent evolution of both components, statelessness (both parties lack information concerning the state of each other) etc. Figure 4-0-1 below shows a very high-level description of the entire system. The succeeding sections of this chapter delve into the various components of the system.

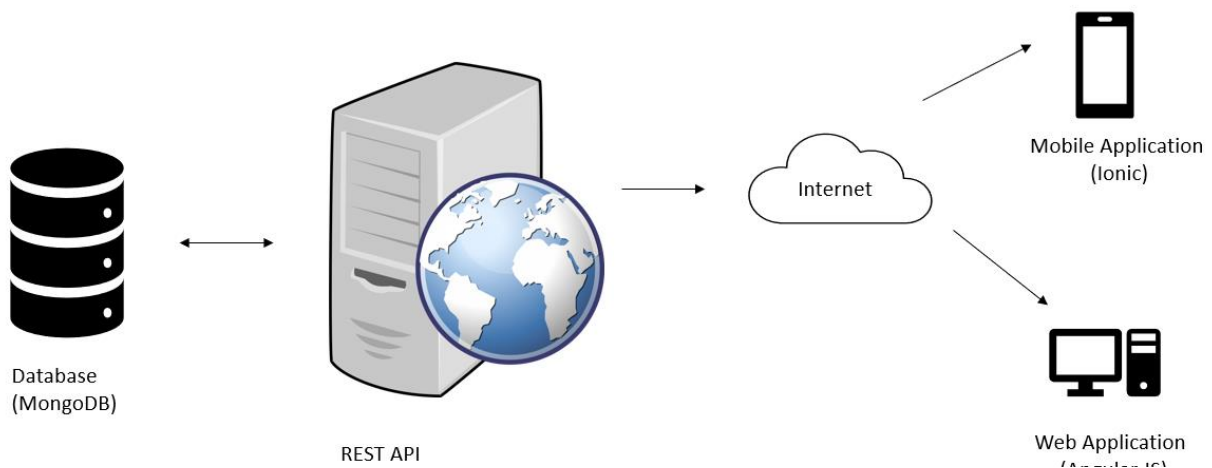


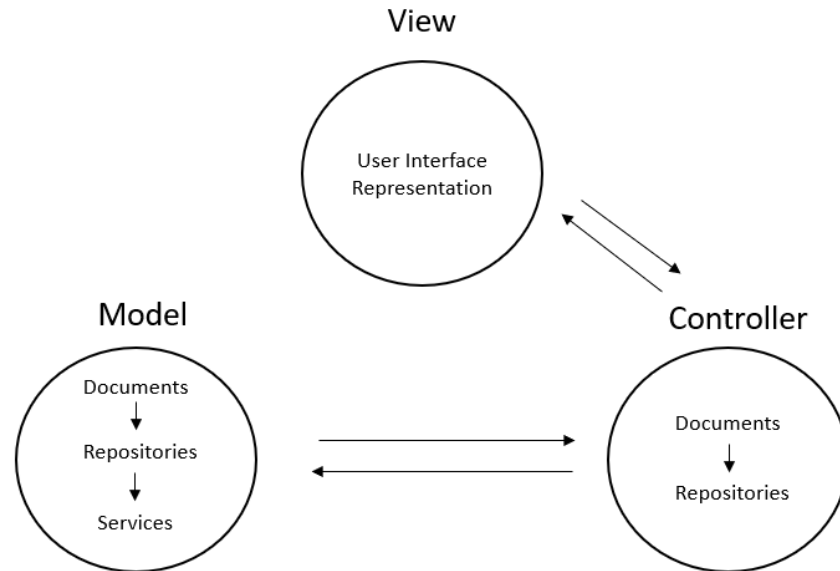
Figure 4-0-0-1 High-level Overview of System

## 4.2 REST Service

The server-side application which is the REST API, acts as a liaison between the various client-side applications (namely the mobile application and the web application) and the database was built with the Model View Controller (MVC) Architectural Framework in mind. The database used in this system's implementation is MongoDB which is a document-oriented NoSQL database which stores data as JSON documents. The JSON document contains the properties specify of the attributes of the entity and multiple documents are referred to as a collection. This decision was taken because of the large scalability potential of MongoDB and because it enables faster data access it uses internal memory for storing the working set.

In MVC the application is separated into the Models which are the data access layer, the Views which is the presentation layer and the Controllers which is in charge of the handling requests and responding to them rendering different views using the data given by the model. The REST API was developed using a unique take on the MVC architecture known as the Java Spring MVC. In this variation the model primarily consists of the definitions of objects that corresponds the documents in a collection. The collections pertaining to the system include User, Vehicle, Vehicle transaction and Retailer Transactions. These documents are linked with Java classes annotated as repositories which hold and contain the data which are then accessed using Java classes annotated as services. The views in this variation of MVC are also Java classes which mainly represent the user interface (forms etc.). The controllers in this variation of MVC which simple Java classes annotated as REST controller contains methods corresponding to different exposure points (URLs that are mapped to requests) which contains method calls to

functions that implement the exposure point's logic. Figure 4-0-2 below gives a pictorial view of the described architecture.



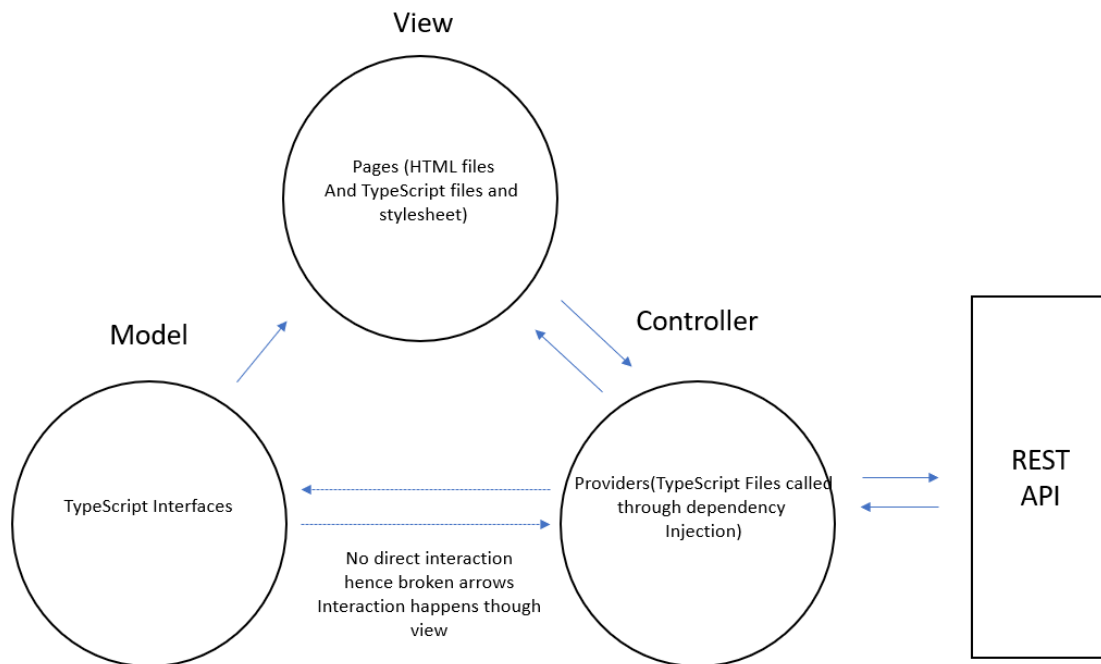
**Figure 4-0-2 Architecture of REST API**

### **4.3 Mobile Application**

The mobile application which is the principal client-side application and the primary source of contact for the public transport user and the vehicle owner was also designed with MVC architecture framework in mind. However, its implementation was done using the Ionic Framework for cross-platform mobile development to enable users of different platform such as Android, IOS and Windows have access to the application. The View in this case is represented by the pages which consists of an HTML file, two TypeScript files and a stylesheet. The Models are represented by TypeScript interfaces which define the various data types and variable names of the properties of the objects corresponding to the documents in the MongoDB collections. Lastly, the Controllers are implemented using special TypeScript files known as provider which contain implementations of requests that are to be sent and handled by the server-side



application. These controllers interact with the model through the View in a process called dependency injection as shown in Figure 4-0-3 below.

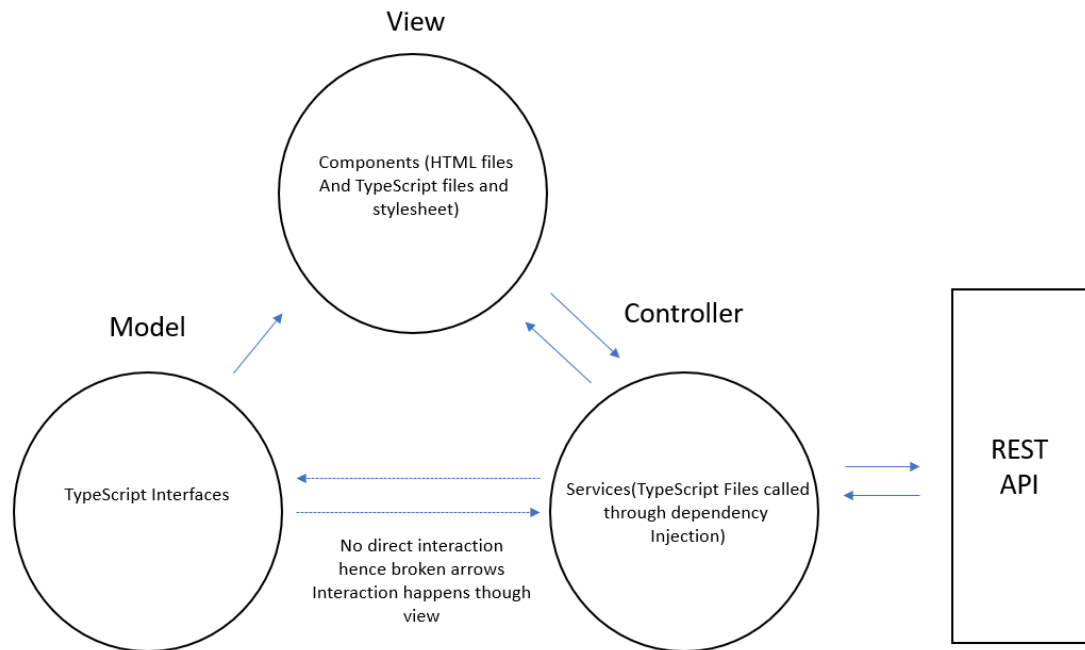


**Figure 4-0-3 Architecture of Mobile Application**

## 4.4 Web Application

The web application can be accessed by all user classes, but it is primarily meant to serve as a medium for retailers to operate and a means for the system status administrator to check up on the system. This component was also implemented with MVC in mind. However, this MVC implementation was done using the Angular JS framework for web application development. The Models in this case are represented by TypeScript interfaces which are exported to enable the View to interact with them. The View in this implementation are referred to as components and consists of an HTML file, two TypeScript files and a stylesheet. The Controllers in this implementation are referred to as services. These services are TypeScript files that contain

methods which send various requests to the REST API. These controllers interact with the model through the View in a process called dependency injection. Figure 4-0-4 summarizes the above explanations.



**Figure 4-0-0-4 Architecture of Web Application**

## 4.5 Overview of Chapter

This chapter contains and discusses and provides a visual description of the overall architecture of the Xipora Payment Platform. It also goes a step further to outline the architecture of the various components of the system including the REST API, the mobile application and the web application. The next chapter discusses the implementation of the system including the individual components in detail.

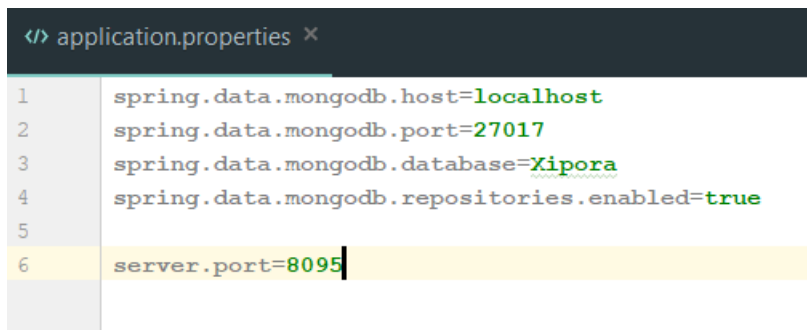
## **Chapter 5: Implementation**

This chapter presents details that are tailored towards the implementation of the Xipora Payment platform. In this section, tools such as libraries, frameworks and APIs used to build the components of the system as well as how they operate are discussed in this chapter. This chapter begins by first discussing the implementation details of the REST API followed by the implementation details for the Web Application which then is succeeded by the implementation details of the Mobile Application.

### **5.1 REST API Implementation**

The REST API was developed using Java as the server-side programming language in a framework known as Java Spring boot. Management of dependencies for the project was done using Maven which is a dependency and project management tool for Java projects. The dependency management for this project is defined in the pom.xml file. These configurations were done automatically using the Spring initializer web application, hence the pom.xml was not created manually. The main dependencies used in this application are Spring Web, MongoDB and Log4j.

The configurations to enable the application to connect to the instance MongoDB were done in two files namely MongoConfig.java and application.properties. The MongoConfig.java contains references which map the various repositories to their respective collections in the MongoDB database. The application.properties file contains configurations such as the server port, the name of the database, reference to the MongoDB host, MongoDB port which is being used etc. Below are snippets of each of these files.

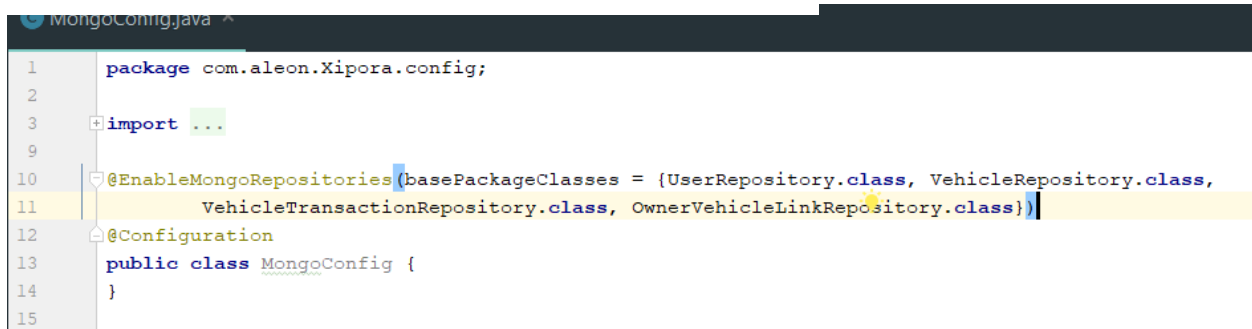


```

</> application.properties x
1  spring.data.mongodb.host=localhost
2  spring.data.mongodb.port=27017
3  spring.data.mongodb.database=Xipora
4  spring.data.mongodb.repositories.enabled=true
5
6  server.port=8095

```

**Figure 5-0-1 Configurations in application.properties**



```

MongoConfig.java x
1  package com.aleon.Xipora.config;
2
3  + import ...
9
10  @EnableMongoRepositories(basePackageClasses = {UserRepository.class, VehicleRepository.class,
11  VehicleTransactionRepository.class, OwnerVehicleLinkRepository.class})
12  @Configuration
13  public class MongoConfig {
14  }
15

```

**Figure 5-0-2 MongoConfig.java file.**

As stated in the architecture and design section, the REST API was developed using MVC. The model consists of a Java file annotated with an '@Document' annotation which defines the properties of the document and the collection associated with that document. The next component of the model in the Java Spring Boot is the Java interface annotated with an '@Repository' annotation which inherits from a parent class known as MongoRespository. This file holds or contains the actual data coming from the database. It also contains some method headers that are implemented in service which is the last component of the model. The service which is the last component of the Java Spring Boot model is a Java file that is annotated with the '@Service' annotation. The service contains methods which facilitate access of data from the repository. Figures 5-0-3, 5-0-4, 5-0-5 show snippets of the files that make up a typical model in Java Spring Boot MVC.

## Model

```
User.java ×
1  package com.aleon.Xipora.documents;
2
3  import org.springframework.data.annotation.Id;
4  import org.springframework.data.mongodb.core.mapping.Document;
5
6  @Document(collection = "Users")
7  public class User {
8
9      @Id
10     private String userID;
11     private String phoneID;
12     private String name;
13     private String phoneNumber;
14     private String email;
15     private String password;
16     private double currentBalance;
17     private String userClass;
18 }
```

Figure 5-0-3 Snippet of User Document class.

```
UserRepository.java ×
1  package com.aleon.Xipora.repositories;
2
3  import com.aleon.Xipora.documents.User;
4  import org.springframework.data.mongodb.repository.MongoRepository;
5  import org.springframework.stereotype.Repository;
6
7  import java.io.Serializable;
8
9  @Repository
10 public interface UserRepository extends MongoRepository<User, Serializable> {
11
12     User findByEmail(String email);
13
14     User findByPassword(String password);
15
16     User findByUserID(String userID);
17
18
19 }
```

Figure 5-0-4 Snippet of UserRepository Interface

```

1  package com.aleon.Xipora.services;
2
3  import com.aleon.Xipora.documents.User;
4  import com.aleon.Xipora.repositories.UserRepository;
5  import org.springframework.beans.factory.annotation.Autowired;
6  import org.springframework.stereotype.Service;
7
8  import java.util.List;
9
10 @Service
11 public class UserService {
12
13     @Autowired
14     UserRepository userRepository;
15
16     // public User findByEmail(String email){
17     //     return userRepository.findOne(email);
18     // }
19
20
21     public List<User> findAll() { return userRepository.findAll(); }
22
23
24     public void save(User user) { userRepository.save(user); }
25
26
27     public void insertIntoUsers(User user) { userRepository.insert(user); }
28
29
30     public User findByEmail(String email) { return userRepository.findByEmail(email); }
31
32
33     public User findByPassword(String password) { return userRepository.findByPassword(password); }
34
35
36     public User findByUserID(String userID) { return userRepository.findByUserID(userID); }
37
38 }

```

**Figure 5-0-5** Snippet of UserService class

The view basically consists of Java files that contains references to data and are gained from forms. These java files are used to hold requests which are then passed to the relevant API methods and are thus sometimes referred to as request bodies. Figure 5-0-6 below is a snippet of the UserForm.java file which is one of the views implemented.



```

1  package com.aleon.Xipora.controller.forms;
2
3
4
5  /**
6   * Created by Leon Ampah on 16/01/2018
7   */
8  public class UserForm {
9
10     private String userID;
11     private String name;
12     private String email;
13     private String gender;
14     private String phoneNumber;
15     private String password;
16     private String phoneID;
17     private String userClass;
18     private double currentBalance;
19
20     public String getUserID() { return userID; }
21
22     public void setUserID(String userID) { this.userID = userID; }
23
24     public String getPassword() { return password; }
25
26     public void setPassword(String password) { this.password = password; }
27
28     public String getPhoneID() { return phoneID; }
29
30     public void setPhoneID(String phoneID) { this.phoneID = phoneID; }
31
32     public String getName() { return name; }
33
34     public void setName(String name) { this.name = name; }
35
36     public String getEmail() { return email; }
37
38     public void setEmail(String email) { this.email = email; }
39
40
41
42
43
44

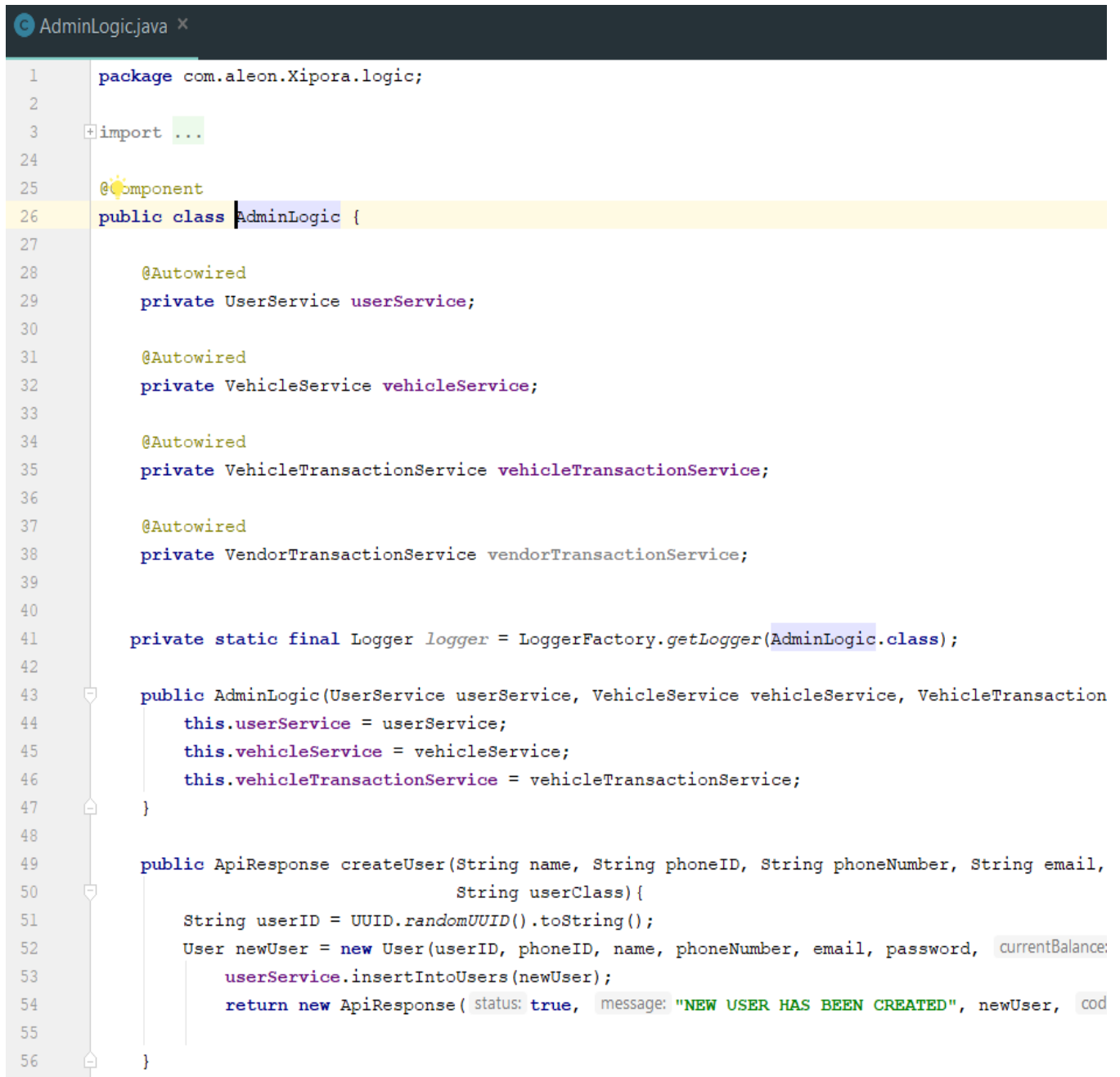
```

Figure 5-0-6 UserForm.java file representing a view

The controllers of this application have two parts namely the logic file and the controller file. The logic file which is a java file and has the difference services which are part of the model as instance variables and methods make use of these services to interact with collections and documents in the MongoDB database. The controller file is annotated with ‘@RestController’ annotation which informs spring that the file is a controller. It also holds a reference to its associated logic file and contains methods which are annotated with ‘@RequestMapping’ (where the URLs for requests are specified and denote type of request that can be made to that URL).

These methods handle requests that come in any of the client-side applications. Figure 5-0-7 and 5-0-8 are snippets of some of the code in these files.

## Controller

A screenshot of an IDE window titled 'AdminLogic.java'. The code is in Java and defines a Spring component. It includes package declarations, imports, and annotations like @Component, @Autowired, and @Logger. The class AdminLogic has several private fields for services (userService, vehicleService, vehicleTransactionService, vendorTransactionService) and a static logger. It also has a constructor and a createUser method that interacts with the userService to create a new user and return an ApiResponse.

```
1 package com.aleon.Xipora.logic;
2
3 import ...
24
25 @Component
26 public class AdminLogic {
27
28     @Autowired
29     private UserService userService;
30
31     @Autowired
32     private VehicleService vehicleService;
33
34     @Autowired
35     private VehicleTransactionService vehicleTransactionService;
36
37     @Autowired
38     private VendorTransactionService vendorTransactionService;
39
40
41     private static final Logger logger = LoggerFactory.getLogger(AdminLogic.class);
42
43     public AdminLogic(UserService userService, VehicleService vehicleService, VehicleTransaction
44         this.userService = userService;
45         this.vehicleService = vehicleService;
46         this.vehicleTransactionService = vehicleTransactionService;
47     }
48
49     public ApiResponse createUser(String name, String phoneID, String phoneNumber, String email,
50         String userClass){
51         String userID = UUID.randomUUID().toString();
52         User newUser = new User(userID, phoneID, name, phoneNumber, email, password, currentBalance;
53         userService.insertIntoUsers(newUser);
54         return new ApiResponse( status: true, message: "NEW USER HAS BEEN CREATED", newUser, cod
55
56     }
```

Figure 5-0-7 AdminLogic.java file snippet



```

11 public class AdminController {
12
13
14     @Autowired
15     private AdminLogic adminLogic;
16
17
18     @RequestMapping(value = {"/user/save"}, method = RequestMethod.POST)
19     public ApiResponse createUser(@RequestBody UserForm user) {
20
21         return adminLogic.createUser(user.getName(), user.getPhoneID(),
22             user.getPhoneNumber(), user.getEmail(), user.getPassword(), user.getUserClass());
23     }
24
25
26     @GetMapping("/user/showAll")
27     public ApiResponse getAllUsers() { return adminLogic.getAllUsers(); }
28

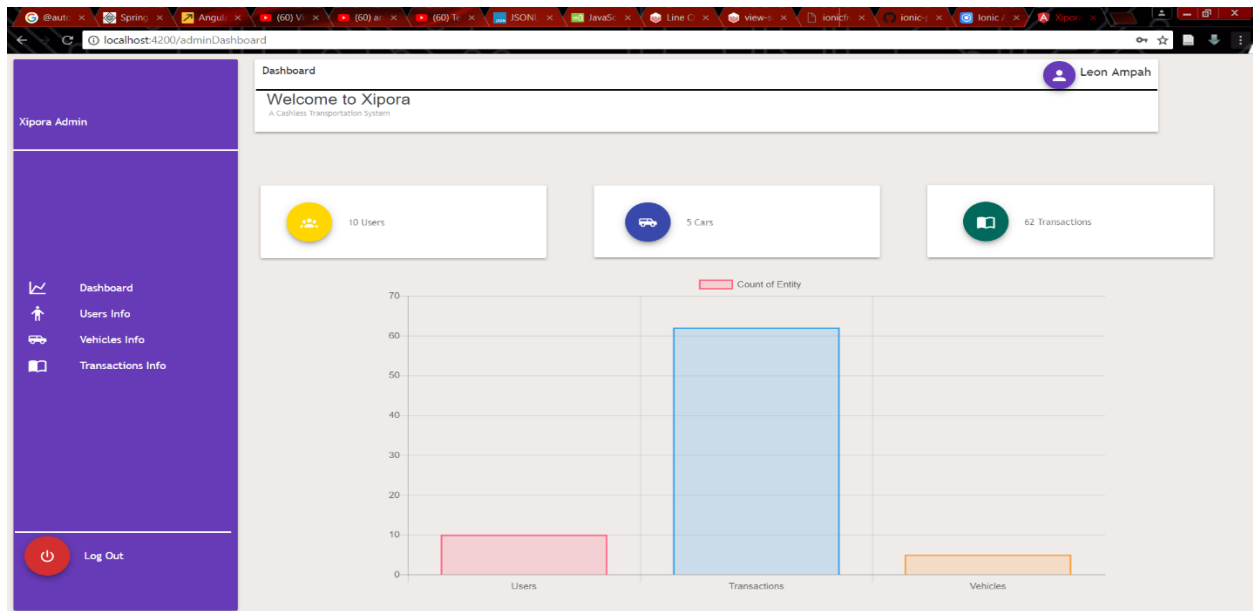
```

**Figure 5-0-8 AdminController.java file snippet**

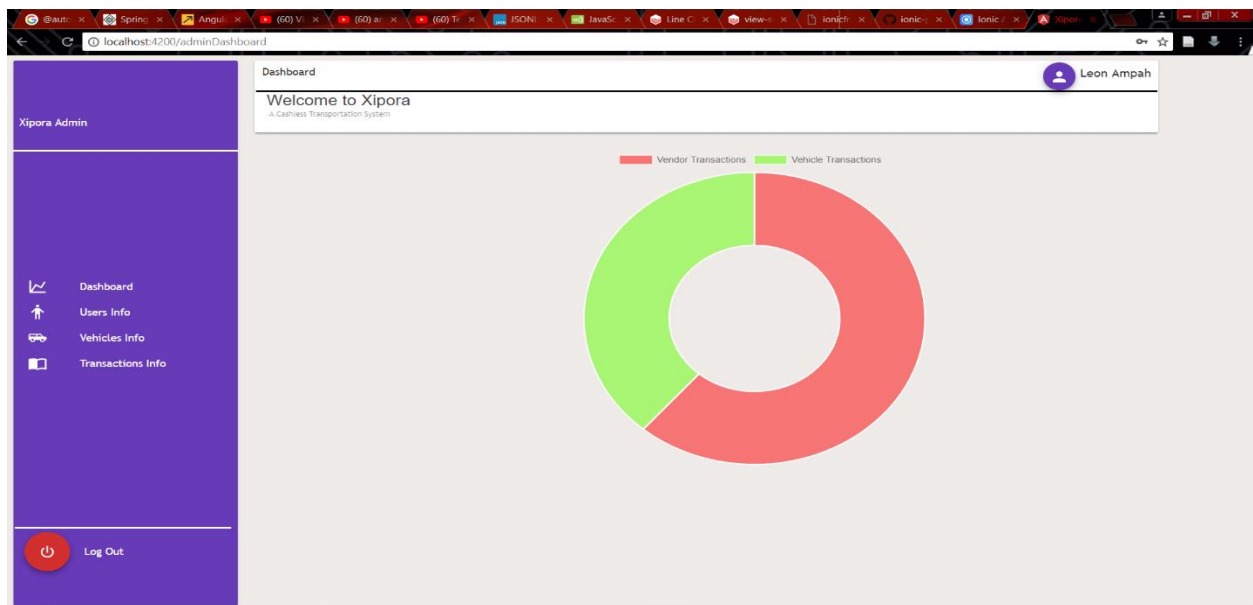
## 5.2 Web Application

The Xipora web application, which implemented MVC using AngularJS can be accessed by all classes of users however, it is primarily designed to meet the needs of the system status administrator and the transportation unit retailer.

The screens that are visible to the system status administrator are optimized so that he/she can perform its associated tasks without seeing specific details about transactions and users; Thus, preserving the privacy of the other user classes. As such, all the system status administrator sees are charts and counts. Figure 5-0-9 and 5-1-0 show the typical screen such as the breakdown of entities in the system and the breakdown of transactions.



**Figure 5-0-9 Screenshot of the system status admin's dashboard**



**Figure 5-0-10 Screenshot the transactions breakdown**

The retailer's screens on the other hand was optimized for this user class to be able to sell transport credit. Thus, it provides a medium which facilitates this user class to make transactions, search through the transactions they have made, see recent transactions and total number of transactions they have made. Figures 5-1-1 and 5-1-2 show screenshots of transaction

creation form and the dashboard page which enable the retailer to create transactions and view the total number of transactions as well as their recent transactions respectively.

The screenshot shows the 'Vendor View' page of the Xipora Vendor application. The left sidebar contains navigation links: 'Dashboard', 'Search Transactions', 'Create Transaction', and a 'Log Out' button. The main content area displays a 'Welcome to Xipora' message and the system name 'A Cashless Transportation System'. Below this, there are four input fields for transaction details: 'Transaction ID \*' (6PKSyFQBIQ6eerDxJYK42vx), 'Retailer ID \*' (5a8f76ddfa607aaf4a419544), 'User ID \*', and 'Transaction Amount \*'. A green 'Submit Transaction' button is located at the bottom of the form.

Figure 5-0-11 Screenshot of transaction creation form.

The screenshot shows the 'Dashboard' page of the Xipora Vendor application. The left sidebar is the same as in the previous screenshot. The main content area displays a 'Welcome to Xipora' message and the system name 'A Cashless Transportation System'. Below this, a large orange number '38' is displayed, indicating the total number of transactions. The text 'You have made' and 'Transactions in total' are positioned to the left of the number. A section titled 'Previous Day's Transactions' is visible, followed by a 'Filter' input field. Below the filter, a table displays transaction data.

Transaction ID	User ID	Transaction Cost	Transaction Date
----------------	---------	------------------	------------------

Figure 5-0-12 Screenshot of Dashboard page

### 5.3 Mobile Application

The Xipora Payment Platform mobile application which implemented MVC using Ionic Framework is designed for users in the transport user class and the vehicle owner class.

The public transport user's view was implemented in an optimum fashion to ensure that this user class can perform actions such as seeing their balance, making payment via NFC or QR code, viewing transactions and searching for transactions. The making of payments through NFC and QR code which is a key aspect of the application was implemented through the use of native plug-ins provided by the Ionic framework. Figure 5-1-3 and Figure 5-1-4 show the homepage and the search transactions page which allow the user to see their balance and recent transactions and to search for transactions.

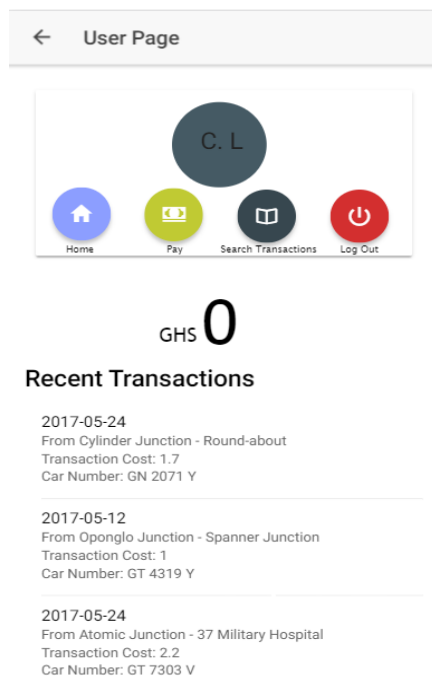


Figure 5-0-13 Homepage screenshot

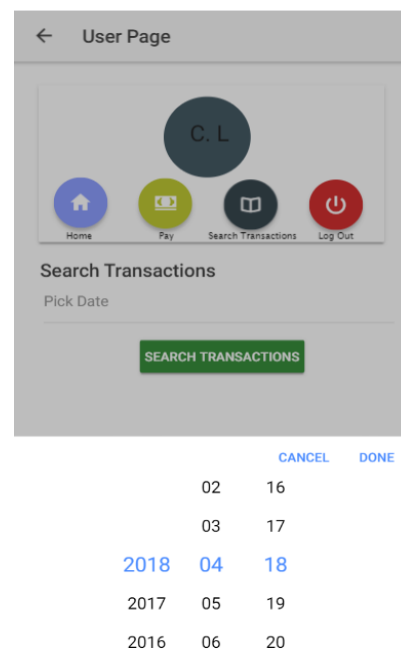


Figure 5-0-14 Search Page

The vehicle owner’s view was also implemented in an optimum and intuitive fashion to ensure that their needs which include a graphical description of the trend of the number of transactions over a period, a view of recent transactions made by the vehicle owner and the avenue to search for transactions. Figure 5-1-5 and 5-1-6 show the homepage and search page of the vehicle owner’s side of the applications.

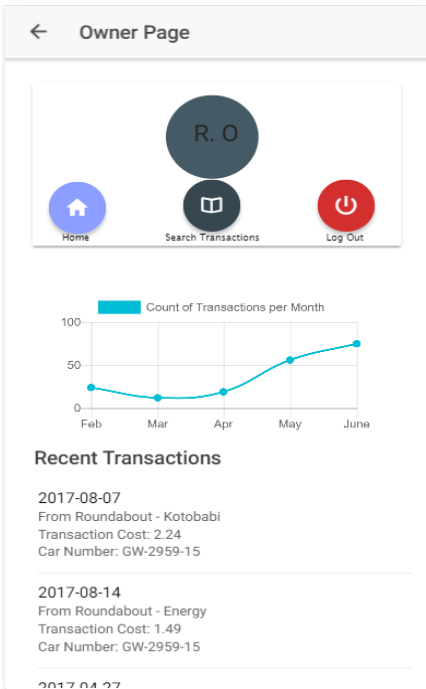


Figure 5-0-16 Vehicle owner Homepage Screenshot

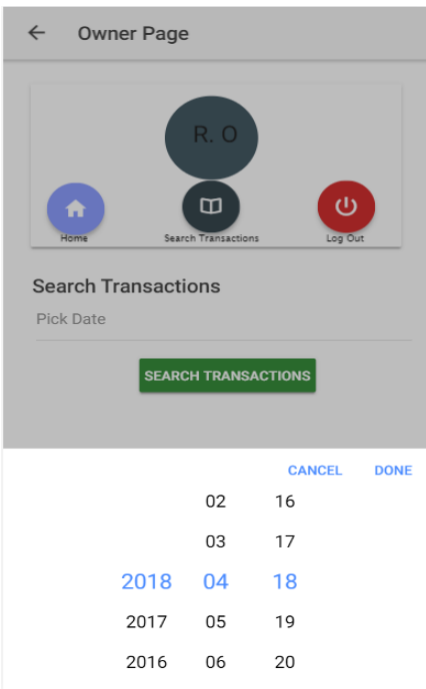


Figure 5-15 Vehicle owner Search page

## 5.4 Chapter Overview

In this chapter, the implementation details of each component of the Xipora Payment Platform has been discussed thoroughly. Additionally, relevant screenshots and code snippets were shown to give the reader a vivid picture of the implementation details talked about in the chapter. In the next chapter, testing of the application is thoroughly discussed.

## **Chapter 6: Testing and Results**

In this section, testing of the system with regards to its performance and how well it meets the requirements are discussed. In this paper three types of testing are discussed namely; Component testing, System-level testing and User testing were performed.

### **6.1 Component Testing**

Component testing was performed on all components including the REST API, the web application and the mobile application. In testing the REST API which was developed using Java spring boot, it was essential to ensure that HTTP requests such as 'GET' and 'POST' were operational. This was achieved using Postman, an API development environment, which allowed the sending 'POST' and 'GET' requests to the REST API as well as displaying the appropriate response associated with that request. For example, when a 'GET' request is made to display all users in the system, a response containing the requested data in JSON format was returned.

In testing both the web application and mobile application which were built using AngularJS and Ionic native, Jasmine which was an opensource testing framework for JavaScript was used.

During the tests, assertions were made as to whether requests were sent from both mobile application and the web application to the correct URLs of the API. Assertions were also made as to whether the responses being received by the respective client-side applications were also valid.

### **6.2 System-Level Testing**

The second level of testing performed was system-level testing. To perform system-level testing, all individual components i.e. the REST API, the mobile application and the web application were joined together to form the Xipora payment platform. This testing was achieved

using Google chrome debug console which provided a visual representation of the requests sent to the REST API from both mobile application and the web application along with their associated responses from the API. Initially a 'No Access-Control-Allow-Origin' error was thrown hence the API was unable to receive request coming from both the mobile and web applications. However, this issue was quickly dealt with by adding the '@CrossOrigin' annotation to the controller classes in the REST API.

### **6.3 User Testing**

The final type of test performed was user testing. This meant giving the application to an individual to test. The application was installed on a Google Nexus 6P running Android 8.0 (API level 26) and was given to two users, one of which owned a vehicle and one who is a user public transport.

An insight gained from the user of public transport was that the mobile applications interface made it difficult to use the system. Upon analysis of the user's feedback, it was realized that the application did not follow the current design trends of currently deployed Fintech applications. Thus, steps were taken to ensure that the interface was modified to fit into current trends to make the application more intuitive to use.

A major insight gained from the vehicle owner was that the application did not notify him when a transaction occurred. Thus, he always had to log into the application to check its status and this became a bother to him. After careful thought on the user's input, it was decided that the next iteration of the application would include an SMS service to facilitate the notification of transactions without having to log into the application.

## **6.4 Chapter Overview**

In this chapter, various levels of testing the application were discussed in detail. In addition, insights concerning the use of the application as well as various work-arounds to problematic scenarios were also discussed. In the next chapter, various challenges and limitations of the application as well as future is discussed.



## **Chapter 7: Conclusion and Recommendations**

In this chapter, the summary of the entire project is discussed. It extensively looks at how the functional requirements of the system were met, limitations of the project and future work associated with the project.

### **7.1 Summary**

The goal of this project was to create a medium in which payment for transportation could be made in a seamless and simple manner. With respect to the core functionalities of the system, all were achieved. Although it was stated in the requirements chapter that additional functionality would be added when the core functionalities were completed, they were not implemented. However, they will be added to future work concerning this project.

### **7.2 Limitations and Challenges**

One major challenge that was encountered during this project was understanding the inner workings of Java Spring Boot such as the need for repositories and configuring the pom.xml file which is used for dependency management. Another major challenge encountered during the project were the multiple alternatives to perform a single action in AngularJS. This was very confusing because as a novice to AngularJS it was very difficult to determine the best way to achieve an objective.

One other challenge encountered during this project was the lack of an existing database that contained all public buses, ‘tro-tro’ routes as well as their associated bus stops. Hence,

One limitation of the developed system is that public transport users will have to buy transport credit from a retailer as preferred to the use of their debit card or mobile money. Furthermore, it would be more convenient if the system was connected to debit cards and mobile money to make payments smarter and more seamless.

Another limitation of the developed system is the inability of the vehicle owner to receive SMS concerning the transactions being made with the vehicle at timely intervals as compared always logging into the application to check on transactions being made.

### **7.3 Future Work**

#### **Mobile Money and Fintech Integration:**

One major addition that would further enhance the user experience of this application from the transport user's point of view is the addition and integration of mobile money from network carriers such as Tigo, MTN, Vodafone etc. and Fintech institutions such as SlydePay, ExpressPay etc. This would eliminate the need to buy transport credit from a retailer since many Ghanaians use at least one of the prior stated services.

#### **SMS Notification Integration:**

Another addition that would further enhance the user experience on the platform from the vehicle owner's point of view is the addition of timed SMS notification service. The user would be able to set his/her time interval preference thus enabling his/her to receive information concerning the vehicle's transaction via SMS. Thus, it would eliminate the need to always log into the application to view information about the vehicle's transactions.

## **7.4 Conclusion**

Overall, this project sought to push the ideal of a cashless economy by providing a means to pay for public transportation in a cashless fashion. It consists of two simple user interfaces namely a web application which is intuitive and simple to use as well as an inherently comprehensible cross-platform mobile application which is simple to use and can run on the three major smart phone platforms namely Android, IOS and Windows.

## References

- Akinola, O. S. (2012, August 8). Cashless Society, Problems and Prospects, Data Mining Research Potentials. *International Journal of Computer Science and Telecommunications*, 3(8), 49-55.
- Ghana Web. (2012, July 12). *Ghana to spend US\$30million on New GH¢50 Notes*. Retrieved from Ghana Web: <https://www.ghanaweb.com/GhanaHomePage/business/Ghana-to-spend-US-30million-on-New-GH-50-Notes-244464>
- Gjerding, K. (2017 , October 24). *The Cashless Economy: Why And Where It's Evolving And What Businesses Can Do Now To Prepare*. Retrieved from Forbes: <https://www.forbes.com/sites/forbesfinancecouncil/2017/10/24/the-cashless-economy-why-and-where-its-evolving-and-what-businesses-can-do-now-to-prepare/#4ffbf8d93e11>
- Issahaku, H. (2012). Challenges of Electronic Payment Systems in Ghana: The Case of e-ZWICH. *American Journal of Business and Management*, 87-95.
- Sun, B. (2009, June 9). *A multi-tier architecture for building RESTful Web services*. Retrieved from IBM: <https://www.ibm.com/developerworks/web/library/wa-aj-multitier/>