AN EFFICIENT FARE PAYMENT SYSTEM FOR PUBLIC TRANSPORT

Student Names: Aruasa Caesar Kipkoech and Kibe Caldwell Wachira.

Student Numbers:15129 and 150839

An Informatics and Computer Science Project Proposal Document Submitted to the School of Computing and Engineering Sciences in Partial Fulfilment of the Requirements for the Award of a Degree in Bachelor of Informatics and Computer Science

Submission Date: July 2024

Declaration

We declare that the project proposal has not been submitted to Strathmore University or any other University for the award of Degree in Bachelor of Science in Informatics and Computer Science.

Student Name: Aruasa Caesar Kipkoech

Student Number: 151729

Signature: _____

Date: _____

Student Name: Kibe Caldwell Wachira

Student Number: 150839

Signature: _____

Date: _____

Supervisor Name: Dr. Kennedy Ronoh

Signature: _____

Date: _____

Abstract

In a country where most people use public transportation, there is need for an efficient automated fare payment system. SACCOs require a platform that will enable a seamless collection of fares from passengers to increase customer satisfaction and overall experience.

Cash-based payment systems which is being used currently often lead to overpayment, underpayment, and a lack of accountability by passengers and conductors which in turn lead to loss of revenue by SACCOs. Cash-based payment also lacks a fare structure therefore estimation of fares and hence a lack of transparency.

The solution involves the development of a mobile application for automated payment in matatus. The application was developed using appropriate technologies and frameworks like flutter, firebase, and dart. Rigorous testing was conducted to identify and resolve any issues before deployment.

The mobile application provides a user-friendly interface for passengers and conductors enabling seamless and real-time transactions using STK prompts. This will improve the quality of service provided, accountability and increase revenue to SACCOs. The transition to an automated fare payment system promises a more reliable and efficient public transport system that benefits both the passengers and the SACCOs.

Keywords: Matatu, Passengers, Conductors, SACCOs, Automated Fare Payment System, Revenue

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CI/CD- Continuous Integration and Deployment

OTP- One-Time Password

RAD-Rapid Application Development

SACCOs-Savings and Credit Cooperative.

STK-Sim Application Toolkit.

Chapter 1: Introduction

1.1 Background

The public transport industry in Kenya faces various challenges, amongst them being fare collection. Currently, most public service vehicles use cash-based fare collection systems which are vulnerable to challenges such as underpayment, overpayment, lack of payment, and lack of accountability of fare collected by conductors that consequently lead to loss of revenue.

The fare collection problem affects matatus who travel short distances within cities and towns. This problem impacts matatu SACCOs by lowering the quality of service being provided to the customers or by affecting the revenues of the SACCOs.

Matatus provides means of transport to many Kenyans daily since they are mostly available and have fair charges. The matatus usually have a conductor who manually collects cash from passengers or asks them to send a certain amount of money via their phones but with these modes of payment, a passenger may be forced to pay more without knowing or may pay less to steal from the matatu. Some passengers often overpay or underpay their fares due to the estimation of prices according to distances because of the lack of a proper fixed fare payment system that informs passengers in case of any changes. There has been a rise in cases where some passengers create fake transaction messages which show that they have paid yet they have not(*Matatu Driver Cornered After Faking Mobile Money Texts to Petrol Station for Months - Kenyans.Co.Ke*, n.d.). There have been scenarios where passengers who try this and are caught are thrown out of a moving vehicle which is very unfortunate. Also, these modes of payment lack standardized fare charges in that a conductor may ask different passengers to pay a different amount even though they have travelled the same distance. Transparency is key in any organization for it to be successful. Without transparency, there is mistrust and dissatisfaction from the users of the organization. .

The research was carried out by a student from The University of Nairobi where they gave questionnaires to operations managers in different SACCOs in Nairobi on whether they would switch to electronic payment and most of them preferred to switch which shows that the current cash-based system is not efficient. (Mageria, n.d.)

There is a need for the development of proper automated fare systems that use STK prompts to address this problem comprehensively. This automated fare system will ensure transparency in the matatu industry hence improving service delivery to passengers and improving revenue collection and management by SACCOs(Macharia, 2017). It promotes accountability in fare transactions and a consistent fare structure where passengers are informed in case of any changes in matatu fare charges. It also improves security in matatus since passengers and conductors will not have to carry around loads of cash leaving passengers vulnerable to robberies

Various SACCOs have tried to enforce a cashless fare system due to losses that owners say amount to more than 30 percent of their daily earnings. However, they have encountered various setbacks during setting up proper systems that enforce cashless fare payments. (*Eastlands Matatu Saccos Enforce Cashless Fare System*, 2020)

Implementing this automated system will benefit and transform the matatu sector in Kenya leading to a safer, more reliable, high-quality, and efficient industry.

1.2 Problem Statement

The Matatu industry in Kenya encounters several challenges associated with cash-based fare systems that hinder efficiency, reliability, and quality of service provided(*Transport Sector Stares at Sh50bn Revenue Loss in Short-Term*, n.d.). This problem impacts the operations of SACCOs leading to loss of revenue and low-quality service provided. The cash-based system leads to underpayment and overpayment of fares by passengers due to the estimation of prices and lack of a standardized fare structure. It also expedites a lack of accountability by conductors since it is difficult to verify fare payments leading to a loss of revenue due to unaccounted fare payments. Security risks are apparent when using cash payments since passengers and conductors carry large amounts of cash increasing their vulnerability to robbery incidents. The lack of an automated fare payment system leads to inconsistent fare charges for example by informing passengers before the trip that fare has been hiked which may lead to a negative reception and hence passengers pay different amounts of money for the same distance travelled(*Matatu Fare Control*, 2023). Cash-based fare systems impede efficient, secure, consistent, and high-quality services by the matatu industry and therefore there is an urgent need to develop and implement an automated fare payment system to facilitate the matatu industry to improve SACCOs and passenger experience.

1.3 General Objectives

To develop a mobile application that will ensure an automated fare payment system in matatus.

1.3.1 Specific Objectives

a) To identify current challenges associated with cash-based fare payment systems in the matatu industry.
b) To provide real-time data access to users for example bus schedules.
c) To develop a mobile application that will ensure an automated fare payment system in Matatus.
d) To test the system.

1.3.2 Research Questions

a) What are some of the current challenges associated with cash-based fare payment systems in the matatu industry?
b) How effective is the implementation of an automated fare payment system using STK prompts in addressing the challenges caused by cash-based fare payment systems?
c) What are some of the strategies that can promote the adoption of automated fare payment systems across various matatu SACCOs to improve service delivery?
d) What are the key functionalities and features that should be incorporated in the development of a mobile application to ensure an effective automated fare payment system in matatus?

1.4 Justification

The current fare payment methods in public transportation around the country are plagued with numerous problems, especially in the country. These problems include slow processing of payments, inflated costs of operation, unrequited payment from passengers, and limited accessibility for some passengers. An efficient fare payment system would go a long way in significantly reducing the issues mentioned, offering a more secure, user-friendly payment system that expedites processing payments. (*Cashless Transportation*, 2019)

The application would improve user satisfaction for both passengers and conductors, reduce unnecessary overhead costs when collecting the fare, improve payment feedback, and automate the process of paying fare which will save time. Many times, you would find some passengers paying less and others paying more cause of a range of factors such as ‘pretty privilege'.

This phenomenon would cause preferential treatment for individuals who are perceived attractive leading to inconsistent fare payments due to negotiated discounts, or arbitrary decisions from the conductors to favour these individuals leading to unrequited payments.(*What Is "pretty Privilege" and Why Is It Such a Problem? | My Imperfect Life*, n.d.)

With advancements in mobile technology being made in the digital space, integrating these technologies in the public transport fare payment system would not only modernize the transport sector but align the sector with digital trends. This goes a long way in increasing its resilience against fraud, theft, and basic corruption(Brian, 2024). The application would reduce staffing and costs by reducing the need for manual processing of tickets and receipts which would often, be subjected to financial losses due to fraud or human error. This would increase the revenue generated by public transport, which minimizes since collection is automated and the reduction of fare evasion by passengers.

Doing away with the physical handling of cash and the need for paper-based tickets and receipts would ensure environmental cleanliness and sustainability. This would streamline operations and promote a cleaner transport system, hence improving the sector(Doynova, 2022). An automated fare payment system can collect information quicker, offering valuable insights that would help transportation authorities analyze peak usage, and adjust travel schedules, various passenger demographics, route planning, and travel patterns. The benefits are multifaceted, ranging from economic benefits to operational efficiencies.

1.5 Scope and Limitations

1.5.1 Scope of the Project

The application covers the development and design of a digital fare payment system. It integrates features like viewing transaction history through analytics, integrating the use of a mass STK push, and balancing top-ups. It also includes an easy-to-use user interface that will help the users navigate through the application without many problems. The application offers a customer support module that seeks to assist end-users through troubleshooting manuals and guidelines and real-time chat support. However, the application does not support broader transactions such as credit services, and general bank services. It mostly focuses on digital payments and does cover the use of physical devices such as fare gates.

1.5.2 Limitation(s) of the project

The application is subject to problems such as security risks and constraints in that ensuring information protection and security for the end-users poses a great challenge, due to it requiring advanced and sophisticated encryption. Users of the system adopting to the use of the application would be difficult, due to the masses being accustomed to traditional payment methods. Appending multiple functionalities that may be too sophisticated for the user may impact the application's adoption.

Chapter 2: Literature Review

2.1 Introduction

This literature review explores the challenges faced by passengers in matatus while paying their fares and by SACCOs while conductors collect fares. It analyses the existing fare payment system and the existing challenges. The review aims to provide a deeper understanding of an automated fare payment system that uses STK prompts and solves most of the problems being faced currently. It also highlights the impacts it will have on both the passengers and matatu SACCOs.

2.2 Challenges faced by the existing fare payment system in Matatus

This section reviews some of the existing problems faced by cash payment systems present in the transport sector and solutions to mitigate some of these problems.

Currently, the matatu industry in Kenya mostly uses a cash-based fare payment system and one of the objectives is to identify the challenges associated with this system. Firstly, cash transactions often lead to overpayment or underpayment because of estimating fares, lack of a standardized fare structure, or even cunning conductors. Cash transactions also lead to inconsistent revenue collection and a lack of accountability by the conductors (*Transforming Public Transport in Africa: Are Automated Fare Systems the Answer?*, n.d.). Fraudulent passengers know that payment confirmation is difficult with cash transactions, and they use it to deceive conductors. This may also lead to unrequited fare payment and on a larger scale,loss of revenue to the transport sector.

Another challenge being faced in matatus is that some passengers are unsatisfied with their matatu experience, and this is mostly caused during payment. Most passengers do not feel like they are paying the correct price due to estimating of fares and this is because there is a lack of a defined fare structure. Also, the fare payment process is not streamlined hence takes a lot of time to transact the payments and this can be solved by an automated system.

Furthermore, communication to the passengers on matters like a change in the fare prices does not exist and this brings a lot of confusion.

The automated fare payment system as implemented by the developed application will enhance proper communication channels where users are notified earlier in case of any changes in prices. This will significantly enhance user satisfaction, streamline transactions, and improve transparency hence general passenger experience will be improved.

The existing fare payment system also leads to loss of revenue by matatu SACCOs since there is no record of transactions and most of the conductors lack accountability. The use of a cash-based system means that transactions are not documented enabling dishonest conductors to take advantage and pocket some of the fares collected. This makes it challenging for matatu SACCOs to keep track of their actual revenue and there is need to adapt an automated system to ensure all transactions are recorded and increase accountability by conductors.

2.3 Existing solutions

2.3.1 O-City application

O-City is a mobile application to digitize fare payment in the public transport sector in Kenya. O-City primary purpose is to ensure a seamless fare payment process in the public sector, improve transparency, and improve user experience.

O-City has 3 main users: passengers who use the app to pay for their fares using their phones, the conductors who use the app to track fare payment by passengers, and matatu SACCOs who use the app to monitor their revenue and hold conductors to account in case of any discrepancy. To understand O-City even better, it was necessary to enrol as a user of the application.

2.3.1.1 Pros of the O-City application

Provides cashless fare payment for passengers.

A user can check their balance.

Real-time tracking of transactions.

Easy to register as a user.

2.3.1.2 Cons of the O-City application

Difficult to operate the application for a new user.

Only available in one city in Kenya (Nairobi).

Adoption by SACCOs may be difficult because of the cost.

User education is needed.

Having a username is not mandatory which will make validation of payment by the conductors difficult since one cannot be identified using a phone number.

2.3.1.3 Critique of O-City

O-City has provided solutions to address the problems faced by matatu passengers, conductors, and SACCOs due to a cash-based fare payment system. However, it is difficult to ensure that SACCOs adopt it because of the costs associated with the application. Also, the application is not well known by many people, and it is only available in one city in Kenya (Nairobi). I also noticed that so long as one inputs their phone number they are set, and having a username is optional. This makes confirmation of payment by conductors difficult.

For a new user, the application is not easy to navigate through as it has many complexities. These are things that can always be improved on.

2.3.2 Masabi application

Masabi application is a mobile application that provides a comprehensive fare payment system through a platform known as Justride. Justride enables ticket purchases and transaction validations via smartphone, enhancing convenience for users.

The application offers account-based ticketing, where the Justride platform manages accounts and offers pre-purchased fares. It offers Open Payments through contactless payments such as Mastercard and Visa payments. Also, it offers cloud services that ensure high reliability and scalability, accommodating the needs of passengers and transit agencies.

2.3.2.1 Pros of the Masabi Application

It supports multiple fare collection methods through mobile ticketing and contactless EMV payments.

The platform is user-friendly enhancing usability and convenience.

The platform is secure which is crucial in handling sensitive personal and financial data and transactions.

2.3.2.2 Cons of the Masabi Application

Transitioning to a new fare collection system for new users could be difficult.

It is heavily reliant on internet access, limiting access to its services.

The users require a level of digital literacy which may be inconvenient for some end-users.

2.3.2.3 Critique of the Masabi application

The Masabi Justride platform streamlines and secures fare payment and collection in public transport. It offers a robust solution to physical payment and handling of fares. Its cloud-based module offers significant advantages in scalability, flexibility, and cost efficiency. However, it is riddled with issues such as reliance on consistent internet access.

2.3.3 Moovit Application

Moovit is a mobile application that serves as an automated fare payment system. It integrates the capability of mobile ticketing which enhances user experience for people in transit. The application offers a platform for users to plan and pay their transportation needs at a go.

It also supports contactless payments across various multimodal transport platforms such as buses, trains, and metros.

2.3.3.1 Pros of the Moovit Application

Offers real-time planning, fare payment, and mobile ticketing.

It has a seamless user interface that allows passengers to book their transit in an effective manner.

It is offered in a great number of countries and cities across the world.

It offers contactless payments.

Offers accessibility features for the disabled.

2.3.3.2 Cons of the Moovit Application

Moovit relies on real-time data hence users must have consistent internet access which may be an inconvenience to some end-users.

It may seem complex to new users due to the multitude of features it displays.

Potential privacy concerns.

2.3.3.3 Critique of the Moovit Application

Moovit is a revelation in the MaaS(Mobility as a Service) landscape. It offers a multitude of advantages to daily commuters, amongst them being that they can plan their transit via the mobile application. It spans various modes of transport which makes it convenient to the user. It also accommodates the disabled by offering accessibility features, which makes it an all-encompassing application.

2.4 Gaps in the existing application/Solutions

2.4.1 Gaps

Difficult for a new user to operate. The applications have a lot of complexities which may take some time to understand.

Only available in one city in Kenya. (Nairobi) for O-City. The application can only be used within Nairobi city yet matatus are found in every town and city in Kenya.

Having a username is not mandatory. The application only asks for a user's phone number and has not put in the necessary measures that will enable a conductor to know who has paid.

2.4.2 Solutions to these gaps

Implementing a minimalistic design that ensures the proposed application is user-friendly and easy to use.

The proposed application should be accessible beyond Nairobi, extending to other cities and towns.

By incorporating a registration process in our proposed application where the end-users can create a unique identifier that will differentiate the passengers with each transaction, the system can effectively verify payments.

2.5 Conceptual Framework

The proposed project will have several key components as shown below. The automated fare payment system will have a user login to the application or register if they are new. The system will have an intuitive user interface so that when the user logins or registers they find it easy to interact with the application.

The passenger will see their fares being calculated and adjusted based on factors such as travel distance and route covered. After calculations, the passenger will interact with a payment processing system which will facilitate fare collection as well as track payments.

In addition, it also has a customer support component where customers can raise issues and give feedback to the admins who will be able to see, and this increases user satisfaction. Moreover, it has a database component where all relevant data like the transactions made will be stored. It ensures that there is a centralized data source for the proposed system and data integrity is guaranteed. The database can be accessed by the admins who are the matatu SACCOs. They will be able to track the revenues collected by querying from the database. The admins are also able to see fares being calculated and adjusted based on travel distance and routes used which increases accountability.

The system also integrates with third-party services such as the 'MPESA Daraja API' to facilitate the payment of fares by passengers enhancing the system functionality and overall usability as seen in Figure 2.1

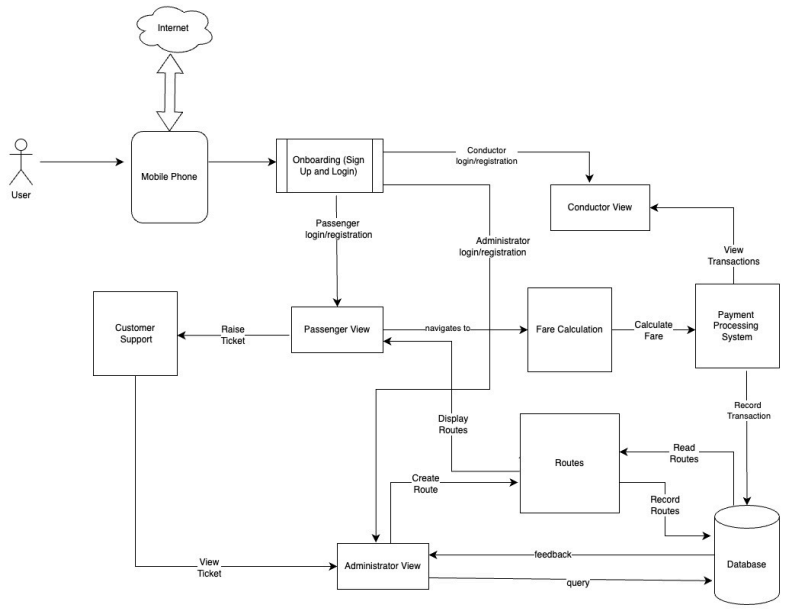


Figure 2.1 Conceptual Framework

Chapter 3: Development Methodology

3.1 Introduction

This chapter will highlight and expound on the methodology adopted to develop an efficient fare payment system for public transportation. The development approach of the proposed project will be structured in a way that ensures the system is secure, scalable, and user-friendly for both daily commuters and transport administration. The chapter will outline the software development life cycle of the proposed project and will include stages of requirements elicitation to the deployment of the application and maintenance. On top of that, this chapter will also highlight some of the tools and techniques deployed and utilized during the development process, alongside the expected deliverables of the project.

3.2 Software Development Methodology

Software development methodology is a series of steps and processes that are used in software development. The project incorporated the use of Rapid Application Development (RAD) methodology (Bondar, 2021). This methodology is centered on facilitating the rapid development of software by building and refining software in sequences of iterative cycles. This methodology was suited to the environment of the project which heavily experiences change due to technological advancements, varying regulative requirements from the government, and evolving stakeholder expectations.

RAD can facilitate immediate feedback and continuous adaptation since it involves quick iterative releases and prototyping, which is crucial for the development of our project and involves complex interactions such as payment processing and a corresponding management system. Employing RAD will ensure the system is robust and adaptable to varying expectations and demands. As seen in Figure 3.1

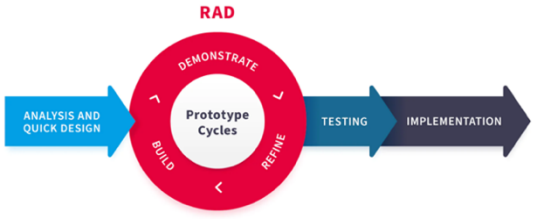


Figure 3.1 Rapid Application Development

3.2.1 Requirements Elicitation

The initial phase in developing an efficient fare payment system involves a panoramic requirements elicitation using the RAD approach. Surveys and questionnaires were distributed amongst passengers, conductors, and people who are accustomed to using public transport to gather a broad spectrum of their attitudes, feelings, needs, and expectations towards the current fare payment system and the idea of an automated fare payment system. This phase also included collaborative workshops that involved stakeholders, actively participating in developing initial prototypes, and offering immediate feedback.

3.2.2 User Design

This phase involved users interacting closely with system analysts to develop prototypes and models that represent system processes, inputs, and corresponding outputs. In this phase, the creation of data flow diagrams and process models took place to ensure that the system design aligns with user needs and expectations.

3.2.3 Rapid Construction

The rapid construction phase focused on building a working model of the automated fare payment system. This phase ensured the development of key features and functionalities by prioritizing functional capabilities and progress over the initial flaws of the application. The system employed the use of rapid prototyping and UML diagrams to ensure clarity.

3.2.4 Developmental Environmental Setup

It was essential to establish a robust development environment. This phase involved setting up the necessary platforms and corresponding tools to facilitate quick iterations. It included developing, testing, and staging appropriate servers, appending the necessary extensions, and ensuring that version control systems were running smoothly. This helped support the development process and deployment by establishing a good framework for use.

3.2.5 Rapid Prototyping

This phase extended rapid construction, dividing the development into short continuous cycles that last between a period of 1-2 weeks. Each cycle focused on developing and refining the functionalities and specific features based on stakeholder feedback and system testing.

This iterative process will ensure the proposed application will evolve quickly and will also cater to evolving user needs and system requirements.

3.2.6 Coding and Development

This phase involved the actual creation of the fare payment system based on the outlines approved in the previous phase, system design. The use of developmental frameworks and programming languages were used for the project. There was incorporation of iterative testing and debugging to ensure code quality and alignment with the project's objectives.

3.2.7 Testing and Deployment

This final phase involved a comprehensive testing of the system to ensure that it satisfies all the user requirements and functions properly. The development of the project includes activities such as system testing, debugging, and user acceptance testing. After thorough testing, the application underwent a cutover process(*Mastering the Art of Cutover Planning: A Step-by-Step Guide* -, n.d.). This helped ensure smooth operation and deployment of the final application.

3.3 System Development Tools and Techniques

This section will highlight some of the tools and techniques that will be used to develop the application.

3.3.1 Flutter

Flutter was used as a development framework for building the fare payment system. It is an open-source software development kit created by Google that allows you to build native natively compiled cross-platform applications(*Pros and Cons of Flutter App Development*, 2023).

3.3.2 Firebase

The proposed application will use Firebase for the backend. It will manage user authentication, real-time databases, analytics, and other backend needs that are crucial for mobile applications.

3.3.3 Dart

Dart will be used as the programming language. Dart works hand in hand with Flutter to facilitate the development of a program that is suited for UI composition.

3.3.4 Test-Driven Development

The development of the application utilized Test-Driven Development (TDD), which involves the creation of test cases before coming up with the actual code that needs to pass the test(*What*

Is Test-Driven Development? | TestDriven.Io, n.d.). This made sure that the system was reliable and bug-free during the early stages of development.

3.3.5 Continuous Integration and Deployment

The development of the project utilized Continuous Integration and Deployment (CI/CD) that involves using tools such as GitHub Actions and GitLab Cl. The development of the project involved the integration of this technique with Flutter, the development framework for the application to automate testing and deployment processes.

3.3.6 Refactoring

Development of the application utilized Refactoring to develop the application. This involved improving the design of the existing code without affecting its external behaviour(*Refactoring Home Page*, n.d.). Regular refactoring will ensure the code is clean and manageable, which will help in the scalability of the program.

3.4 Deliverables

3.4.1 User Account Management System:

The application has a User Account Management System module that will serve as the foundation for user interaction within the system. It allows users to register, log in, and manage their profiles. This was crucial for ensuring the management of users and their personal information.

3.4.2 Automated Fare Payment System

The application includes an automated fare payment system to dynamically ensure each passenger has paid the adequate amount for each trip. Conductors have a view that enables them to see each transaction and accurately determine who has paid and who has not paid.

3.4.3 Real-time data

A real-time data and scheduling module will be provided to keep users up to speed with up-to-date information on routes and price fluctuations. This included a page that displays all the routes that each matatu uses during the transportation of passengers from one place to another.

3.4.4 Administration and Reporting

An administration and reporting module, that will provide a dashboard for the system administrators to oversee the entire system. This module will be used to monitor user activities, manage routes, and generate detailed analytics of the system.

3.4.5 Final System Documentation

A final system documentation module that comprehensively details descriptions of the application's modules, architecture, and capabilities. It also includes a complete guide for both end-users and system administrators, instructing them on how to navigate through the system from both views, respectively. This assisted the deployment of the application and efficiently handled operational issues.

Chapter 4: System Analysis and Design

4.1 Introduction

This chapter will discuss the various components of the system identifying both the functional and non-functional requirements of the system. It will also discuss the methodologies used in the development of the mobile application. These methodologies will be represented in the form of diagrams, system analysis, and system design diagrams.

4.2 Software Requirements Analysis

This is the process of determining user expectations for a certain product and these expectations are met by functionalities in the system(*Activities Involved in Software Requirement Analysis - GeeksforGeeks*, n.d.). The functionalities required in the system are grouped under functional and non-functional requirements and are stated below.

4.2.1 Functional Requirements

Functional requirements specify the behaviour of a system, describing its features, functions, and its operations(*Functional Requirements Examples and Templates*, n.d.).

The application includes an account management system that has two views: the admin side and the user side. The admin incorporates analytics based on the number of passengers a day, the total fare paid today by the passengers within a specified period, and other analytics. The user view incorporates real-time data access such as bus routes used. This maintains high levels of convenience for users, enabling them to plan their journeys effectively.

A fare calculation module that adjusts fares based on prevailing circumstances such as travel distance and selected routes. This will help in accurately computing fare from the given criteria.

The system includes a payment processing system that will incorporate the payment gateway Mpesa via an STK push. This will enhance the convenience of transactions and will make sure everyone pays the same amount as required.

A customer support feature that will ensure that the stakeholders can easily raise issues and make inquiries regarding the system. This will help the users receive assistance in a prompt fashion which is key in maintaining user satisfaction.

The application includes a real-time data access feature that keeps the users up to speed with up-to-date information on updated bus routes and fare prices based on the existing routes. This proved invaluable in planning and efficiency.

4.2.1 Non-Functional Requirements

These are a set of specifications that describe the system's operation and its capabilities and constraints(*Nonfunctional Requirements*, 2024):

Security: The application will handle sensitive information such as personal and financial information. The application must implement encryption methods for data transmission and storage to ensure the system is not susceptible to potential breaches and vulnerabilities.

Scalability: The system should be able to accommodate growth in user numbers as well as transactions without any negative changes in performance.

Accessibility: The system must ensure that it is accessible to every passenger and all the users of the system.

Maintainability: The application should be designed in a way that will facilitate easy updates and overall maintenance of the stem. Incorporating a modular design that allows individual components to be updated without impacting the entire system, will be essential.

Reliability: The application must function as expected under normal circumstances and must be robust to maintain operational stability and prevent data leakage and loss.

4.2.2 System Narrative

A system narrative is an account of how a system operates from the perspective of an end-user. It describes the interactions and corresponding processes a user will have with the system(*High Level Design in System Design - Javatpoint*, n.d.). A new user will register for an account by entering their details. A One-Time Password(OTP) is sent to the phone number of the registered user and they are consequently redirected to the OTP verification page. After the onboarding process the user is able to access the payment page and other pages such as 'view routes'and the customer support page. When they enter a public vehicle, they will select the matatu they are in as stated in the public vehicle through a dropdown and enter the amount and phone number. They thereafter receive an STK prompt that warrants them to pay the fare. After confirmation of payment, the user is rerouted to a thank you page.

4.3 System Design

A system design is a key phase in the development process that involves defining the architecture, interfaces, and information for a system to fulfill specified requirements.(*What Is System Software? – Definition, Types, Examples and More*, n.d.)

4.3.1 Use Case Diagram

A use case diagram is a type of behavioral diagram that summarizes the details of a system's users and their interactions with the system. (*UML Use Case Diagram Tutorial*, n.d.)

Some use cases such as registration and login are common for all the actors in the system. The use cases associated to the passenger alone include viewing real-time data such as viewing bus routes and fare prices and making payment via STK prompt. Use cases associated to the administrator include managing user accounts by performing CRUD operations on all the users, managing fare rates by creating routes and monitoring the transactions made and revenue collected. The conductor is able to verify the payments made by the passengers by checking the payment status. As seen in Figure 4.1.

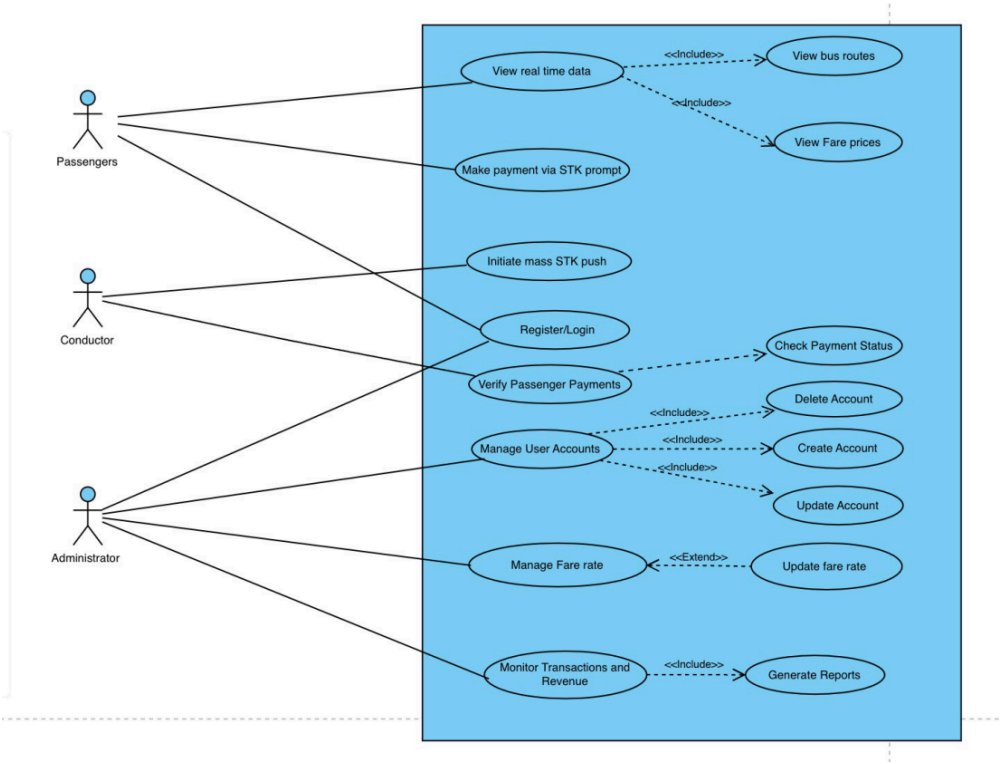


Figure 4.1 Use Case Diagram

4.3.2 Class diagram

A class diagram is a type of structure diagram in the Unified Modelling Language that describes the structure of a system, showcasing its classes, attributes, and relationships between different objects.(*What Is Class Diagram?*, n.d.)

The system has several classes that include users, admin, and transaction classes. The figure below shows the attributes and methods of these classes and their relationship.

The user class has the following attributes: user ID, name, email address, and phone number. Its methods include registering, login and viewing the route schedules.

The admin's class attributes include admin ID, name, and email address. Its methods are managing users, generating reports, updating transport schedules, and viewing transactions. Attributes in the transaction class are transaction ID, user ID, the amount paid, date of payment, and the status of payment.

Its methods are processing payments, verifying transactions, and also viewing the transaction history. Other classes include the fare calculator class and the payment gateway class. The attributes in the fare calculator class are fare ID, the distance of the trip covered, the route used and the rates of the trip.

Its methods are fare calculation and updating rates in case of any changes in fare calculation. The payment gateway class has the following attributes: payment ID and transaction ID. The only method in in payment gateway class is confirmation of payment. As seen in Figure 4.2

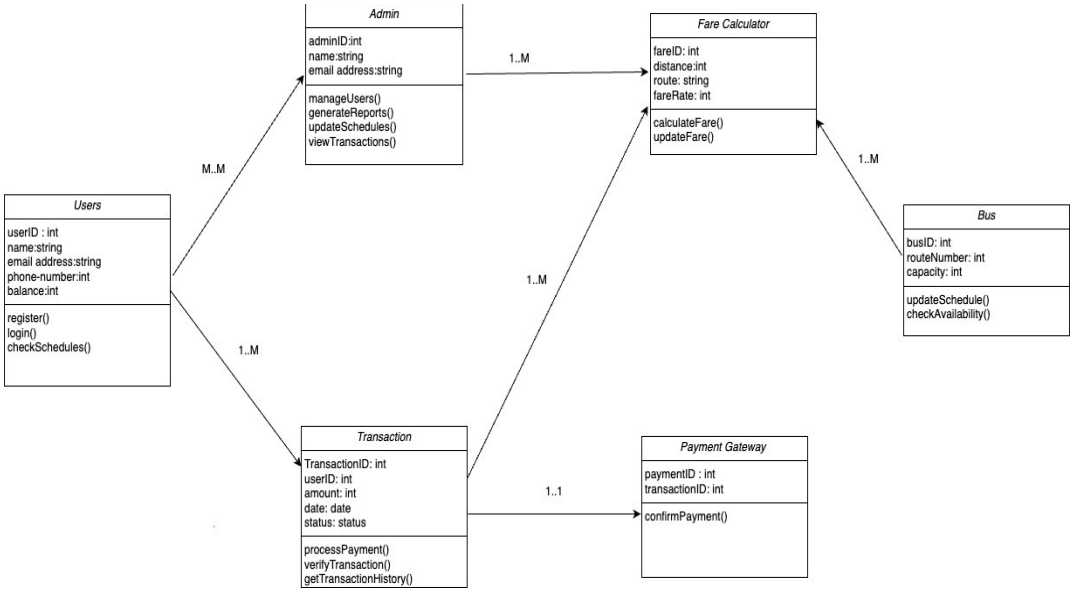


Figure 4.2: Class Diagram

4.3.3 Entity Relationship Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how "entities" such as people, objects, or concepts relate to each other within a system(*ER Diagram (ERD) - Definition & Overview | Lucidchart*, n.d.).

It helps visualize the structure of data flow and how different data elements interact within a system. In the context of the developed application, the diagram below will illustrate the key entities such as Transactions, Conductors, Vehicles, Fare rates, and their corresponding relationship with each other. A passenger represents a user using the transportation application service. A transaction captures the details of each payment and maps passengers with fare rates. The payment gateway facilitates the processing of each transaction made by each passenger, ensuring secure handling of electronic payments via Mpesa. The admin is an individual responsible for managing users of the application and each matatu. The fare calculator module ensures equitable fare prices are set based on routes used. As seen in Figure 4.3

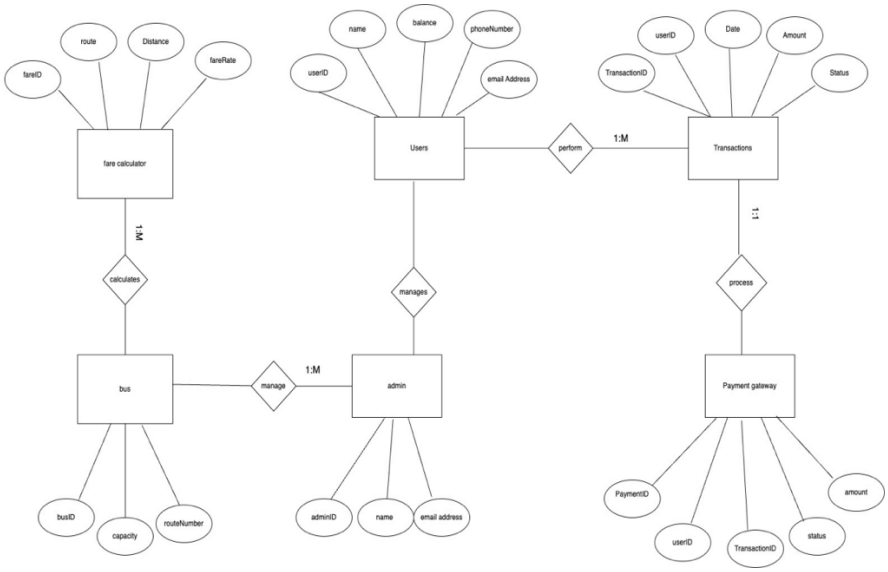


Figure 4.3: Entity Relationship Diagram

4.3.4 Database Schema

A database schema is the blueprint of a database that defines how data is organized within a relational database; inclusive of logical constraints such as table names and fields between these entities. (*What Is a Database Schema?*, 2024).

The schema consists of six tables, users, transactions, payment gateway, admin, bus, and fare calculator. There is a one-to-many relationship between users and transactions indicating that a single user can perform multiple transactions across the platform. There is a one-to-one relationship between transactions and the payment gateway, indicating that each transaction is processed through the Mpesa payment gateway. There is a one-to-many relationship between users and payment gateway, indicating that a single user can have multiple transactions processed through the Mpesa payment gateway.

By structuring our database in this format, we were able to ensure effective handling of user data and corresponding transactions in our system as seen in Figure 4.4

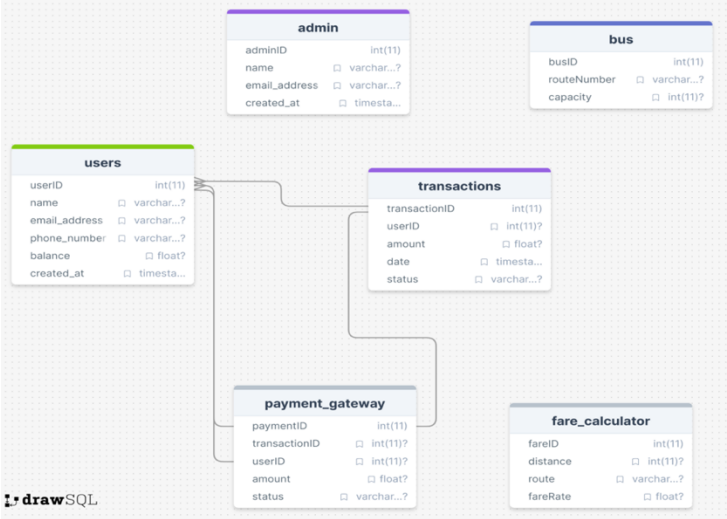


Figure 4.4: Database Schema

4.4 Project Wireframes

A wireframe is a top-level blueprint that illustrates the skeletal framework of an application. It demonstrates how elements in an application relate to each other and how they are structured.(*What Is a Wireframe & Its Role in the Design Process | Miro*, n.d.)

The passenger, administrator, and conductor all have a uniform registration, login, and verification page. An already-registered user logs in whereas unregistered users are redirected to the registration page.

The users receive an OTP to verify their phone numbers. Once a passenger has been verified, they are redirected to the passenger landing page where they can view routes, make payments, select seats, and give customer feedback.

An administrator is verified and redirected to the admin landing page where they can create routes, calculate fares, check the payment history, manage all system users while performing basic CRUD functionality, and view the feedback provided by the passengers.

The conductor once verified is redirected to the landing page and can verify the payments made by the passengers by viewing the payment history. As seen in Figure 4.5

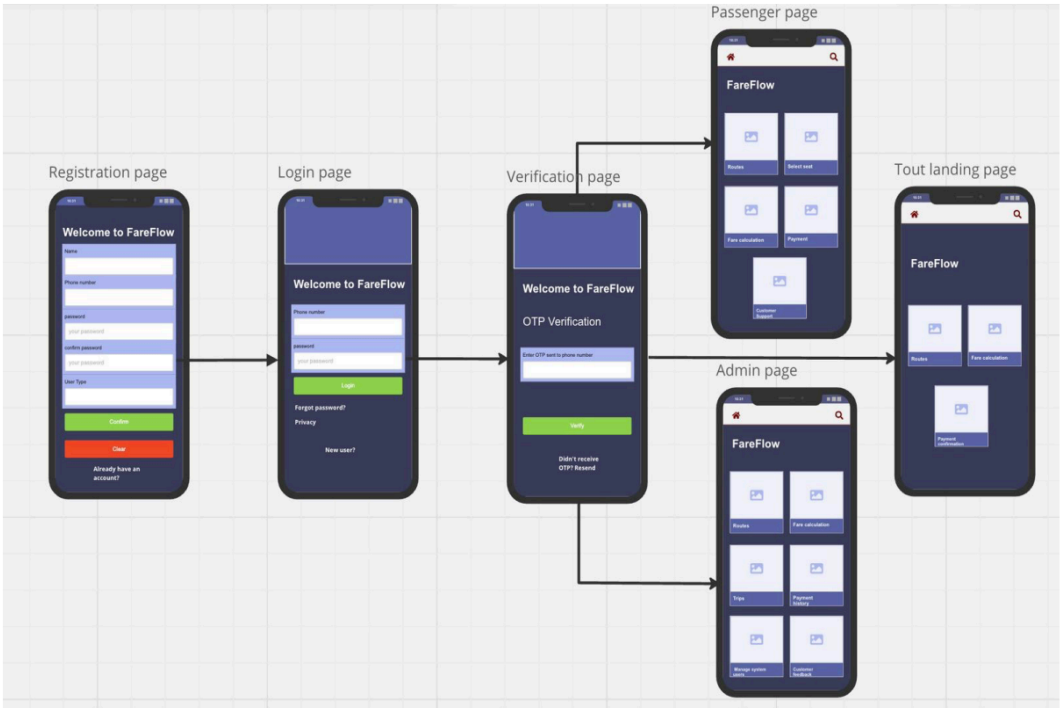


Figure 4.5: Wireframes

4.5 System Architecture

This is a conceptual model that defines the structure, behaviour, and different views of the system. It sets a foundation of how the developed application is structured.(*System Design and System Architecture | by Brandon Kang | Bootcamp*, n.d.)

The diagram illustrates how users input information into their phones and get information from their phones. The conductor once logged in can only view the system. The passenger and the administrator can view the system but input different required information such as payment and routes into the system. The whole system communicates with the database by requesting and storing data. Also, the system uses Google Maps to create the different routes used by passengers.

It provides a definitive illustration of how the application operates under multiple viewing based on the actors of our application and how it orients itself to handle each view.

As seen in Figure 4.6

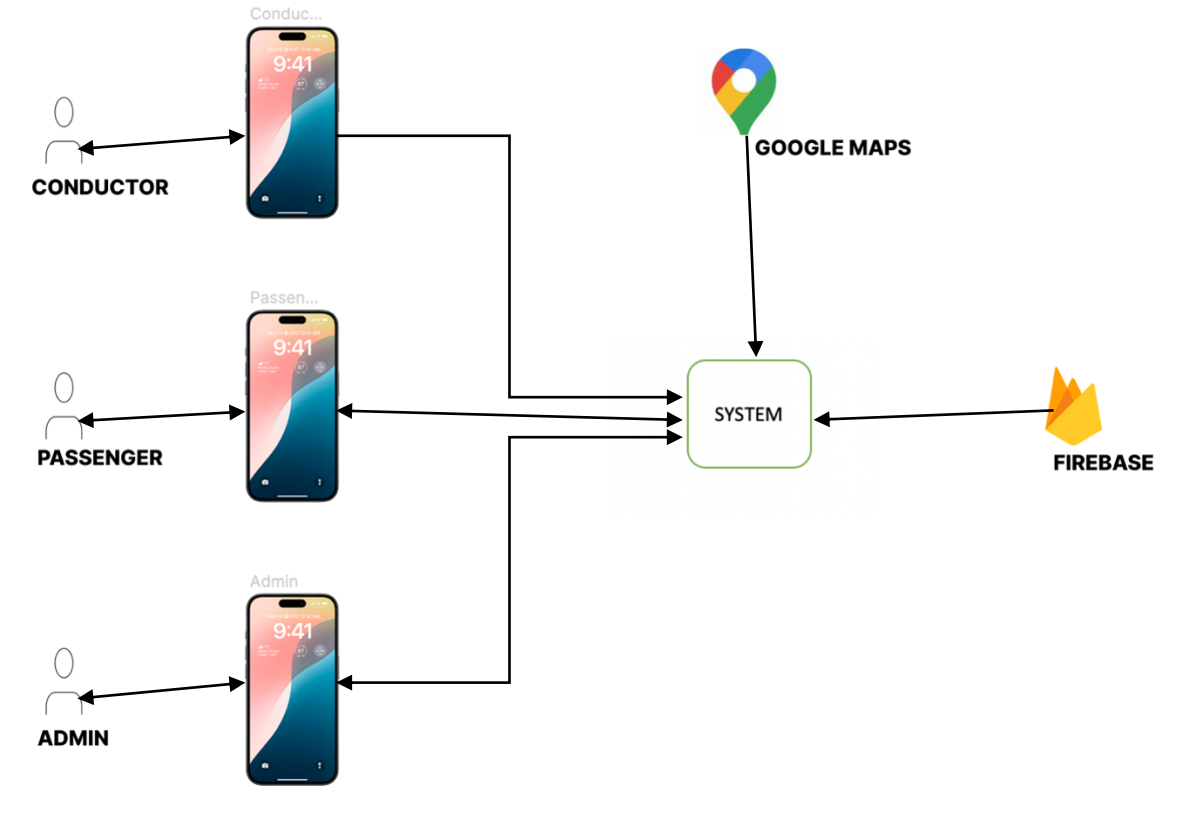


Figure 4.6 System Architecture

Chapter 5: System Implementation and Testing

5.1 Introduction

This chapter describes the practical aspects of implementation and testing for the automated fare payment system for Matatus. It covers the description of the implementation environment, the testing paradigms, and the implementation process.

5.2 Description of the Implementation Environment

This section describes the hardware and software specifications utilized during the testing and development of the application.

5.2.1 Hardware Specifications

These are the hardware's minimum technical requirements and configurations for the system.

These specifications ensured the smooth development, testing and deployment of our application. As seen in Table 5.1

Hardware	**Specification**
RAM	8 GB minimum
Screen Resolution	1280 x 800
Disk Storage	8 GB of available space
CPU architecture	ARM64
Processor	A7 chip and ARM Cortex-A53

Table 5.1 Minimum Hardware Requirements

5.2.2 Software Specifications

These are the software's minimal technical requirements and corresponding configurations for the system.

The operating system used to develop the system is macOS Sonoma 14.0. The application is intended to run on IOS devices with a minimum of iOS 11.0.

The system uses the Firebase Real-time Database, which is a NoSQL database that syncs data across clients in real-time even when the mobile application is offline.

The system utilizes Xcode as an integrated development environment to develop the IOS version of the application.

The system uses Android Studio which is an integrated development environment that is utilized to develop and test the Android version of the application.

5.3 System Implementation

This section describes the key modules tested and developed in the application.

5.3.1 Authentication Module

The Authentication module handles the registration of new users, OTP verification, and login

Firebase was used to manage user accounts and the creation and modification of accounts.

The user interface for this module was designed using Dart on a Flutter framework.

A user registers by writing their phone number and following the steps of OTP verification. This information is handled by Firebase.

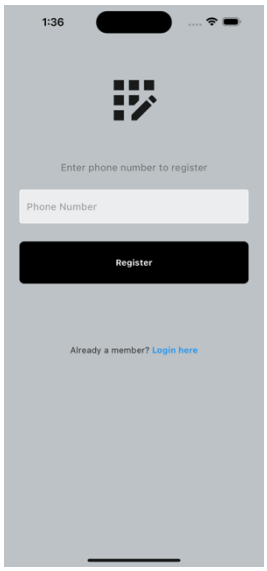


Figure 5.1:Register Page

A user logs in using his phone number. This process is optimized by the Firebase phone provider authentication module.

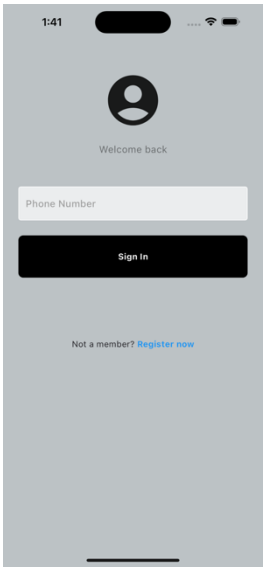


Figure 5.2: Login Page

The Feature, Tracking, and Reporting module allows stakeholders of the application to track features, report issues regarding the application, and provide concise feedback.

5.3.2 Grids Module

The Grids Module handles each view in the application. It assigns each actor to a view and displays the functionality of each. These actors constitute of

The passenger grid will display four grid components:

‘Routes’ which redirects them to the view-routes page, ‘Fare Calculation’ which redirects to the Fare Calculation page, ‘Payment’ which redirects to the payment page user will pay and ‘Customer Support’ which redirects the user to a page where they can raise tickets.

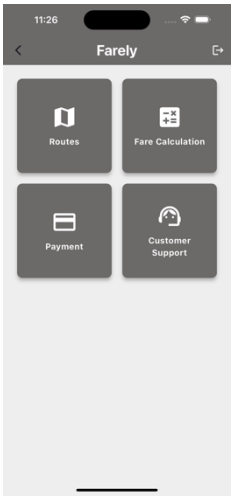


Figure 5.3 Passenger Grid

The conductor grid will display one grid component, ‘View Transactions’ that redirects the conductor to view each transaction made. It will redirect them to a page that displays all the transactions made and how they are mapped to their specific matatu.

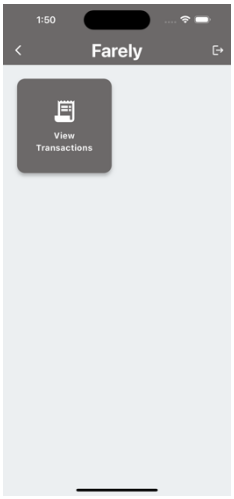


Figure 5.4 Conductor Grid

The administrator grid displays four grid components:

‘View Analytics’ redirects the administrator to a page that displays the feedback of the usage of the application. ‘View Tickets’ redirects the administrator to see each raised ticket from individual passengers. ‘Create Routes’ reroutes to a page where the administrator can create routes. ‘Create Matatu’ displays a page where the administrator can create matatus.

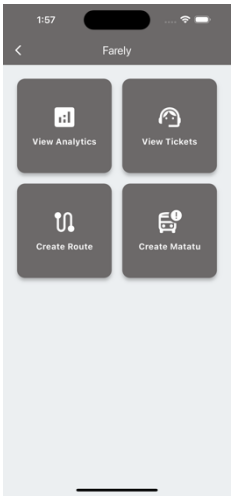


Figure 5.5 Administration grid

5.3.3 Routes Module

This module is used to create routes that the user can see. It uses poly points to indicate the distance between two routes. The starting location is entered in the first text box and the destination location is entered in the next text box. After clicking the button, a leaflet map appears and maps the input of the text boxes as poly points on the map.

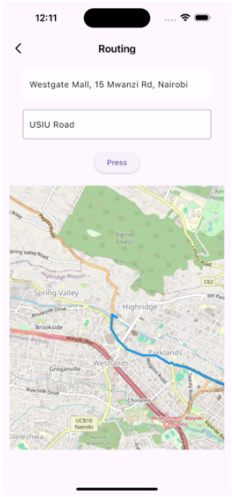


Figure 5.6: Create Routes

A user can view these routes and gauge the distance from their starting point to their destination point through polylines.

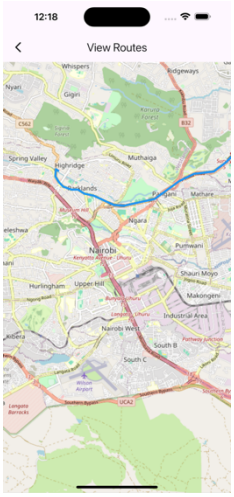


Figure 5.7: View Maps

5.3.4 Customer Support Module

This module is used to gather insights and feedback on the positives and drawbacks of the application from end-users. The user types a message on the text box and clicks the submit button. This information is consequently stored in the Firestone database.

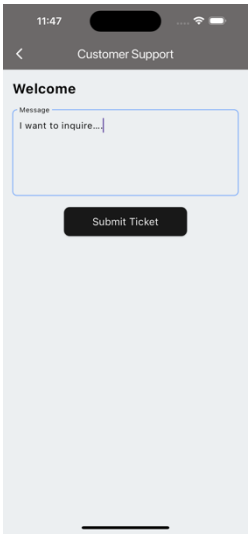


Figure 5.8 Customer Support Page

The admin can view these routes through the View Support Tickets page. Each ticket has a phone number section indicating who has raised, a message section indicating the feedback from the user, and the date and time that the user raised the ticket.

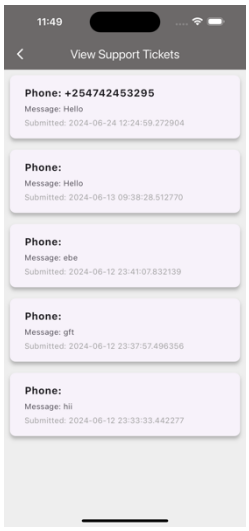


Figure 5.9 View Support Tickets Page

5.3.5 Payment Module

This module utilizes Mpesa as a payment gateway to confirm transactions made by passengers. The user selects the matatu they are in using the dropdown menu; it will read from the database and display all the matatus. They will enter the amount of fare to be paid on the next text box and key in the number they wish to pay. After providing all the necessary information, they will click the initiate payment button and consequently receive a prompt to their phone number to pay.

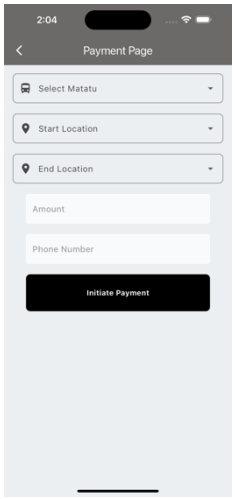


Figure 5.10 Payment Page

After payment, a loading circle will appear and will be eventually popped after the amount prompted to the passenger's phone has been paid. The passenger is thereafter directed to a thank you page that expresses gratitude to the passenger and displays a button to redirect them back to the Home Page.

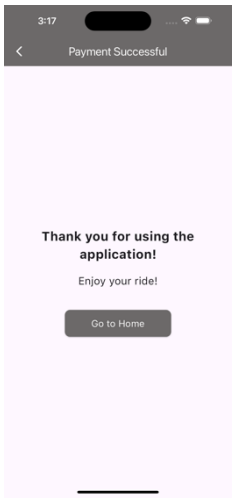


Figure 5.11 Thank You Page

The conductor side will be able to pull the transaction details from the database and will be retrieved and displayed in the form of a ticket. For each matatu selected by the user, a drop-down menu is displayed to indicate all the transactions mapped onto that matatu.

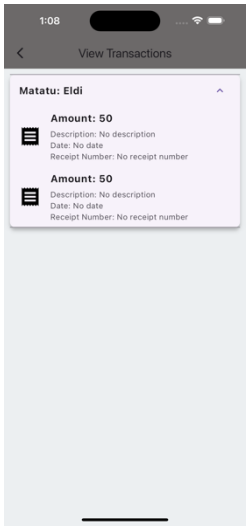


Figure 5.12 View Transactions Page

5.3.6 Fare Calculation Module

This module handles the calculation based on each route. On the payment page, as seen in Figure 5.10, the user is expected to enter a fare amount in the text field with the label ‘Amount’ after selecting their start location and their end location.

This is calculated by the Fare Calculation module on a separate page to help the user determine how much they must pay. It uses a function that passes the latitude and longitude coordinates as parameters, calculates the distance between the two locations, and multiplies them equitably.

The passenger is required to enter their start location and end locations in the dropdown menus displayed and click the button that indicates ‘Calculate Fare’. The fare amount is consequently calculated and displayed on the Card Widget below the button. The passenger keys in this amount on the payment page.

This module helps the user to determine how much amount to pay via the Mpesa Gateway, efficiently and equitably.

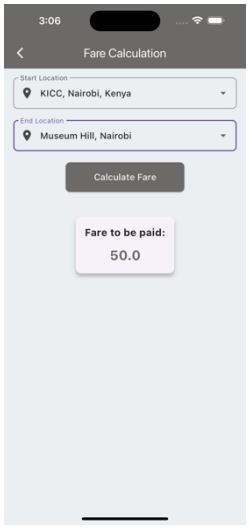


Figure 5.13 Fare Calculation Module

Figure 5.10 Payment Page

5.4 System Testing

System testing is a procedure that involves evaluating the performance and overall functionality of a fully integrated system. It tests if the system meets specified requirements defined by stakeholders of the application and if it is suitable for delivery and usage to end-users.(*System Testing - Software Engineering*, 2019)

5.4.1 Testing Paradigm

System testing was accomplished using both white box testing and black boxing.

5.4.1.1 White Box Testing

White Box testing is a form of application testing that provides adequate knowledge of the application being tested and reviewed through verification of the inner workings of the code, infrastructure, and integrations with external systems.(*System Testing - Software Engineering*, 2019.)

The application benefitted from white box testing because we employed Rapid Application Development. This enabled the employment of continuous improvements to our application.

5.4.1.2 Black Box Testing

Black Box Testing is another form of application testing where the tester of the application is not concerned with the internal knowledge of the source code and its implementation but rather

focuses on the behaviour of the system from a user's perspective.(*Black Box Testing - Software Engineering - GeeksforGeeks*, n.d.)

The application gained value from black box testing since it offered a more realistic and refined assessment of the application and its functional requirements from a user's perspective.(phanivedala, 2023)

5.4.2 Authentication Module Testing

The following table summarize the results of various test cases performed on the system. They are associated with the authentication module of the developed application. As seen in Table 5.2

Table 5.2 Authentication Module Test Results

Test Case	Description	Test Data	Expected Outcome	Actual Results	Test Verdict
1	Passenger registration by providing their phone number	A test number was used	User details are recorded in the database and the user is redirected to the opt verification page	Redirection to OTP verification page	As expected
2	Further Registration through a One Time Password	A correct six digit-OTP was used	User receive an OTP and follows the steps to confirm their verification and are redirected to landing page	OTP message is sent from Cloud OTP and user is redirected to landing page.	As expected
3	Further Registration through an incorrect One Time Password	A wrong six-digit test OTP was used.	Error message indicating OTP details entered are incorrect.	Loading circle stops and an error message is displayed	As expected

4	Logging in using a registered phone number	A test number that was already registered was used	Redirection to landing page after loading circle is popped.	The user is redirected to the landing page.	As expected

5.4.3 Payment Module Testing

The table below summarizes the results accrued from the test cases performed on the system application. It is associated with the payment gateway and processing of transactions. As seen in Table 5.3

Table 5.3 Payment Module Testing Results

Test Case	Description	Test Data	Expected Outcome	Actual Results	Test Verdict
5	Selecting a start location and end location , entering the key details and consequently paying the fare amount	Clicking the Drop-down for both the start location and end location, entering the amount to be paid and the phone number	A message appears indicating 'payment initialisation successful' and a prompt is sent to the phone.	Message indicating payment successful appeared, prompt sent to the user's phone.	As expected
6	Transactions save to database and can be viewed as receipts on conductor's side	Click on View Transactions button on the conductor's grid.	A List-View indicating all the transactions made for each matatu	A list of each matatu and their each transaction	As expected

5.4.4 Routes Module Testing

The table below summarizes the results accrued from the test cases performed on the system application. It is associated with the creation of routes to be used by the matatus and viewing of each route. As seen in Table 5.4

Table 5.4 Routes Module Testing Results

Test Case	Description	Test Data	Expected Outcome	Actual Results	Test Verdict
7	Creation of map routes on administrator's grid indicating start and destination for the trips	Clicking on 'Create Routes' and entering the source and destination in two separate text-fields	A leaflet map appears with polypoints starting from the source to destination as keyed in.	Map appears with polypoints configured	As expected
8	Viewing of each route indicated by polypoints on the passenger grid	Clicking on 'View Routes' button	A leaflet map indicating all the routes appears.	Map appears with all the routes configured	As expected

5.4.5 Customer Support Module

The table below summarizes the results accrued from the test cases performed on the system application. It is associated with the raising of tickets based on the application's feedback and how they are retrieved and viewed by administrators. As seen in Table 5.5

Table 5.5 Customer Support Module Testing Results

Test Case	Description	Test Data	Expected Outcome	Actual Results	Test Verdict
9	Raising tickets in the customer grid to provide feedback and rise	Clicking the 'Customer Support' button and raising a	Message indicating 'Your ticket has been raised	The message appears after clicking the	As expected

	issues regarding the application	ticket through the text-field that appears	successfully' appears.	submit appears.	
10	Viewing of tickets raised in the administrator view through a list view of each ticket.	Clicking the 'View Tickets' button and viewing the list containing each ticket raised.	List with tickets raised, ordered by the time they were raised	A list was observed with each ticket raised, ordered in the right way.	As expected

5.4.6 Fare Calculation Module

The table below summarizes the results accrued from the test cases performed on the system application. It is associated with the calculation of fare based on distance determined by differences in latitude and longitude. As seen in Table 5.6

Table 5.6 Fare Calculation Module Testing Results

Test Case	Description	Test Data	Expected Outcome	Actual Results	Test Verdict
11	Reviewing if each route from the database is listed as a drop-down in the 'start location' dropdown menu and the 'end location' drop-down and can be pointed towards.	Clicking the drop-down and seeing the displayed results	A dropdown of each route created should appear	A dropdown of each route created appeared without any obstruction	As expected
10	Reviewing if the fare amount based on the	Clicking the 'Calculate Fare' button and seeing	A value should be displayed on the card widget	A value displays on	As expected

	selection in the drop-down menus can be displayed	the displayed results	below the button indicating the total amount of fare to be paid by the passenger.	the Card Widget	

Chapter 6: Conclusions, Recommendations, and Future Works

6.1 Conclusion

The project aimed to develop an automated fare payment system to address the inefficiencies faced in the Matatu industry in Kenya due to the existing cash-based fare collection. The research identified challenges such as underpayment, overpayment, lack of accountability and inconsistency in the fare charges.

These challenges often lead to passenger dissatisfaction and revenue losses to the SACCOs. To address all these, a mobile application was developed and implemented to provide an automated fare payment solution. The application replaces the need for cash transactions hence improving efficiency in the Matatu industry.

The system has features that allows users to access real-time data on routes and changes in the fare rates. This transparency ensures that passengers stay informed about any changes which allows them to plan themselves in advance.

The entire system was rigorously tested to ensure proper functionality, usability and reliability. Testing involved different scenarios to validate the application's ability to handle transactions, provides routes and real-time data access. These tests confirmed that the system met the objectives that had been set.

The project successfully achieved its specific objectives.

6.2 Recommendations

The automated fare payment application was developed to solve the challenges faced in the Matatu industry in Kenya due to the existing cash-based fare collection. It is recommended that Matatu SACCOs adopt this system to streamline their fare collection process, and improve transparency which will increase passenger satisfaction and reduce revenue losses.

It is also recommended that to ensure smooth implementation, SACCOs should take time and train the conductors and passengers on how to use the system.

It is recommended that other public transport systems such as trains, ferries and boda-boda adapt to this payment system to ensure a unified fare payment system across the different modes of transport.

Finally, it is highly recommended that all SACCOs comply with the data protection regulations to ensure that all users' personal and financial data are protected and secure.

6.3 Future works

Several areas could be explored further to improve the performance of the system. Currently, the system uses M-Pesa as the payment gateway. The system could integrate fare payment with other payment gateways such as Paypal, Masterpass and Pesapal. These gateways support various payment methods such as Visa, MasterCard pre-paid, Debit & Credit card, and American Express which will provide more payment options for passengers.

Additionally, implementing advanced data analytics that could provide valuable insights into passenger behaviour and peak travel hours. This data will help SACCOs in proper planning and decision-making.

Also, an offline mode of the application could be implemented to ensure that the system always remains functional and in places that may have poor network connectivity to provide seamless service delivery to all users of the system.

Finally, this project has laid a foundation for automated fare payment systems and there is ample space for future works to further improve the system.

References

Activities involved in Software Requirement Analysis—GeeksforGeeks. (n.d.). Retrieved July 14, 2024, from https://www.geeksforgeeks.org/activities-involved-in-software-requirement-analysis/

Black Box Testing—Software Engineering—GeeksforGeeks. (n.d.). Retrieved July 10, 2024, from https://www.geeksforgeeks.org/software-engineering-black-box-testing/

Bondar, J. (2021, January 27). *Rapid Application Development: Advantages and Disadvantages – NIX United*. NIX United – Custom Software Development Company in US. https://nix-united.com/blog/the-ultimate-guide-to-rapid-application-development/

Brian, B. (2024, May 9). Payment Systems on Public Transport—Digital Fare Collection. *Oregon Passenger Rail*. https://www.oregonpassengerrail.org/payment-system-on-public-transport/

Cashless Transportation: Kenya And Rwanda As Case Studies. (2019, January 1). Mondato Insight. https://blog.mondato.com/cashless-transportation-kenya-rwanda/

Doynova, E. (2022, May 22). The Benefits Of Automated Fare Collection. *Modeshift*. https://www.modeshift.com/benefits-of-automated-fare-collection/

Eastlands matatu saccos enforce cashless fare system. (2020, June 29). Nation. https://nation.africa/kenya/counties/nairobi/eastlands-matatu-saccos-enforce-cashless-fare-system-1064988

ER Diagram (ERD)—Definition & Overview | Lucidchart. (n.d.). Retrieved July 5, 2024, from https://www.lucidchart.com/pages/er-diagrams

Functional requirements examples and templates. (n.d.). Jama Software. Retrieved July 14, 2024, from https://www.jamasoftware.com/requirements-management-guide/writing-requirements/functional-requirements-examples-and-templates/

High Level Design in System Design—Javatpoint. (n.d.). Retrieved July 14, 2024, from https://www.javatpoint.com/high-level-design-in-system-design

Macharia, C. G. (2017). *Regulation in the Transport Industry: A Case of Matatu Sector in Kenya* [Thesis, United States International University - Africa]. http://erepo.usiu.ac.ke:8080/xmlui/handle/11732/3539

Mageria, N. G. (n.d.). *UNIVERSITY OF NAIROBI COLLEGE OF BIOLOGICAL AND PHYSICAL SCIENCES SCHOOL OF COMPUTING AND INFORMATICS*.

Mastering the Art of Cutover Planning: A Step-by-Step Guide -. (n.d.). Retrieved July 14, 2024, from https://www.enov8.com/blog/mastering-the-art-of-cutover-planning-a-step-by-step-guide/#

Matatu Driver Cornered After Faking Mobile Money Texts to Petrol Station for Months—Kenyans.co.ke. (n.d.). Retrieved July 14, 2024, from https://www.kenyans.co.ke/news/88882-matatu-driver-cornered-after-faking-mobile-money-texts-petrol-station-months

Matatu fare control: An analysis and feasibility of the proposed legislation | Vellum Kenya. (2023, August 14). https://vellum.co.ke/matatu-fare-control-an-analysis-and-feasibility-of-the-proposed-legislation/

Nonfunctional Requirements: Examples, Types and Approaches. (2024, February 29). AltexSoft. https://www.altexsoft.com/blog/non-functional-requirements/#

phanivedala. (2023, August 10). *Black Box Testing: Exploring Types and Benefits*. Learning Center. https://www.extnoc.com/learn/security/black-box-testing

Pros and Cons of Flutter App Development. (2023, December 8). AltexSoft. https://www.altexsoft.com/blog/pros-and-cons-of-flutter-app-development/

Refactoring Home Page. (n.d.). Retrieved July 14, 2024, from http://refactoring.com

System Design and System Architecture | by Brandon Kang | Bootcamp. (n.d.). Retrieved July 17, 2024, from https://bootcamp.uxdesign.cc/system-design-and-system-architecture-e963d030bc7b

System Testing—Software Engineering. (2019, May 30). GeeksforGeeks. https://www.geeksforgeeks.org/system-testing/

Transforming public transport in Africa: Are automated fare systems the answer? (n.d.). Retrieved July 14, 2024, from https://blogs.worldbank.org/en/transport/transforming-public-transport-africa-are-automated-fare-systems-answer

Transport sector stares at Sh50bn revenue loss in short-term. (n.d.). Retrieved July 14, 2024, from https://www.the-star.co.ke/business/kenya/2020-04-08-transport-sector-stares-at-sh50bn-revenue-loss-in-short-term/

UML Use Case Diagram Tutorial. (n.d.). Lucidchart. Retrieved May 3, 2024, from https://www.lucidchart.com/pages/uml-use-case-diagram

What Is a Database Schema? | IBM. (2024, April 4). https://www.ibm.com/topics/database-schema

What is a Wireframe & Its Role in the Design Process | Miro. (n.d.). Https://Miro.Com/. Retrieved July 17, 2024, from https://miro.com/wireframe/what-is-a-wireframe/

What is Class Diagram? (n.d.). Retrieved July 14, 2024, from https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/

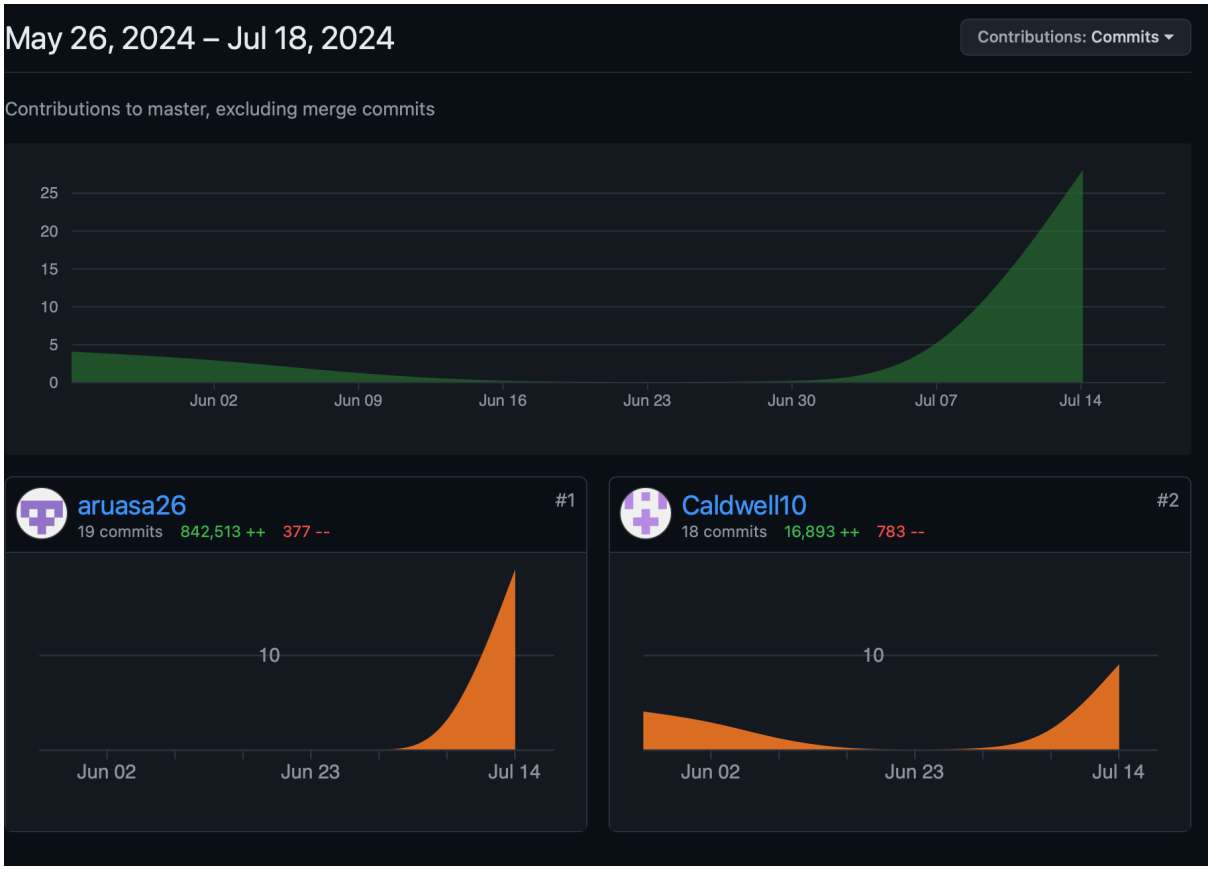
What is "pretty privilege" and why is it such a problem? | My Imperfect Life. (n.d.). Retrieved July 14, 2024, from https://www.myimperfectlife.com/features/pretty-privilege

What is System Software? – Definition, Types, Examples and More. (n.d.). Retrieved July 14, 2024, from https://www.geeksforgeeks.org/system-software/

What is Test-Driven Development? | TestDriven.io. (n.d.). Retrieved July 14, 2024, from https://testdriven.io/test-driven-development/Ap

Appendices

Appendix A1: GitHub Report



URL link: https://github.com/Caldwell10/Fare_payment_system

Appendix A2: Project Time Schedule