FareFlow: An IoT and Cloud-Based Smart Bus Fare Collection System for Sustainable Urban Transport

PUTSHU LUNGHE Samuel [1], Dr. Mayur Kumar Chhipa [2]

[1] Student, Department of Electronics and Communication Engineering, ISBAT University
[2] Head of Engineering, Faculty of Engineering, ISBAT University
*[1] Email Address: samuellunghe@gmail.com
*[2] Email Address: mayur.fict@isbatuniversity.com

Abstract

Public transportation in developing countries, including Uganda, continues to rely heavily on manual cash-based fare collection methods. These systems are prone to inefficiencies such as delays during boarding, revenue leakage, lack of transparency, and hygiene risks from physical cash handling. This study proposes FareFlow, a smart RFID-based bus fare payment system that leverages Internet of Things (IoT) devices and cloud computing to address these challenges while contributing to the United Nations Sustainable Development Goals (SDG 9: Industry, Innovation & Infrastructure, and SDG 11: Sustainable Cities & Communities). The system integrates an RC522 RFID reader with a NodeMCU ESP8266 microcontroller to enable contactless passenger identification and fare deduction. A GPS module provides real-time bus tracking, while cloud services based on Firebase Firestore and an Express.js backend manage user accounts, balances, and transaction logs.

FareFlow also features web-based applications for three stakeholder groups: passengers, bus operators, and administrators. Passengers can view balances, monitor transaction history, and receive SMS/email alerts. Operators gain real-time visibility of fare activities on their routes, while administrators access centralized dashboards for system oversight, card registration, and revenue analysis. Prototype implementation and functional testing confirmed the system’s reliability, with transactions processed in under 1.4 seconds and real-time updates synchronized across the cloud. Results demonstrate improvements in efficiency, accountability, and user convenience. By enhancing operational transparency, reducing cash dependency, and enabling digital infrastructure for public transport, FareFlow illustrates how IoT and cloud technologies can support sustainable, inclusive, and resilient urban mobility in Uganda and similar contexts.

Keywords: RFID, IoT, Smart Transportation, Cloud Computing, Bus Fare System, Uganda, SDGs

1. Introduction

Public transportation is vital to economic and social activities in developing countries. In Uganda, buses and commuter vehicles form the backbone of daily mobility, yet fare collection remains largely manual and cash-based. This leads to inefficiencies such as long boarding times, revenue leakage, fare disputes, theft risks, and hygiene concerns; issues highlighted further during public health crises like COVID-19. These challenges show the need for smarter and more transparent fare management systems.

Advances in the Internet of Things (IoT), cloud computing, and contactless payments present opportunities to modernize fare collection. While many developed countries have adopted RFID-based systems to reduce cash dependency and improve efficiency, adoption in Africa has been limited due to costs, infrastructure, and lack of localized designs.

This paper introduces FareFlow, a smart RFID-based bus fare payment system built with cost-effective components and open-source tools. It uses RFID cards for passenger identification, a NodeMCU ESP8266 microcontroller for IoT communication, and Firebase with Express.js for real-time backend processing. The system targets three stakeholders: passengers (contactless payments, notifications, balance history), operators (real-time monitoring and revenue insights), and administrators (centralized control and reporting).

The study's objectives are:

1. To design and implement a contactless fare collection system using RFID and IoT hardware.
2. To integrate a cloud-based backend for real-time transaction processing and data logging.
3. To develop web-based applications tailored to passengers, operators, and administrators.

The hypothesis is that an IoT- and cloud-based RFID fare system can significantly enhance efficiency, transparency, and hygiene compared to cash. By leveraging accessible technology, FareFlow offers an affordable solution for Uganda's transport sector, improving passenger convenience, financial accountability, and public health.

The rest of this paper is organized as follows: Section 2 reviews related literature, Section 3 presents system design, Section 4 describes implementation and testing, and Section 5 concludes with recommendations.

2. Literature Review

The advancement of smart transportation systems has been closely linked with developments in Radio Frequency Identification (RFID), Internet of Things (IoT), and cloud computing technologies. Oberli et al. (2010) evaluated UHF RFID technologies for real-time passenger recognition, demonstrating their potential to improve operational efficiency in public transit systems through fast and reliable passenger identification. Similarly, Li and Wei (2008) proposed a real-time vehicle routing system for RFID-tagged goods, highlighting the scalability of RFID applications beyond simple identification into logistics and mobility solutions.

In the context of IoT-enabled services, Al-Kababji et al. (2019) showcased how Firebase Real-time Database can be integrated with wireless monitoring systems, confirming the reliability of cloud-based infrastructures in handling real-time data for healthcare, which parallels the transportation sector's need for speed and accuracy. Other researchers have emphasized the importance of simulation and optimization tools in system design, such as Chiriță et al. (2017), who applied SolidWorks modeling for flywheel design. Their work illustrates how computer-aided prototyping can accelerate the development of complex engineering solutions like FareFlow.

Commercial and open-source platforms also play a critical role in deploying scalable solutions. For instance, Firebase (n.d.), Express.js (n.d.), React (n.d.), and NodeMCU (n.d.) provide robust frameworks for backend processing, frontend design, and IoT device integration. The availability of low-cost RFID modules such as RC522 (n.d.) has further reduced implementation barriers for resource-constrained regions. Together, these studies and tools underline the global trend toward digital, contactless, and transparent fare collection systems, setting a foundation for the proposed FareFlow system.

3. Methodology and System Design

The design and development of the FareFlow system followed a structured methodology that combined hardware prototyping, backend development, and user interface design. The system was developed with the objective of creating a low-cost, scalable, and transparent fare collection platform that leverages IoT and cloud computing technologies.

3.1 System Architecture

FareFlow was developed using a three-tier architecture (input layer, processing layer, and output layer). Input Layer: Comprises the hardware installed on buses, including the RC522 RFID reader, NodeMCU ESP8266 microcontroller, NEO-6M GPS module, LCD display, buzzer, and LED indicators. This layer is responsible for capturing passenger card data, providing feedback, and transmitting data to the server. Processing Layer: Implements the backend logic through an Express.js server and Firebase Firestore database. This layer manages user authentication, fare deduction, transaction logging, and system analytics. It also integrates SMS and email notification services for real-time communication.
Output Layer: Provides feedback and monitoring to stakeholders via web interfaces, SMS, and email. Passengers interact through a web portal, bus operators use a real-time dashboard, and administrators oversee the system through a centralized control panel.

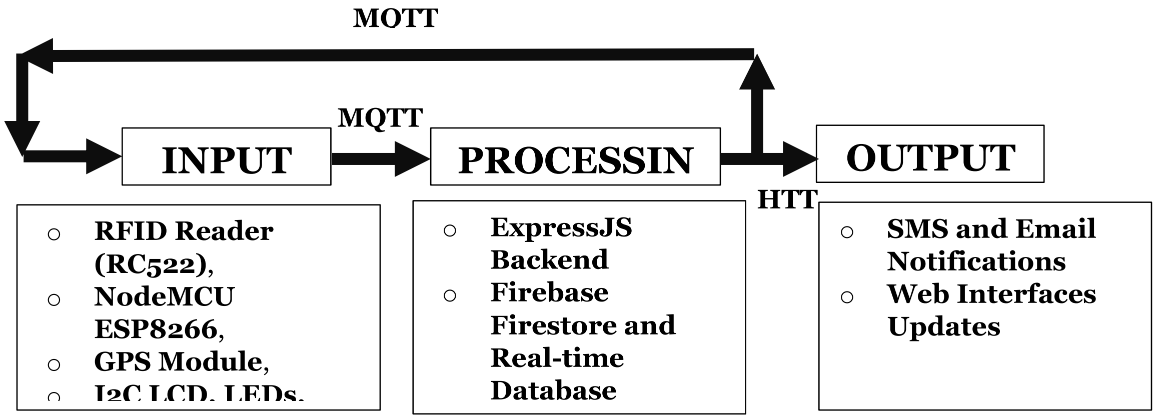


Figure1: System Architecture

3.2 Hardware Components

The hardware prototype was implemented with cost-effective and readily available components:

- NodeMCU ESP8266: Serves as the microcontroller and Wi-Fi-enabled IoT hub. It communicates with the RFID reader and backend server.
- RC522 RFID Reader: Reads the unique ID (UID) of each passenger's RFID card.
- RFID Cards/Tags: Store user account information and balance in the backend system.
- 16x2 I2C LCD Display: Provides real-time transaction messages (e.g., "Payment Success," "Low Balance").
- LEDs and Buzzer: Deliver instant audio-visual confirmation of successful or failed transactions.
- NEO-6M GPS Module: Tracks bus location, enabling route analysis and future integration with dynamic pricing or live tracking.
- Lithium-Ion Battery Pack: Provides portable power for the system, ensuring uninterrupted operation.

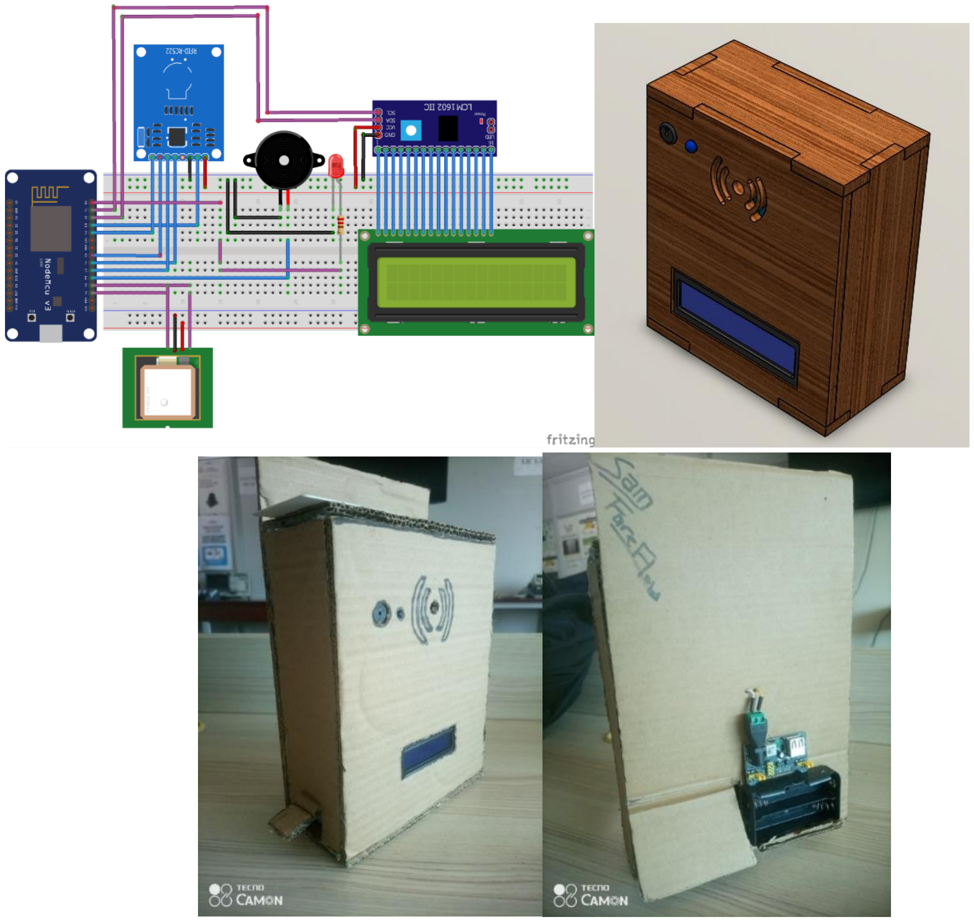


Figure 2: Hardware Setup

3.3 Software Components

The software stack integrates microcontroller firmware, backend services, and frontend applications:

- Firmware Development: Implemented in Arduino IDE (C/C++). The firmware enables the NodeMCU to read RFID UIDs, communicate with the backend, and control LEDs, buzzer, and LCD outputs.
- Backend Server: Built with Express.js and deployed on Railway. The backend verifies user identities, deducts fares, and logs transactions. Firebase Authentication provides secure login and role-based access.

- Database: Firebase Firestore stores user profiles, balances, and transaction histories, while Firebase Real-time Database manages live bus activity data.
- Frontend Interfaces: Developed using React.js and deployed on Vercel. Each stakeholder (passenger, operator, administrator) has access to a dedicated web portal.
- Notification Services: Email JS for automated email alerts and Africa's Talking API for SMS notifications.

3.4 Workflow of the System

The operational workflow of the FareFlow system is designed to ensure real-time fare processing, accurate balance management, and seamless passenger interaction. The process proceeds as follows:

- Card Tapping: A passenger taps their RFID card on the bus-mounted RC522 reader.
- UID Retrieval and Transmission: The NodeMCU ESP8266 retrieves the card's unique identifier (UID) and transmits it securely to the backend Node.js server via Wi-Fi using the MQTT protocol.
- Verification and Minimum Balance Check: The server queries Firebase Firestore to verify the UID and retrieves the passenger's account details. It checks whether the account balance is above the minimum threshold of 1000 UGX. If the balance is insufficient, the card is rejected and an error notification is issued.
- Route Type Determination: The backend determines the applicable route type:
  - Fixed Fare Route: The fare is a flat, predetermined rate for the entire trip. If the passenger has sufficient balance, the fare is deducted immediately, and the passenger is notified via SMS and email.
  - Dynamic Fare Route: The fare depends on the distance travelled by the passenger.
- Dynamic Fare Processing: For dynamic routes, the backend checks whether the passenger's card is already registered on the active trip:
  - Entry Tap: If the passenger is boarding, the system records the GPS coordinates of the entry point and registers the passenger as active on the bus.
  - Exit Tap: When the passenger exits, they tap their card again. The system records the exit coordinates, calculates the travel distance using the Euclidean distance formula, and computes the fare at a rate of 500 UGX per kilometer.

The Euclidean distance between two geographic coordinates $(x1, y1)$ and $(x2, y2)$ is given by:

$$D = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

where:

- $x1, y1$ = entry latitude and longitude,
- $x2, y2$ = exit latitude and longitude,
- D = distance traveled (in kilometers).

The fare is then calculated as:

$$Fare = D \times 500\, UGX/km$$

Notifications detailing the trip cost and remaining balance are sent via SMS and email to the passenger.

Dynamic fare calculation using GPS coordinates adds fairness and adaptability to the payment system.

Passenger Feedback: Based on the backend response, the NodeMCU updates the I2C LCD display with a confirmation or error message, while LEDs and buzzer signals provide instant feedback.
Transaction Logging and Notifications: All completed transactions, including UID, timestamp, route type, amount deducted, and bus ID, are recorded in Firebase Firestore. SMS and email notifications are dispatched to ensure transparency and passenger awareness.This workflow ensures real-time validation, transparent fare deduction, and automated logging, while supporting both fixed and distance-based dynamic pricing models.

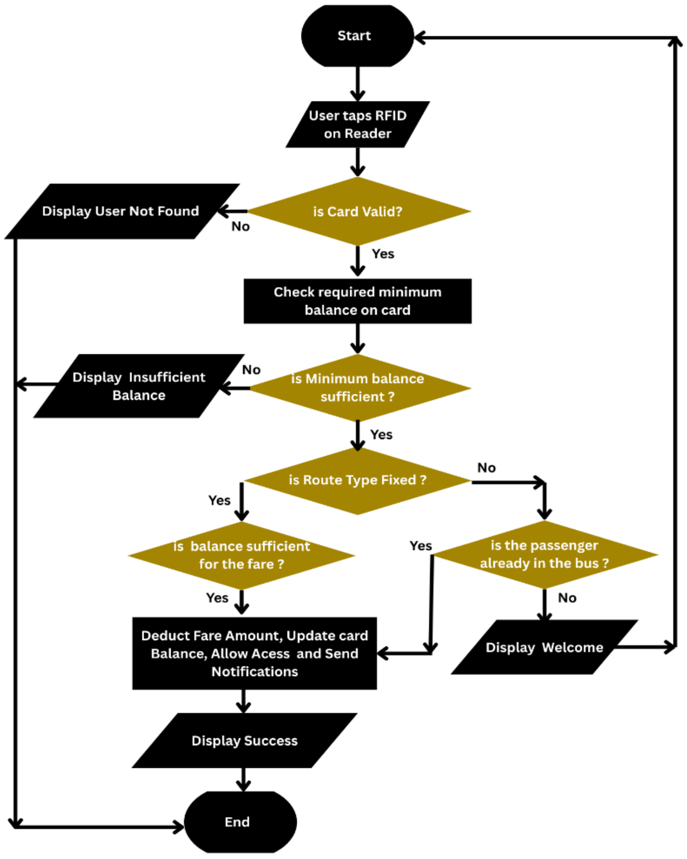

Start
User taps RFID on Reader
is Card Valid?
No
Display User Not Found
Yes
Check required minimum balance on card
is Minimum balance sufficient ?
No
Display Insufficient Balance
Yes
is Route Type Fixed ?
Yes
No
is balance sufficient for the fare ?
Yes
is the passenger already in the bus ?
Yes
No
Deduct Fare Amount, Update card Balance, Allow Acess and Send Notifications
Display Welcome
Display Success
End

Figure 3: System Flowchart

4. Results and Evaluation

The FareFlow prototype was implemented and tested to validate its effectiveness in addressing the inefficiencies of cash-based fare collection in Uganda's public transportation. The evaluation focused on functional accuracy, system performance, user experience, and overall feasibility for deployment.

4.1 Testing Objectives

The testing phase aimed to:

- Verify RFID card scanning and UID validation.
- Ensure real-time balance deduction and transaction logging.
- Assess backend reliability and communication between hardware and cloud.
- Evaluate notification systems (SMS and email).

- Validate user interface functionality for passengers, operators, and administrators.

4.2 Testing Environment

Hardware Setup: NodeMCU ESP8266, RC522 RFID reader, 16x2 LCD, buzzer, LEDs, GPS module, and rechargeable battery pack.
Software Setup: Backend (Express.js) hosted on Railway, frontend (React.js) deployed on Vercel, Firebase Firestore as cloud database.
Network: Wi-Fi enabled environment.
Devices: RFID cards and desktop browser interfaces for dashboards.

4.3 Functional Testing

Functional tests were carried out on different card scenarios:

Table 1: Functional Testing Results

Test Case	Expected Result	Outcome
Valid UID with sufficient balance	Fare deducted, LCD success message, SMS/email sent	Passed
Valid UID with insufficient balance	Transaction rejected, LCD error message, alert sent	Passed
Fixed fare route	Flat rate deducted, confirmation sent	Passed
Dynamic route (entry + exit taps)	Distance-based fare calculated and deducted	Passed
Inactive bus	No transactions processed, error displayed	Passed

All tests confirmed that the system responded appropriately under different operational conditions.

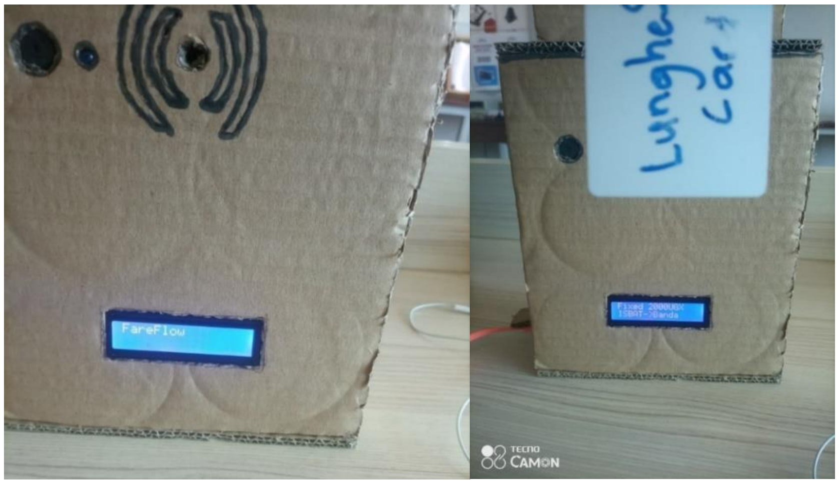


Figure 2: Device Starting and Route Setting

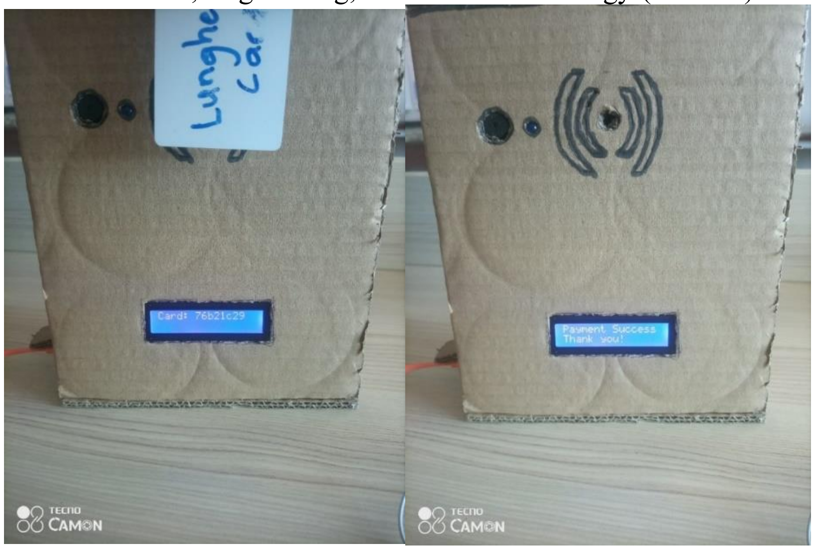


Figure 3: Card Reading and Successful Payment

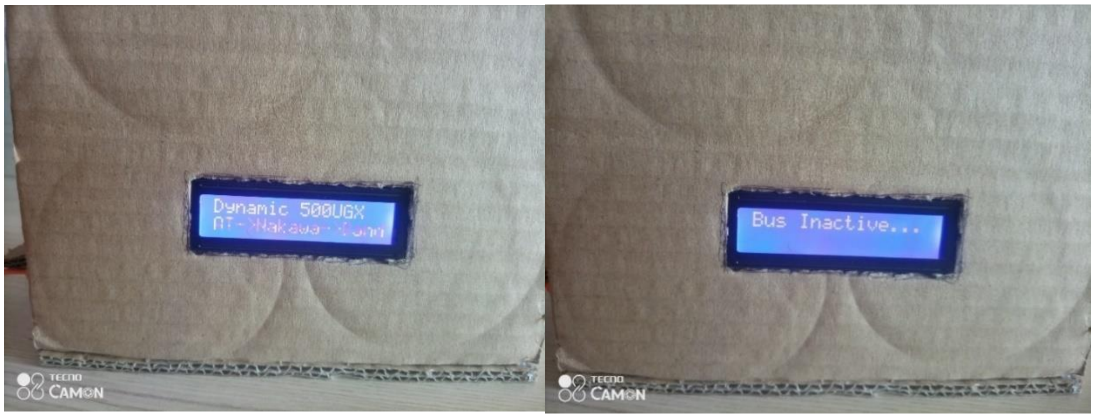
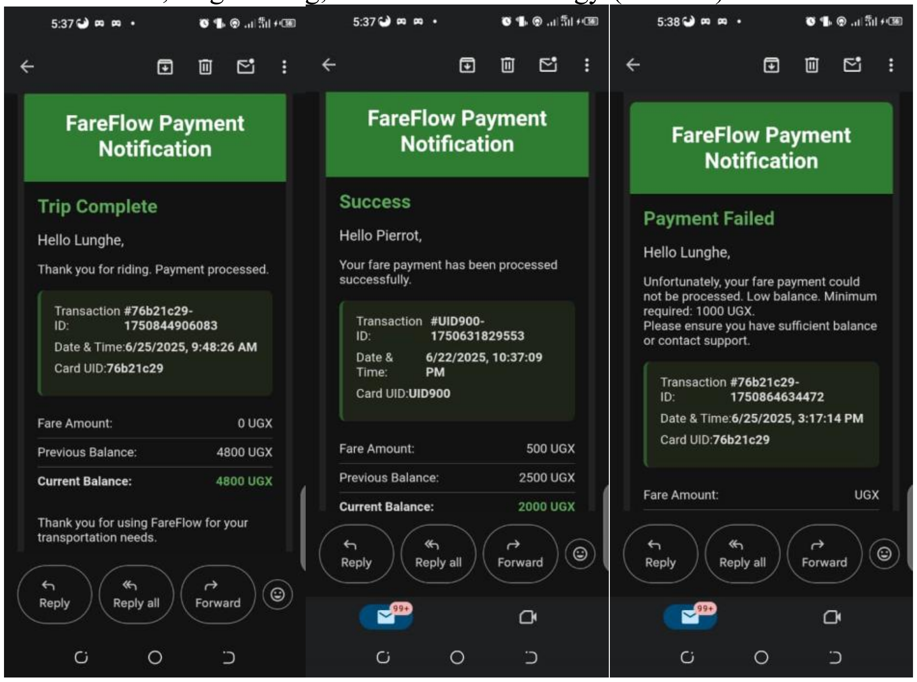


Figure 4: Dynamic pricing snapshot and Bus Deactivation

4.4 Integration Testing

Integration testing validated communication between system modules:

- Hardware-Backend Communication: The NodeMCU transmitted card data reliably via MQTT, with server responses received in <1.3 seconds.
- Backend-Database Interaction: Firebase Firestore updated balances and logged transactions in real time.
- Backend-Notification Services: SMS messages (Africa’s Talking) and email alerts (EmailJS) were delivered consistently, though occasional delays were observed due to API rate limits.


5:37

FareFlow Payment Notification

Trip Complete

Hello Lunghe,

Thank you for riding. Payment processed.

Transaction ID: #76b21c29-1750844906083
Date & Time: 6/25/2025, 9:48:26 AM
Card UID: 76b21c29

Fare Amount:	0 UGX
Previous Balance:	4800 UGX
Current Balance:	**4800 UGX**

Thank you for using FareFlow for your transportation needs.

5:37

FareFlow Payment Notification

Success

Hello Pierrot,

Your fare payment has been processed successfully.

Transaction ID: #UID900-1750631829553
Date & Time: 6/22/2025, 10:37:09 PM
Card UID: UID900

Fare Amount:	500 UGX
Previous Balance:	2500 UGX
Current Balance:	**2000 UGX**

5:38

FareFlow Payment Notification

Payment Failed

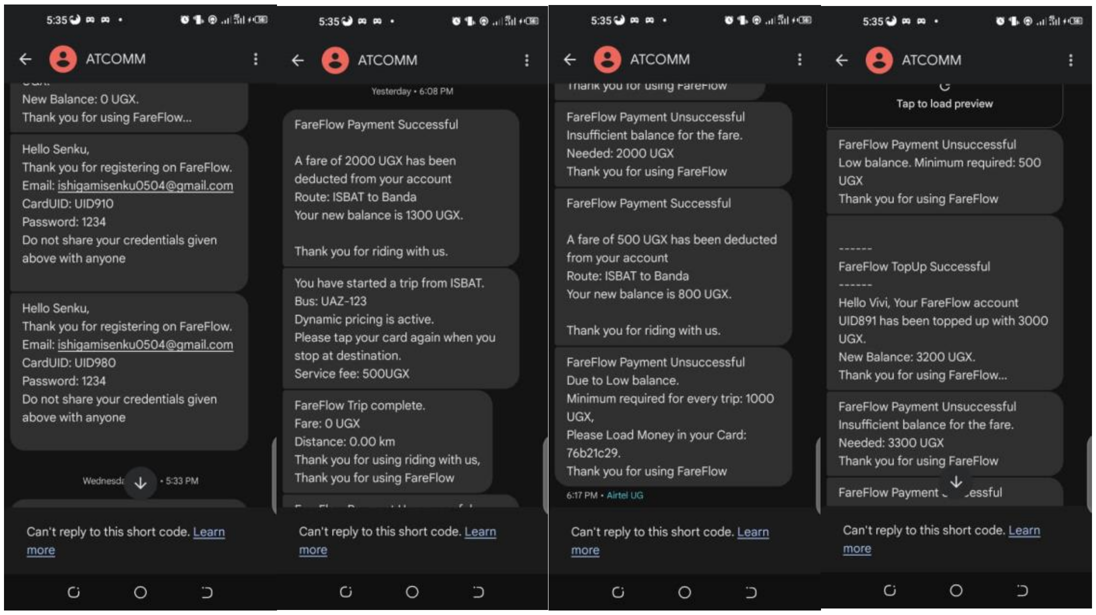
Hello Lunghe,

Unfortunately, your fare payment could not be processed. Low balance. Minimum required: 1000 UGX.
Please ensure you have sufficient balance or contact support.

Transaction ID: #76b21c29-1750864634472
Date & Time: 6/25/2025, 3:17:14 PM
Card UID: 76b21c29

Fare Amount:	UGX

Figure 5: Email Notifications


5:35 ATCOMM

New Balance: 0 UGX.
Thank you for using FareFlow...

Hello Senku,
Thank you for registering on FareFlow.
Email: ishigamisenku0504@gmail.com
CardUID: UID910
Password: 1234
Do not share your credentials given above with anyone

Hello Senku,
Thank you for registering on FareFlow.
Email: ishigamisenku0504@gmail.com
CardUID: UID980
Password: 1234
Do not share your credentials given above with anyone

Wednesday • 5:33 PM

Can't reply to this short code. Learn more

5:35 ATCOMM

Yesterday • 6:08 PM

FareFlow Payment Successful

A fare of 2000 UGX has been deducted from your account
Route: ISBAT to Banda
Your new balance is 1300 UGX.

Thank you for riding with us.

You have started a trip from ISBAT.
Bus: UAZ-123
Dynamic pricing is active.
Please tap your card again when you stop at destination.
Service fee: 500UGX

FareFlow Trip complete.
Fare: 0 UGX
Distance: 0.00 km
Thank you for using riding with us,
Thank you for using FareFlow

Can't reply to this short code. Learn more

5:35 ATCOMM

Thank you for using FareFlow

FareFlow Payment Unsuccessful
Insufficient balance for the fare.
Needed: 2000 UGX
Thank you for using FareFlow

FareFlow Payment Successful

A fare of 500 UGX has been deducted from your account
Route: ISBAT to Banda
Your new balance is 800 UGX.

Thank you for riding with us.

FareFlow Payment Unsuccessful
Due to Low balance.
Minimum required for every trip: 1000 UGX,
Please Load Money in your Card: 76b21c29.
Thank you for using FareFlow

6:17 PM • Airtel UG

Can't reply to this short code. Learn more

5:35 ATCOMM

Tap to load preview

FareFlow Payment Unsuccessful
Low balance. Minimum required: 500 UGX
Thank you for using FareFlow

FareFlow TopUp Successful

Hello Vivi, Your FareFlow account UID891 has been topped up with 3000 UGX.
New Balance: 3200 UGX.
Thank you for using FareFlow...

FareFlow Payment Unsuccessful
Insufficient balance for the fare.
Needed: 3300 UGX
Thank you for using FareFlow

FareFlow Payment ...essful

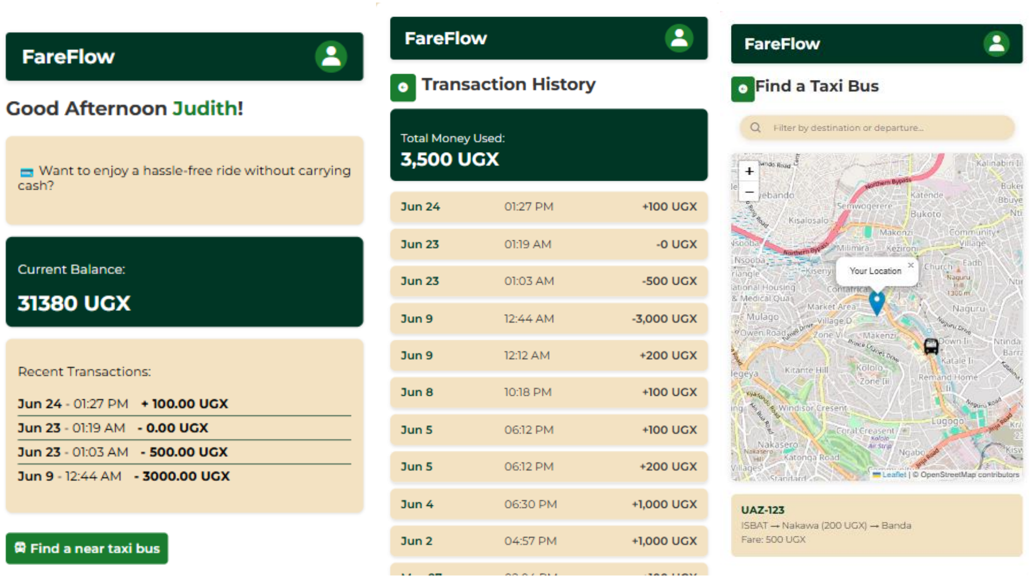
Can't reply to this short code. Learn more

Figure 6: SMS Notifications

4.5 User Interface Evaluation

User interfaces were assessed for usability and correctness:

- Passenger Portal: Allowed users to view balances and transaction history. Recharge request functionality worked correctly.


FareFlow

Good Afternoon Judith!

Want to enjoy a hassle-free ride without carrying cash?

Current Balance:
31380 UGX

Recent Transactions:
Jun 24 - 01:27 PM + 100.00 UGX
Jun 23 - 01:19 AM - 0.00 UGX
Jun 23 - 01:03 AM - 500.00 UGX
Jun 9 - 12:44 AM - 3000.00 UGX

Find a near taxi bus

FareFlow

Transaction History

Total Money Used:
3,500 UGX

Jun 24	01:27 PM	+100 UGX
Jun 23	01:19 AM	-0 UGX
Jun 23	01:03 AM	-500 UGX
Jun 9	12:44 AM	-3,000 UGX
Jun 9	12:12 AM	+200 UGX
Jun 8	10:18 PM	+100 UGX
Jun 5	06:12 PM	+100 UGX
Jun 5	06:12 PM	+200 UGX
Jun 4	06:30 PM	+1,000 UGX
Jun 2	04:57 PM	+1,000 UGX

FareFlow

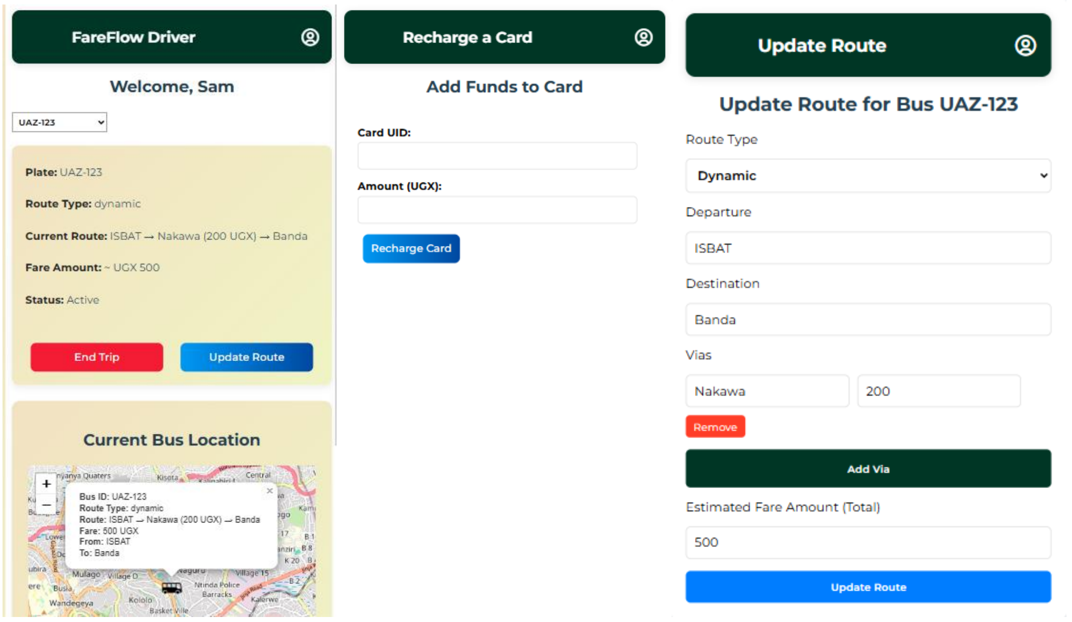
Find a Taxi Bus

Filter by destination or departure...

UAZ-123
ISBAT → Nakawa (200 UGX) → Banda
Fare: 500 UGX

Figure 7: Passenger App UI

- Operator/Driver Dashboard: Displayed live passenger activity and fare deductions per trip.


FareFlow Driver

Welcome, Sam

UAZ-123

Plate: UAZ-123
Route Type: dynamic
Current Route: ISBAT → Nakawa (200 UGX) → Banda
Fare Amount: ~ UGX 500
Status: Active

End Trip | Update Route

Current Bus Location

Bus ID: UAZ-123
Route Type: dynamic
Route: ISBAT → Nakawa (200 UGX) → Banda
Fare: 500 UGX
From: ISBAT
To: Banda

Recharge a Card

Add Funds to Card

Card UID:

Amount (UGX):

Recharge Card

Update Route

Update Route for Bus UAZ-123

Route Type: Dynamic
Departure: ISBAT
Destination: Banda
Vias: Nakawa 200

Remove

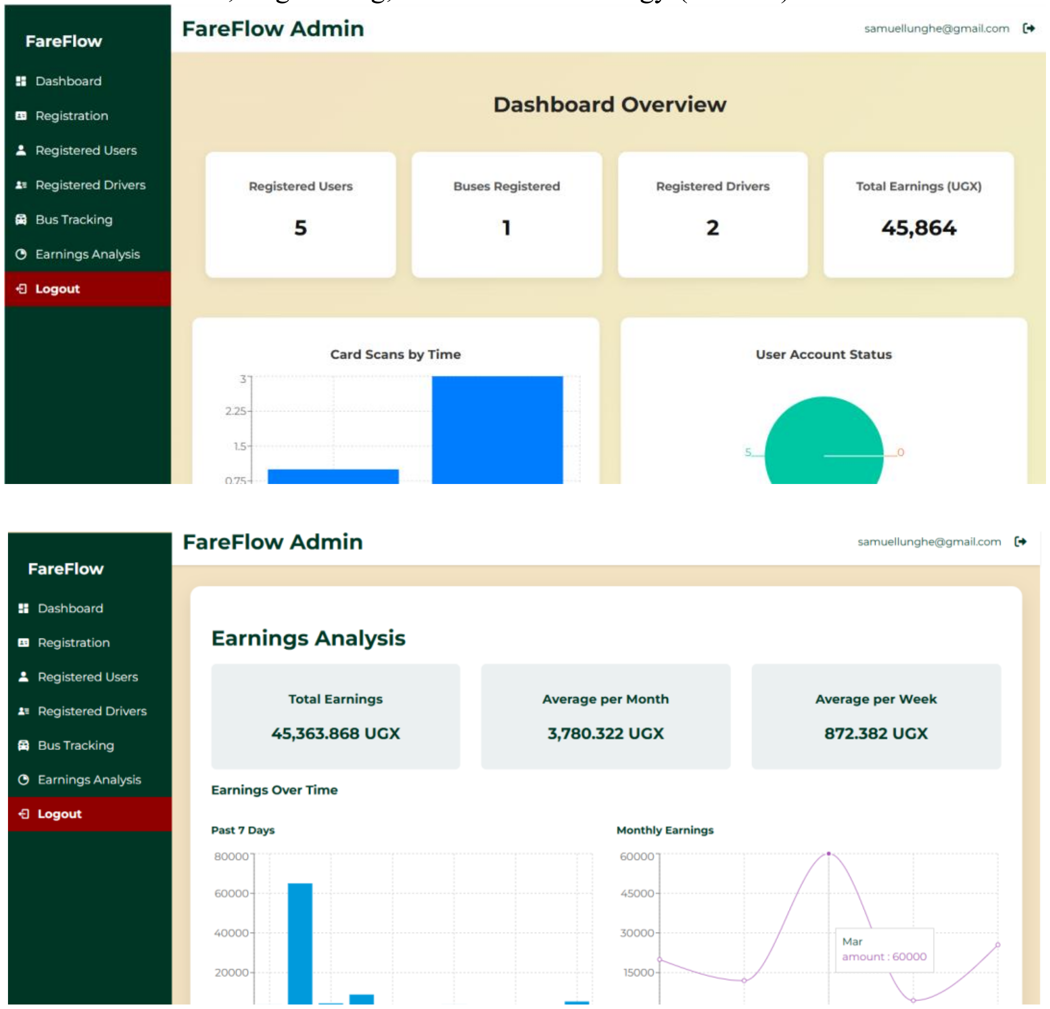
Add Via

Estimated Fare Amount (Total): 500

Update Route

Figure 8: Operator/Driver Web UI

- Admin Control Panel: Enabled administrators to register/deactivate cards, manage users, and monitor analytics. Role-based access control was enforced.



Dashboard Overview

Registered Users 5

Buses Registered 1

Registered Drivers 2

Total Earnings (UGX) 45,864

Earnings Analysis

Total Earnings 45,363.868 UGX

Average per Month 3,780.322 UGX

Average per Week 872.382 UGX

Figure 9: Admin Dashboard UI

4.6 Limitations Identified

Despite the successful results, several limitations were observed:

- The system's dependency on stable internet connectivity may limit usability in rural or low-infrastructure areas.
- Hardware prototype needs improved ruggedness and environmental resilience for real-world deployment
- The dynamic fare computation currently uses Euclidean (straight-line) distance between entry and exit points, which may underestimate the actual travel distance along curved or non-linear routes. Incorporating road-network-based or GPS-trace distance calculations (e.g., Haversine or map-matching algorithms) would improve accuracy and fairness.
- Recharge process not fully integrated with mobile money APIs, limiting user convenience.
- Occasional RFID scan alignment errors, requiring passengers to retap cards.

5. Conclusion

This study presented FareFlow, a smart bus fare collection system that integrates RFID technology, IoT hardware, and cloud-based services to address inefficiencies in Uganda's cash-dependent public transportation sector. Prototype implementation demonstrated that FareFlow can process transactions in under 1.4 seconds, synchronize balances and logs in real time, and provide transparency through web, SMS, and email notifications. The system improves efficiency, accountability, and hygiene by minimizing human error, reducing boarding delays, and lowering risks associated with cash handling.

The evaluation further showed that FareFlow is both feasible and scalable using cost-effective components such as NodeMCU ESP8266, RC522 RFID reader, and Firebase Firestore. Stakeholders; passengers, operators, and administrators; benefit from tailored interfaces that enhance user experience, financial oversight, and system governance. Nonetheless, challenges such as dependence on stable internet connectivity, limited mobile money integration, and non-ruggedized hardware must be addressed before large-scale deployment.

Overall, FareFlow demonstrates how affordable IoT and cloud-based technologies can modernize public transportation in developing countries while directly contributing to SDG 9 by fostering innovation and digital infrastructure, and to SDG 11 by promoting sustainable, safe, and inclusive urban mobility. Future work should focus on integrating mobile payment systems, improving hardware durability, and piloting the solution at scale. By doing so, FareFlow can serve as a model for smart, transparent, and sustainable fare collection across Africa and beyond.

References

Oberli, C., Torres-Torriti, M., & Landau, D. (2010). Performance evaluation of UHF RFID technologies for real-time passenger recognition in intelligent public transportation systems. *IEEE Transactions on Intelligent Transportation Systems, 11*(3), 748–753. https://doi.org/10.1109/TITS.2010.2048429

Li, F., & Wei, Y. (2008). A real-time vehicle routing system for RFID-tagged goods transportation. In *2008 IEEE International Conference on Service Operations and Logistics, and Informatics* (pp. 2892–2897). IEEE. https://doi.org/10.1109/SOLI.2008.4683029

Al-Kababji, A., et al. (2019). IoT-based fall and ECG monitoring system: Wireless communication system based Firebase Realtime Database. In *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)* (pp. 1480–1485). IEEE. https://doi.org/10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00267

Chiriță, I., Tănase, N., Apostol, S. E., Ilie, C., & Popa, M. (2017). Design optimization of a flywheel using SolidWorks modeling and simulation capabilities. In *2017 International Conference on ENERGY and ENVIRONMENT (CIEM)* (pp. 344–348). IEEE. https://doi.org/10.1109/CIEM.2017.8120858

Firebase. (n.d.). *Firebase documentation*. https://firebase.google.com/docs
Express.js. (n.d.). *Express.js guide*. https://expressjs.com
React. (n.d.). *React documentation*. https://reactjs.org/docs/getting-started.html
NodeMCU Documentation. (n.d.). *ESP8266 developer guide*. https://nodemcu.readthedocs.io
RC522 RFID Module. (n.d.). *Datasheet and setup*. https://components101.com
EmailJS. (n.d.). *Email sending via JavaScript*. https://www.emailjs.com/docs
Africa's Talking. (n.d.). *SMS API documentation*. https://developers.africastalking.com
MQTT.org. (n.d.). *MQTT protocol technical overview*. https://mqtt.org/documentation
Vercel. (n.d.). *Frontend deployment platform*. https://vercel.com
Railway. (n.d.). *Express.js deployment platform*. https://railway.app