TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PASCHIMANCHAL CAMPUS

A Major Project Report On

SAHAJ YATRA

”A Digital Bus Fare Management System”

Submitted by:

Aayush Shrestha (076BEI002)

Dikshyanta Aryal (076BEI010)

Pramish Gurung (076BEI023)

Santosh Kumar Rajbhandari (076BEI033)

Submitted to:

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

PASCHIMANCHAL CAMPUS

LAMACHAUR, POKHARA

Falgun, 2080

COPYRIGHT

ACKNOWLEDGMENT

We would like to express our sincere gratitude to our project supervisor, Mr.Suraj Basanta Tulachan for his unwavering support, valuable insights and continuous encouragement during this entire project. We would also like to express gratitude towards the Electronics and Computer Engineering Deparment, associated teachers and all the well wishers.
We solely take the responsibility of any possible mistakes that may have occurred in preparing this report and we would like to welcome comments and queries during the submission of this report.

ABSTRACT

Sahaj Yatra is a digital bus fare management system, which address the problems faced by both passengers and bus owners in fare collection. This system utilizes RFID-based scanning and the NodeMCU for hardware integration. The system aims to enhance the passengers experience and improve operational efficiency by offering seamless fare validation and convenient payment options. This project aims to provide a fair fare management system providing consumers with a easy and hassle free experience. The passengers will be issued RFID cards, which will be scanned by onboard RFID scanners installed on buses to deduct the fare amount in real-time. The Node server will handle user registration, fare deduction, recharge history, and transaction logs, all of which will be saved in a database, ensuring secure storage and efficient data management. Moreover, the passengers will be able to track the bus in real-time, helping them manage their schedule accordingly. Additionally, they will be provided with the facility to recharge their cards using local payment methods, enabling a hassle-free experience. The bus operators will be provided with the facility to get an overview of the details of buses owned by them. The proposed Digital Bus Fare Management System brings efficiency and convenience to the existing transportation system, improving the daily lives of millions of passengers and introducing a hassle-free service for both passengers and bus operators.

Keywords: Database, GPS, Nodejs, NodeMCU, RFID, Transaction logs

TABLE OF CONTENTS

List Of Figures

LIST OF ABBREVIATIONS

AFC	*Automatic Fare Calculation*
EEPROM	*Electrically Erasable Programmable Read-Only Memory*
GLONASS	*GLObalnaya NAvigatsionnaya Sputnikovaya Sistema in Russian*
GPS	*Global Positioning System*
GPIO	*General Purpose Input Output*
MISO	*Master In Slave Out*
MOSI	*Master Out Slave In*
NEMA	*National Marine Electronics Association*
RFID	*Radio Frequency Identification*
RISC	*Reduced Instruction Set Computer*
SBAS	*Satellite Based Augmentation System*
SCL	*Serial Clock*
SDA	*Serial Data Line*
TTMF	*Time-To-First-Fix*
USART	*Universal Synchronous and Asynchronous Receiver-Transmitter*
USB-TTL	*Universal Serial Bus Transistor-Transistor Logic*

CHAPTER 1: INTRODUCTION

1.1 Background

As the population is increasing rapidly, especially in city area, one of the many issues is transportation management. Anyone who had travelled in public buses can only understand the difficulty and headache of travelling in our public buses. Public buses are always crowded and if not are waiting for passengers. Reason of the crowded buses is not only due to insufficient number of vehicles but also poor management system. Infrequent service intervals can lead to overcrowding, as passengers tend to accumulate while waiting for the limited number of vehicles. Another reason is certain routes or lines may experience heavier passenger loads due to population distribution or specific destinations, causing congestion.

system aims to revolutionize the public transportation system. This system involves IC card, mobile app, server. Passenger scan IC card once they get into the bus and scan again while getting off the bus. The server calculates the fare according to the distance he/she has travelled. Whole system is autonomous, passenger just has to load money to the account using mobile application. Passenger can also track different buses available in area using the app. Owner of the bus also has dashboard that shows revenue collected. Buses can be sent to the area, where there are large number of passengers during pick hour.

IC card gates require authenticating the card before processing it and verifying that it is present in the signal field. After that, data are read, assessed, written, and confirmed again. According to field testing, the fastest users only took 0.2 seconds to place their cards in the reader signal field. The processing time between the card and reader must be shorter than 0.1 s in order to avoid the passenger having to stop when going through the door, given that the gate processing time is around 0.1 s. Therefore, the goal of development was to achieve high-speed processing between the ticket gate scanner and the IC card.

Thus, implementing solutions that reduce overcrowding helps enhance passenger safety and reduces the risk of accidents or incidents. Passengers experience shorter wait times and reduced travel durations, making their journeys more convenient and time-efficient. Also reduces stress and anxiety among passengers and driver. Overall, implementing solutions to address crowded public transportation improves the overall quality of public transportation services, enhances the passenger experience, and promotes sustainable and efficient urban mobility.

1.2 Problem Statement

The existing system relies on paper tickets or cash transactions, leading to inefficiencies such as time-consuming ticketing processes, potential errors, and delays for passengers and bus operators. Cash transactions onboard buses also create challenges in terms of handling change, accounting, and security risks.

The current lack of a reliable real-time tracking system poses several issues. Firstly, without accurate real-time tracking data, it becomes difficult to provide passengers with timely and accurate information regarding bus arrival times, leading to inconvenience and frustration. Secondly, ineffective resource allocation due to the absence of real-time tracking can result in inefficient bus deployment, overcrowding on certain routes, and underutilization of buses on others[1]. Furthermore, the lack of a robust tracking system makes it challenging to track the maintenance needs, fuel consumption, and overall performance of the bus fleet, resulting in increased costs and reduced reliability. These limitations underscore the importance of implementing a reliable bus tracking system to enhance passenger information, optimize resource allocation, improve route planning, and ensure effective fleet management in the transportation system.

1.3 Objectives

The main objectives of the ”Digital Bus Fare Management System” project is:

• Integrate the digital payment system with existing bus infrastructure to create a modern and data-driven transportation network.

1.4 Scope of Project

The scope of the ”Digital Bus Fare Management System” project encompasses the development and implementation of a robust system for the automated payment and hassle-free passenger experience in public transportation. The project aims to create a sustainable solution that addresses the challenges of traditional transport system. The key aspects of the project scope include:

The scope of this project encompasses the development and implementation of a comprehensive system that integrates hardware and software components. The project focuses on key areas like, Hardware Integration, Software Development, User Management, Fare validation and deduction, Payment Integration, Real-Time Bus Tracking and Reporting and Analytics.

The system's hardware integration includes the installation of scanners on buses and the integration of the SIM808 module for real-time communication. This allows passengers to tap their cards on the onboard scanners for seamless fare validation and deduction. The software development involves the creation of a Node server and a secure database to manage user registration, fare deduction, recharge history, and transaction logs.

Passengers will have the convenience of multiple payment options to recharge their cards, including credit/debit cards, mobile wallets, and cash payments at designated recharge points. Additionally, the system provides real-time bus tracking, enabling passengers to monitor bus locations and plan their journeys more efficiently.

The project also focuses on user management, securely storing and managing passenger data such as contact details and transaction history. Compliance with regulatory standards set by the Department of Transport Management, Nepal, ensures adherence to fare rates and taxes. Furthermore, the system generates comprehensive reports and analytics based on fare transactions, recharge history, and passenger usage patterns. This information empowers bus operators to make data-driven decisions, optimize routes, and improve overall operational efficiency.

The scope of the project encompasses the development, implementation, and deployment of a comprehensive system that integrates hardware and software components. It emphasizes enhancing the passenger experience, improving operational efficiency, and providing a secure and scalable solution for fare collection in the public transportation system.

1.5 Feasibility Analysis

The feasibility of our project can be evaluated in terms of technical, economical and operational feasibility.

1.5.1 Technical Feasibility

The project leverages existing technologies that are readily available and widely used. The use of GPS tracking enables real-time bus tracking and use of cards and scanners streamlines the fare calculation process by automating user authentication. While web applications and mobile platforms offer a user-friendly interface for card balance checks, recharge, and revenue analysis. These technologies have been proven effective in similar systems, ensuring the technical feasibility of the proposed solution.

1.5.2 Economic Feasibility

The project is economically feasible as it is expected to be cost-effective. The system requires minimal specialized equipment and trained personnel. In addition, the results obtained from the project can be used to determine the optimal route for the bus and analyze the passenger's travelling pattern and routine to result in an increase in profit.

1.5.3 Operational Feasibility

The project is also operationally feasible as it involves automatic fare calculation and reduced manual processes. The system simplifies the ticketing process for both passengers and bus operators. The user-friendly interface makes it easy for passengers to track buses, check their card balance, and recharge their cards, enhancing the overall user experience. These operational improvements contribute to the feasibility and practicality of the proposed system.

CHAPTER 2: LITERATURE REVIEW

The use of RFID and SIM module for bus tracking and digital fare management has been the subject of numerous studies in the literature. A literature review for this project would involve researching and analyzing existing studies, articles, and publications related to use of RFID and SIM module for Bus tracking and fare calculation.The review covers key aspects such as Bus tacking, digital fare collection and application of digital card system.

The study on potential of public transport smart card data by Bagchi, M., White, P. R. (2005) explores the potential of smart card data analysis in understanding passenger behavior, planning services, for improved decision-making.[2]

The research by Bonneau, W. and editors (2002) on the role of smart cards in mass transit systems examines the benefits and applications of smart cards in mass transit systems, including their impact on fare collection efficiency.[3]

Another research conducted by Clarke, R. (2001) on Person location and person tracking: Technologies, risks, and policy implications discusses person location and tracking technologies, including smart cards, and explores associated risks and policy implications.[4]

Smart card evolution by Shelfer, M., Procaccino, J. D. (2002) provides an overview of the evolution and applications of smart card technology, including its relevance in transportation systems.[5]

Origin and Destination Estimation in New York City with Automated Fare System Data done by Barry, J. J., Newhouser, R., Rahbee, A., Sayeda, S. (2002) explores the use of automated fare system data for estimating origin and destination patterns in New York City.[6]

Design and development of GSM and GPS (Global Positioning System) tracking module by U. Bharavi and R. M. Sukesh (2017) discusses the design and development of a GSM and GPS tracking module, highlighting its potential applications in tracking and monitoring systems for fleet management, and location-based services.[7]

Automatic bus fare collection system by using GPS and RFID technology by Karthika J, Varshanapriyaa S, Sai Haran S, SuriyaPrakash C (2020) discusses the effectiveness of automatic fare collection system compare to ticket based system. They discuss the paper based ticket system being one of the reason for financial loss in transportation in India.[8]

The Use of Smart Cards in Transportation Systems: A European Perspective by C.M. Shield,

P.T. Blythe (2017) discusses the use of smart card technology in the field of transport applications. They review some major areas of application and cites some of the more innovative implementations of the technology in Europe. The paper covers both contact and contactless (proximity) smart cards for transport applications, used on their own as a payment and information carrier device, or with a reader to provide contactless communications.[9]

Networked RFID: systems, software and services by George Roussos (2008) introduces the technologies and techniques of large-scale RFID-enabled mobile computing systems, set in the context of specific case studies in his book. The text explores RFID technology fundamentals, including operating principles, core system components and performance trade-offs involved in the selection of specific RFID platforms.[10]

Bus fare collection system using RFID and GPS bus fare collection system using RFID and GPS by Shazid Bin Zaman, Richard Victor Biswas and Eftakharul Islam Emon (20023) eliminates the need for paper tickets, offers onboard ticket inspection, and possibility of introducing spatial validation elements to enhance its usefulness.[11]

CHAPTER 3: HARDWARE DESCRIPTION

The major hardware components are RFID, NEO-6M GPS module, NodeMCU, LED and buzzer.

3.1 RFID

Tags and readers make up the wireless system known as radio frequency identification (RFID). RFID tags are compact electronic devices made up of an antenna and an integrated circuit (IC). Each tag is embedded with a unique identification code or data that can be read by RFID reader. There are different types of RFID tags:

- Passive RFID Tags: There is no independent power source for these tags and are activated by the radio frequency signal from the RFID reader.
- Active RFID Tags: These tags are powered by their own source (usually a battery) and can transmit signals independently.

Figure 3.1: RFID Card
(Source:[Online] Available: https://images.app.goo.gl/Dqbb6bFTXZh8iNVx6)

RFID reader sends out radio waves and gets signals back from RFID tags. It communicates with the tags through antennas.

Figure 3.2: RFID Reader
(Source:[Online] Available: https://images.app.goo.gl/w3e8HzYsctUdjqiQA)

3.2 NEO-6M GPS module

NEO-6M is a GPS module used for obtaining accurate time and location information. It is based on the u-blox NEO-6M GPS chipset. This chipset provides high sensitivity and fast time-to-first-fix (TTFF) performance. The module communicate with a microcontroller using USART (Universal Asynchronous Receiver-Transmitter) serial communication. It sends NMEA (National Marine Electronics Association) formatted data strings containing GPS information, such as latitude, longitude, altitude, speed, and satellite information. It requires an external active GPS antenna to receive signals from GPS satellites. The antenna is connected to the module through an SMA (SubMiniature version A) connector. The module operates on a 3.3V DC power supply and typically consumes around 45mA of current during normal operation. It support for GPS, GLONASS, and SBAS (Satellite-Based Augmentation System) satellite systems. Support for up to 50 channels for tracking satellites. Onboard EEPROM for storing configuration settings and aiding in faster satellite acquisition. It has a 2.5m horizontal position precision and can update its location five times in a second, which sets it apart from other GPS modules.

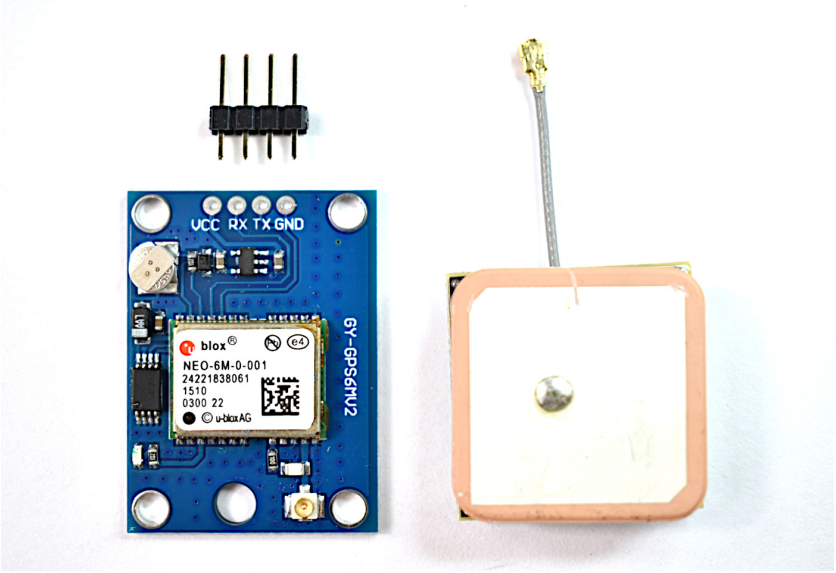


Figure 3.3: NEO-6M GPS Module
(Source:[Online] Available: https://images.app.goo.gl/r3CfNtwKBfkd7JvC7)

3.3 NodeMCU

It is an open-source development board based on the ESP8266 microcontroller. It has integrated WiFi connectivity and a range of GPIO pins for interfacing with sensors, actuators, and other devices. ESP8266 microcontroller features a 32-bit RISC CPU running at 80 MHz and integrated Wi-Fi connectivity. The General Purpose Input/Output (GPIO) pins of the NodeMCU are utilized to communicate with various devices like as an RFID reader, buzzer, LED, and GPS module. The NodeMCU includes a USB-TTL converter chip, which allows it to be easily programmed and powered via USB. NodeMCU was originally delivered with software written in the Lua programming language, which made it straightforward to develop and upload code to the board. But, it can also be programmed using the Arduino IDE or other development environments.

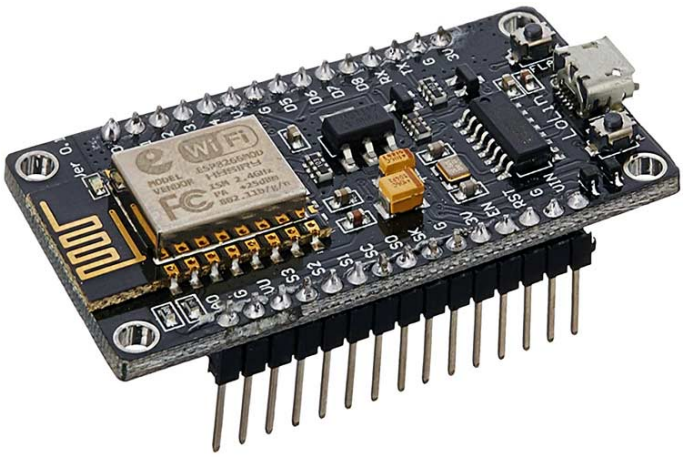


Figure 3.4: NodeMCU EPS8266
(Source:[Online] Available: https://images.app.goo.gl/HTDne6T3HBKyMycT7)

3.3.1 Pin Configuration

- Power Pins: One VIN pin and three 3.3V pins.
  - This pin enables the NodeMCU and its peripherals to receive external power supplies (up to 5V). This is converted to 3.3V for internal operations via the onboard regulator.
  - These pins supply controlled 3.3V power, which is obtained via VIN, to power external parts that are attached to the NodeMCU.
- GND is ground pin.
- I2C sensors and peripherals are connected using I2C pins. It should be mentioned that the I2C clock frequency needs to be higher than the slave device's slowest clock frequency.
- GPIO Pins: The 17 GPIO pins of the NodeMCU/ESP8266 can be dynamically allocated to various functions, including I2C, I2S, UART, PWM, IR remote control, LED light, and button.
- ADC channel: ADC can be utilized to implement the two tasks. Testing the input voltage of the TOUT pin and the power supply voltage of the VDD3P3 pin.
- USART Pins: NodeMCU can interact at up to 4.5 Mbps and support asynchronous protocols (RS232 and RS485). It has two ports(UART0, UART1)
- SDIO Pins: SD cards can be directly interfaced with NodeMCU/ESP8266 thanks to its Secure Digital Input/Output Interface (SDIO) capability.

- PWM Pins: There are four Pulse Width Modulation (PWM) channels on the board. Programmatically implemented PWM output can be utilized to drive LEDs and digital motors..
- To operate the NodeMCU/ESP8266, utilize the control pins. These pins are the WAKE pin, the reset pin (RST), and the chip enable pin (EN).
  - EN: When the EN pin is pulled HIGH, the ESP8266 device is activated. Pulling LOW uses the least amount of power for the chip.
  - RST: The ESP8266 chip can be reset using the RST pin.
  - WAKE: The processor can be awaken from deep sleep via the wake pin.

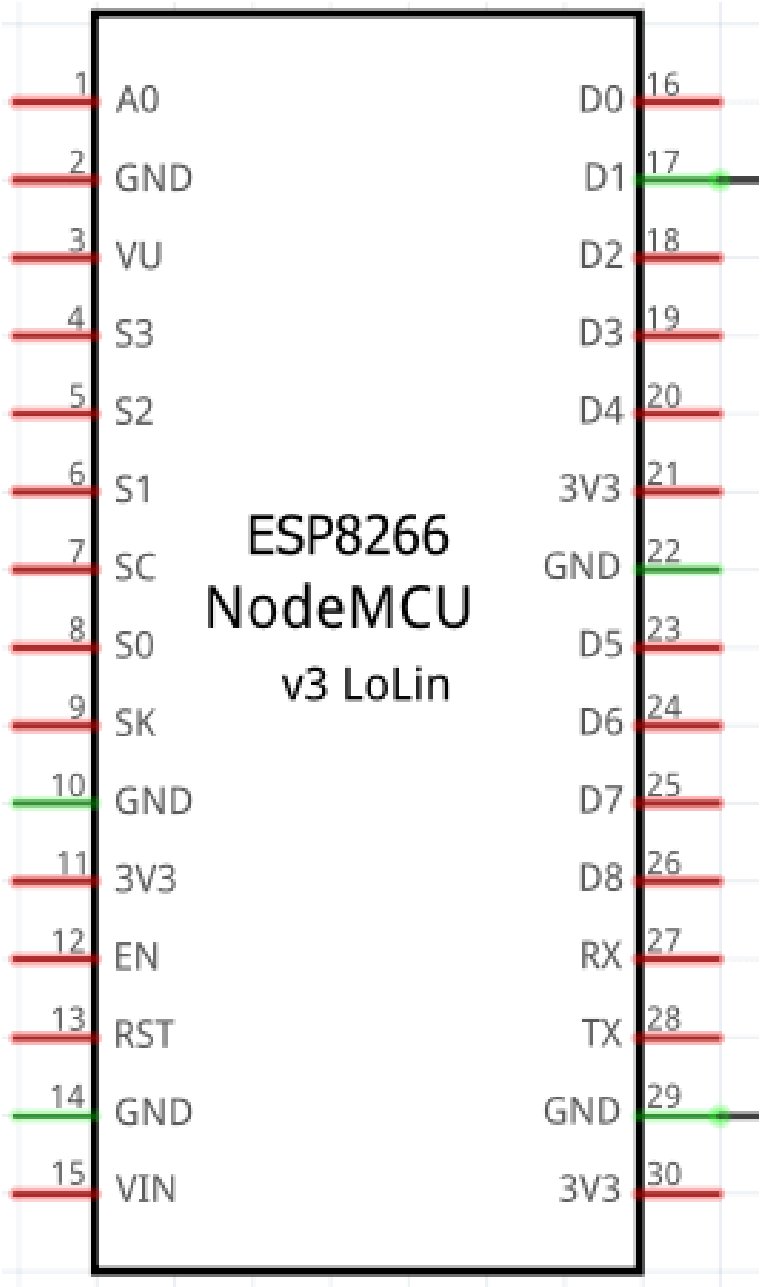


Figure 3.5: Pin Configuration of NodeMCU EPS8266
(Source:[Online] Available: https://images.app.goo.gl/LErsjrzUD1yNUSUA7)

CHAPTER 4: SOFTWARE IMPLEMENTATION

4.1 Next JS

Next.js is a React framework for creating full-stack web apps. React Components is used for UI development, whereas Next.js is used for extra features and optimizations. Beneath the surface, Next.js also abstracts and automatically sets up the necessary React tooling, including bundling, compilation, and other tasks.

A React framework called Next.js is used to create full-stack web apps. React Components is used to create user interfaces, and Next.js is used for extra features and optimizations. Beneath the surface, Next.js also abstracts and automatically sets up the necessary React tools, including bundlers, compilers, and more.In order to maximize SEO performance, NextJS reduces the sluggish rendering and loading times that come with client-side rendering. Its out-of-the-box integration of server-side rendering improves both user experience and overall development efficiency.

Incremental Static Regeneration and static site generation are also supported by Next.js; typically, a compiled version of the website is created at build time and saved in a.next subdirectory. The pre-built version, which consists of static HTML pages, is cached and given to the user upon request. This results in an extremely fast load time, but it isn't appropriate for all websites, especially interactive ones that require a lot of user interaction and change frequently.[12]

4.2 Node JS

An open-source, cross-platform JavaScript runtime environment is called Node.js. It is a well-liked instrument for practically any type of undertaking.Outside of the browser, Node.js powers Google Chrome's core, the V8 JavaScript engine. This enables the high performance of Node.js.A Node.js application doesn't start a new thread for each request; instead, it operates in a single process. Blocking behavior in JavaScript code is avoided by Node.js's standard library, which includes a set of asynchronous I/O primitives. Additionally, libraries created in Node.js are often designed to avoid blocking, thus blocking is the exception rather than the rule.

Rather than halting the thread and consuming CPU cycles while it waits for a response, Node.js will continue its I/O activities when it receives one, such as reading from the network, accessing a database, or reading from the disk.Because of this, Node.js can manage hundreds of connections at once with a single server without adding the complexity of thread concurrency management, which might be a major cause of errors.

One special benefit of Node.js is that it doesn't need learning a new language—millions of frontend developers who write JavaScript for the browser can now write server-side code as well as client-side code.

The new ECMAScript standards work well with Node.js because you don't have to wait for every user to update their browser; instead, you can choose which ECMAScript version to use by simply changing the version of Node.js. Additionally, you can use Node.js with flags to enable particular experimental features.[13]

4.3 MongoDb

MongoDB stores information as dynamic, JSON-like documents, allowing fields to alter over time and the data structure to be flexible from document to document.Working with data is made simple by the document model's mapping to the objects in your application code. Effective methods for accessing and analyzing your data include indexing, real-time aggregation, and ad hoc queries. Since MongoDB is fundamentally a distributed database, features like global distribution, horizontal scaling, and high availability are pre-installed and simple to use.

MongoDB is a NoSQL database management system available as open source. Conventional relational databases being replaced with NoSQL (not only SQL). NoSQL databases can be handy when handling big, dispersed data sets. Managing document-oriented data and storing and retrieving data are capabilities of MongoDB.MongoDB helps enterprises store a lot of data quickly while maintaining great performance. It is used for big-volume data storage. Ad hoc queries, indexing, load balancing, aggregation, server-side JavaScript execution, and other features are some of the other reasons why businesses utilize MongoDB.

The design of the NoSQL MongoDB database consists of collections and documents rather than the tables and rows found in relational databases. Key-value pairs, the fundamental building block of data in MongoDB, make up documents. Document sets are contained in collections, which are similar to SQL tables. Numerous programming languages, including C, C++, C, Go, Java, Python, Ruby, and Swift, are supported by MongoDB..[14]

CHAPTER 5: METHODOLOGY

The methodology for implementing the project ”Digital Bus Fare Management System” involves several key steps, including hardware integration, software development and system assembly. These steps are essential to ensure the successful development and implementation of our purposed system.

5.1 System Development Life Cycle

The agile model for digital bus fare management system involves defining the project vision and goals, listing the works that needs to be done (product backlog), and planning short development iterations known as sprints. During each sprint, the team implements selected items from the backlog, collaborates with stakeholders for feedback, and conduct meetings in timely manner to address challenges. The iterative development approach allows for ongoing refinement and expansion of the system based on changing requirements. During each iteration it goes through different steps which are plan, design, develop, test, deploy and review. Finally, the system is launched.

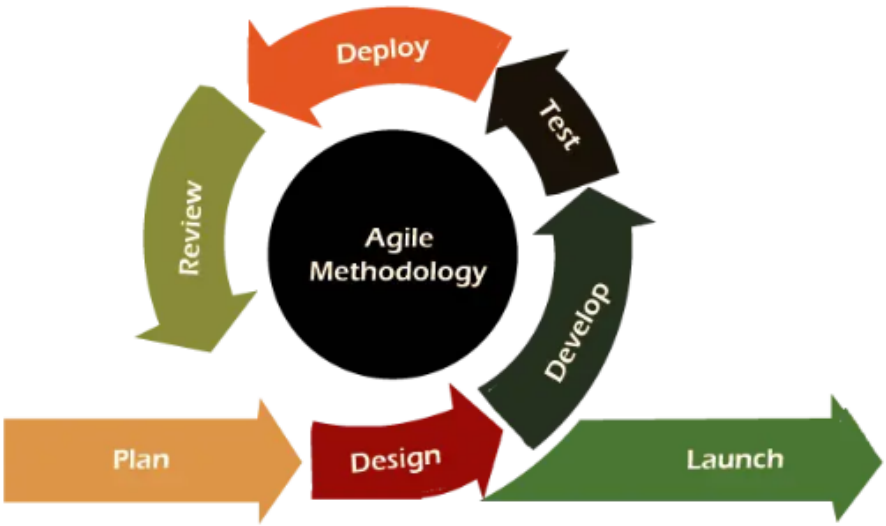


Figure 5.1: Agile Development Model
(Source:[Online] Available: https://images.app.goo.gl/pFAtfNn9xRqidS2p6)

5.2 System Requirement

The system needs to read RFID card from the passengers using RFID scanner.The system should accurately detect and verify RFID card. The system should calculate the fare of passenger according to the start point and end point and deduct the amount from total balance

available in user account. The system should track all registered buses and provide live location of the buses to the passenger.
The system should exhibit a high level of accuracy in identifying user and calculating fare. Each user is provided with a RFID card of unique ID which help to identify each user accurately. The system should read RFID quickly and efficiently to achieve optimal throughput and productivity. NodeMCU has baud rates of up to 115200 and can communicate at up 4.5 Mbps hence provide high throughput ans productivity. The system should be reliable and robust, capable of handing jerks and environmental factors. The RFID scanner is made compact and sustain minor jerks. Scalable systems should be able to manage a lot of buses and people. Asynchronous, non-blocking I/O operations are utilized by NodeJS to develop server-side web applications, which enable it to process several requests at once without experiencing a block. This makes it ideal for handling a large number of connections efficiently. The system should be designed in a modular and maintainable manner, allowing for the future updates, enhancements and repairs if needed. React JS uses a component-based architecture, allowing developer to create reusable component that can be used in different pages and section.The system should adhere to safety standards to ensure the protection of users data and efficient operation in the transportation management system. Validating user input to prevent injection attack.

5.3 System Design

The primary phase of the system development process is the design of the system. It involves converting the system requirements into architecture. Steps involved in system design are:

- Breaking down the system into smaller modules based on functional and non-functional requirement.
- Specifying how each components interact with each other and data flow.
- Defining the data structure for the system, including databases, tables, and relationships between data entities.
- creating a high-level architecture that outlines the system's general construction, incorporating the network topology, data flow, hardware, and software components.
- Defining the procedures that will be used by the system to perform its functions, including data processing and calculations.
- Designing the user interface for the system, including layout, navigation, and user interaction elements, to ensure usability and user satisfaction.

- Identifying and implementing security measures to protect the system from unauthorized access, data breaches, and other security threats.

5.4 System Block Diagram

The system's fundamental block diagram "Digital Fare Management System" is shown in figure below.

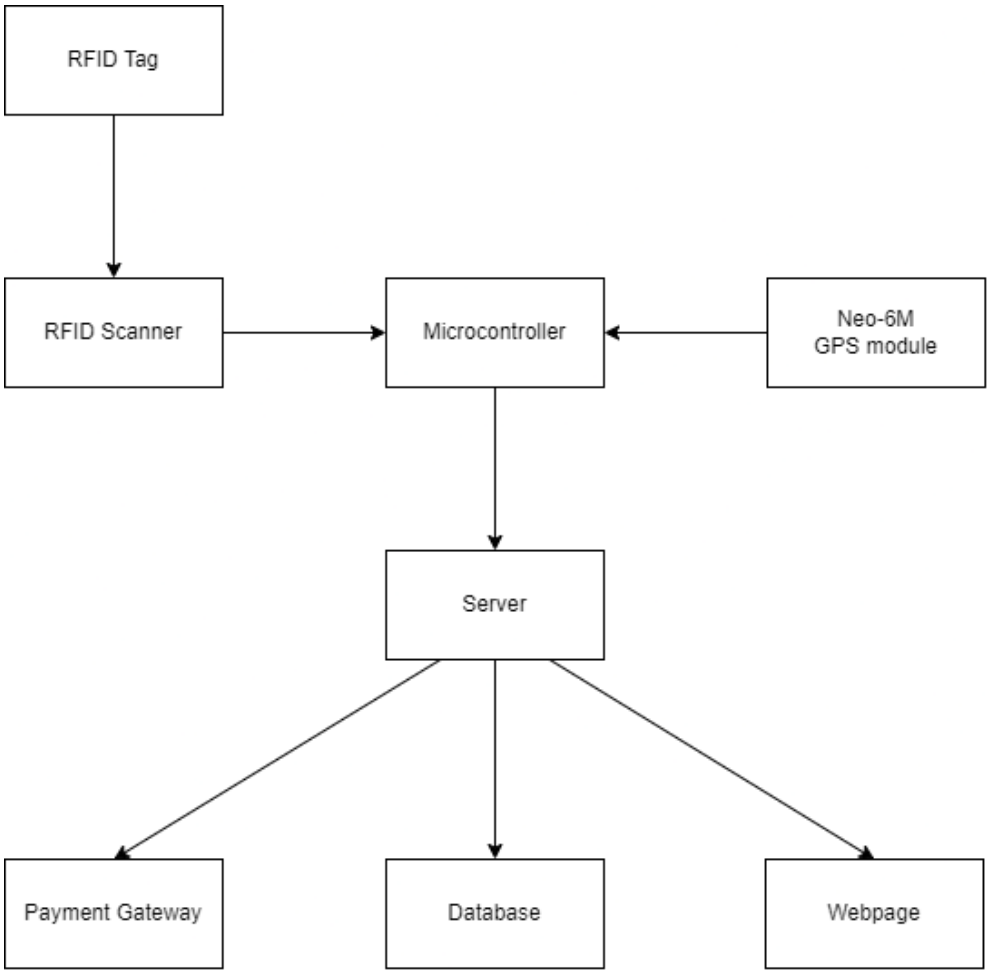


Figure 5.2: Block diagram

5.5 Circuit Diagram

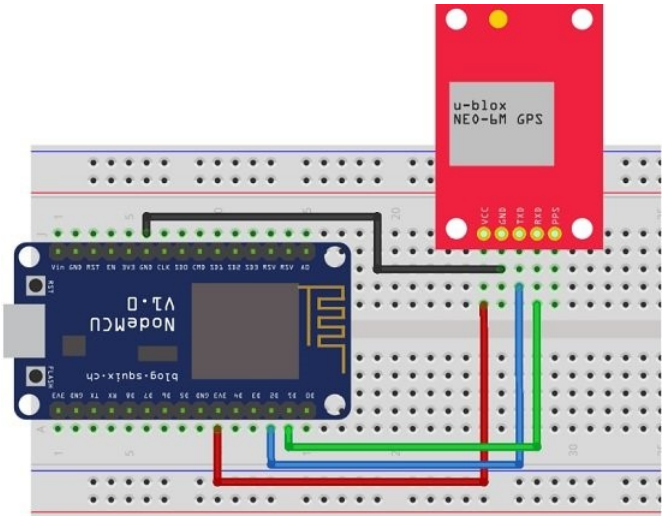


Figure 5.3: Connection between nodemcu and neo-6m gps module
(Source:[Online] Available: https://images.app.goo.gl/MJmP94wXHfN2yLPJ9)

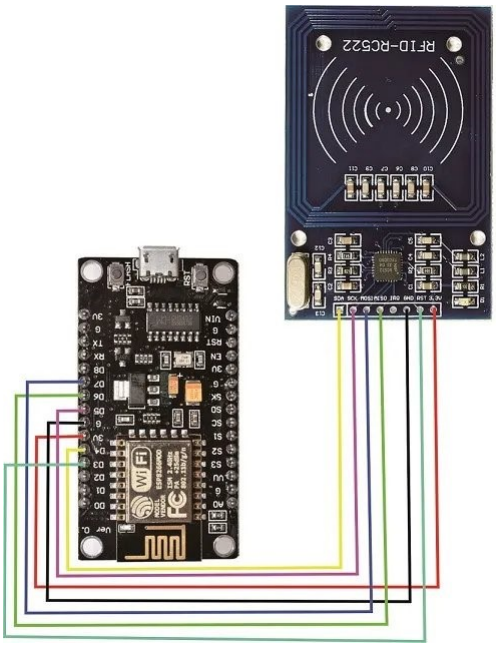


Figure 5.4: Connection between node mcu and RFID reader
(Source:[Online] Available: https://images.app.goo.gl/wj2x4osCYEeWRN1g9)

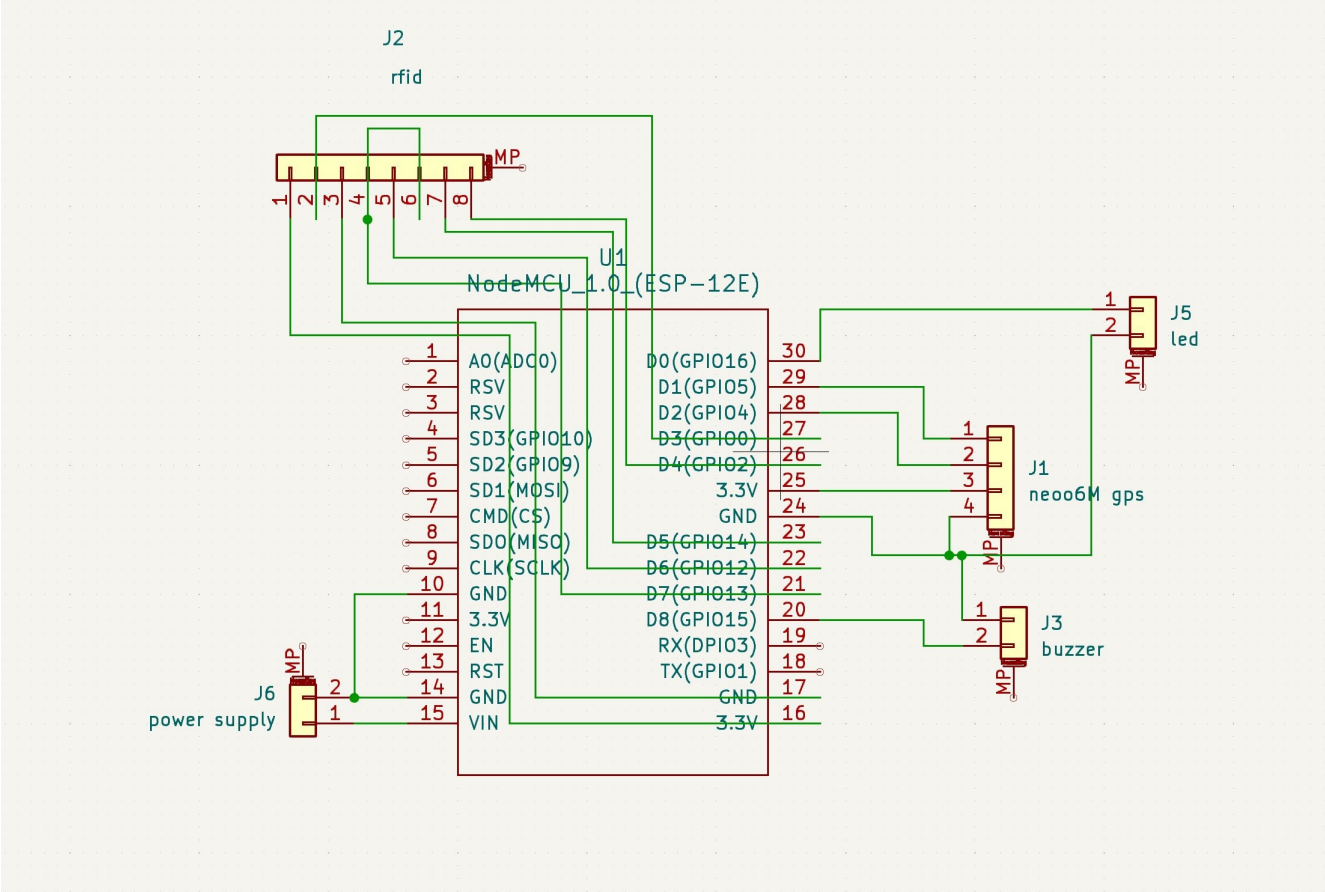


Figure 5.5: Circuit Diagram

In the circuit diagram, there are mainly seven components they are: buzzer ,led, resistor, nodemcu, rfid scanner, power supply, neo-6m gps module. In nodemcu, 11 pins D0, D1, D2, D3, D4, D5, D6, D7, D8, 3V, GND pin and micro usb type A port were used. In rfid scanner 7 pins SDA, SCK, MOSI, MISO, RST, 3.3V, GND pin were used.In neo-6m gps module, 4 pins Rx, Tx, Vcc, GND pin were used. Following circuit connection were made between RFID Scanner and NodeMCU. 3.3V of RFID to 3V, RST to D3 pin, SDA to D4, SCK to D5, MOSI to D6, MISO to D7, GND to GND. These connection set up communication between the RFID scanner and the NodeMCU utilizing the serial peripheral interface (SPI) convention. NodeMCU to Neo6m GPS Module.D1 to TX, D2 to RX, VCC to 3V, GND to GND.The positive terminal of buzzer is connected to D8 pin.Positive terminal of LED connected with Resistor to D0 and negative terminal to GND.Power is supplied through micro usb a port.

5.6 Flowchart

The flowchart explains how the system work in real time. The scanning device attempts to connect to wifi when its green LED is lit up or power is on. After the device has connected to the wifi, the green light will turn off and it will attempt to connect to the server. Once connected, it will retrieve the bus's current GPS coordinate and send it to the server. The device is now prepared to read RFID cards. The RFID reader will read the card and perform a validation check if it is positioned less than five centimeters away from it. The buzzer will sound invalid sound if the card is invalid; otherwise, it will sound valid beep if the card is legitimate. The RFID tag log with the current GPS coordinate is provided to the server after the RFID card has been confirmed. Until the power is turned off, this cycle will continue.

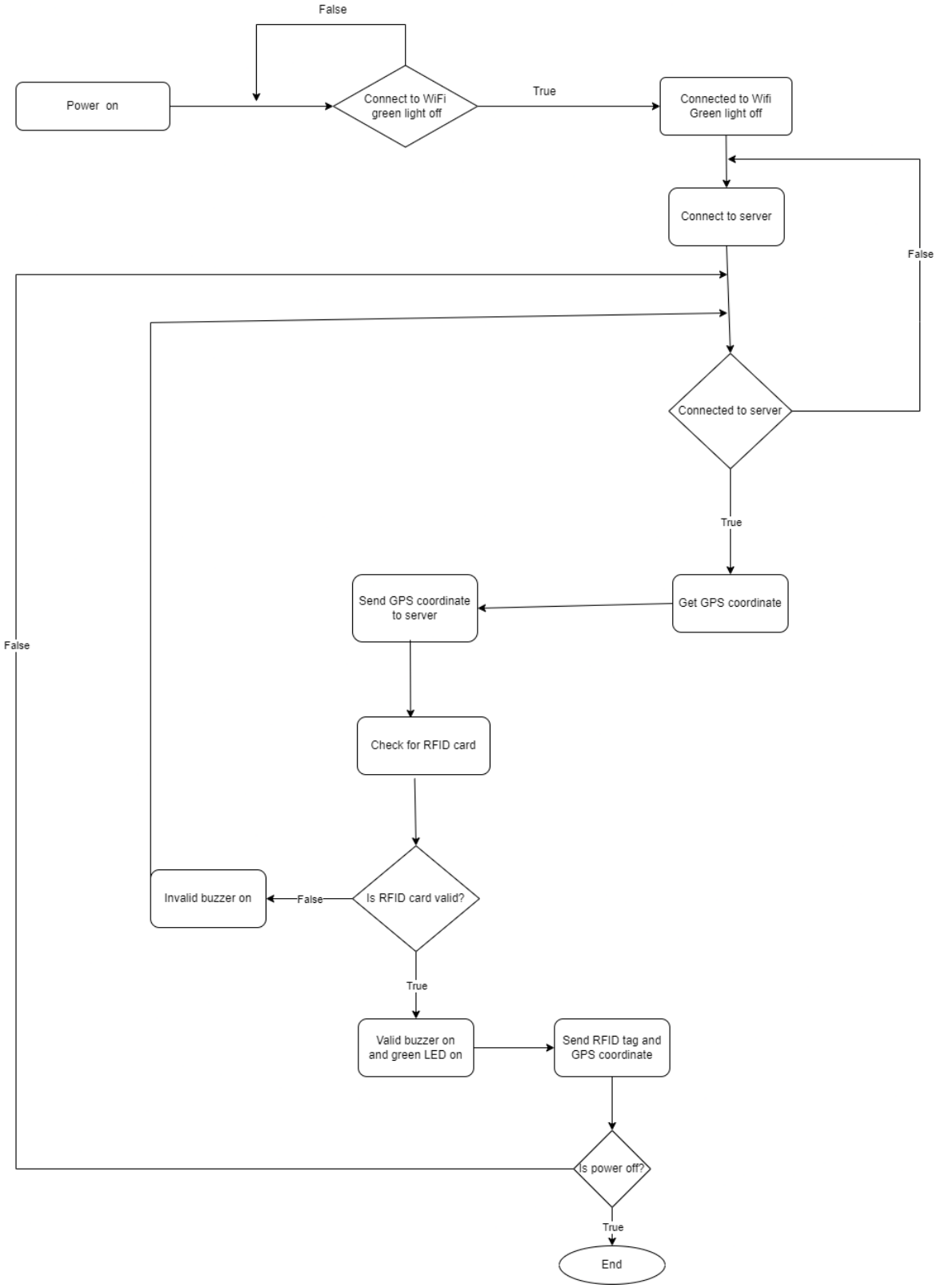


Figure 5.6: General Work Flow of System

5.6.1 Software Flowchart

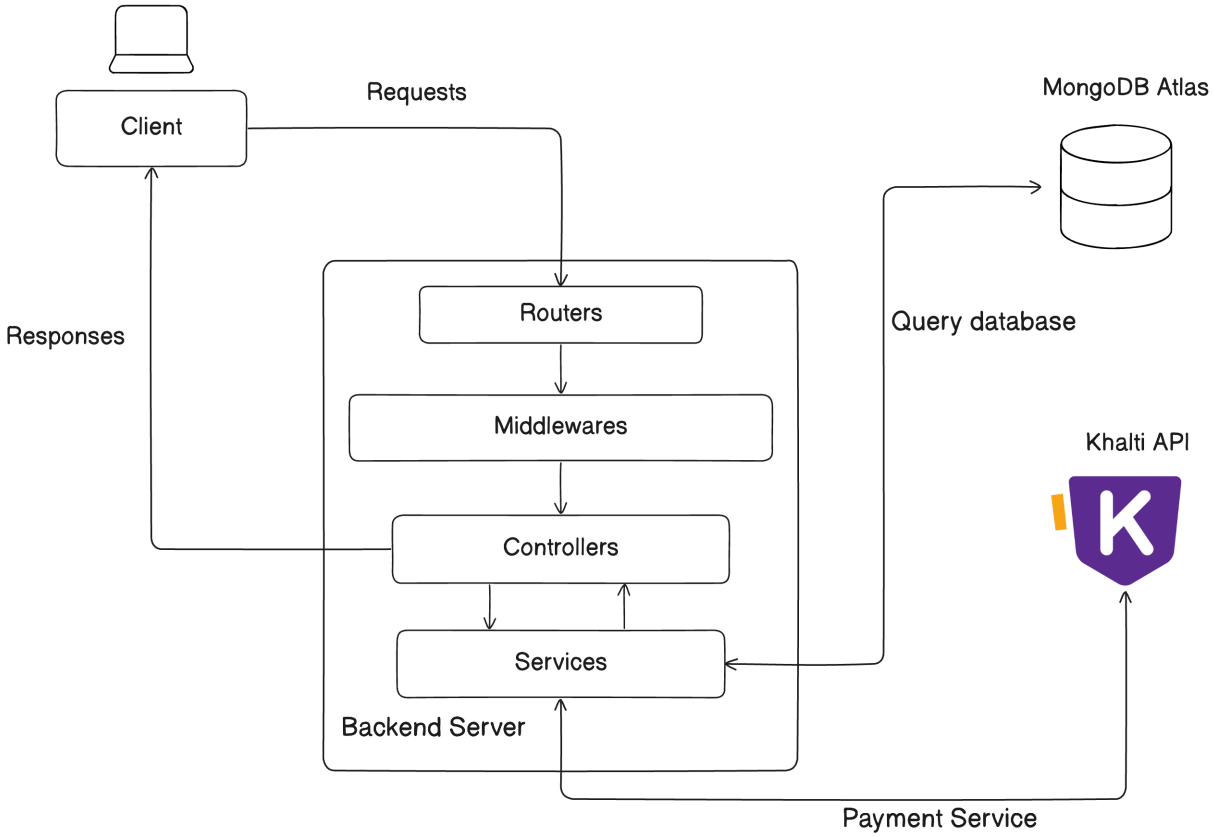


Figure 5.7: Software Flowchart

A backend server has components like routers, middleware, controllers, and services to manage incoming requests and data flow. Requests from clients are first routed through the server's routers, which direct them to the appropriate endpoints. Middleware layers intercept these requests, providing essential functionalities such as authentication, logging, and request preprocessing. Following middleware processing, requests are handed over to controllers, responsible for executing business logic and coordinating interactions with databases or external services. Controllers delegate tasks to services, encapsulating application logic, data manipulation, and service integrations, such as interacting with a MongoDB database for data storage and retrieval. Additionally, if the request involves payment processing, the services interact with payment gateway Khalti to facilitate transactions securely. Once operations are completed, responses are crafted and returned to clients, completing the server's workflow. This structured approach ensures efficient request handling, modular development, and scalable architecture within backend systems.

5.7 Database Schema

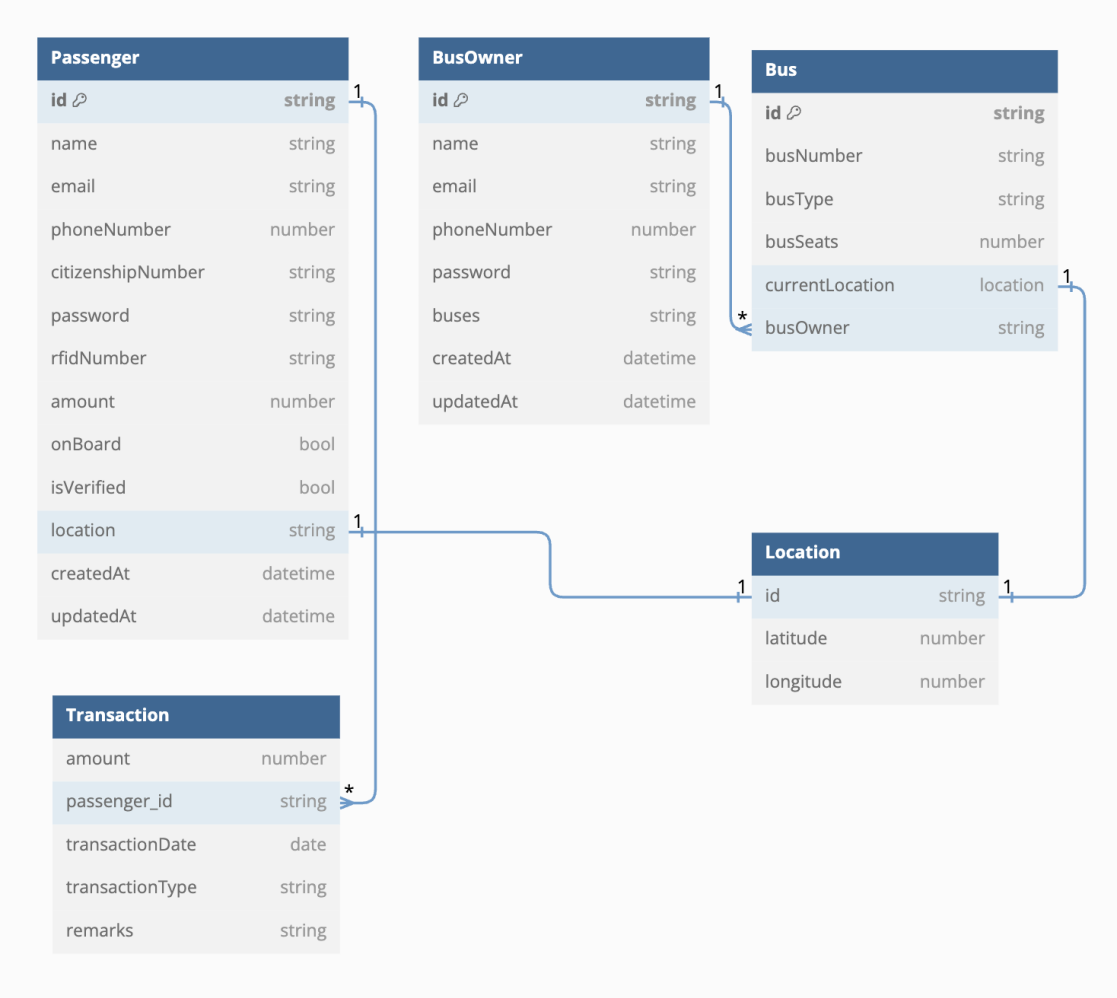


Figure 5.8: Schema Diagram

The database schema explains the database structure of the system for. It contains five tables passengers, bus owners, buses, locations, and transactions.
Passenger Table: Records details of passengers, including their name, email, phone number, citizenship number, RFID number, account balance, onboard status, verification status, current location, and timestamps for creation and updates.
BusOwner Table: Contains information about bus owners, such as their name, email, phone number, password, buses they own, and timestamps.
Bus Table: Stores data related to buses, including their number, type, seating capacity, current location, and reference to the bus owner.
Location Table: Holds geographical coordinates (latitude and longitude). Transaction Table: Records financial transactions involving passengers, including the transaction amount, passenger ID, transaction date, type, and any additional remarks.

Relationships between tables the tables aer as follows:

Passengers and Transactions: Linked through passenger ID in transaction referencing the passenger's ID.
Buses and Owners: Linked through busOwner in Bus referencing the ID in BusOwner.
Buses and Locations: Both bus and passenger tables reference the location table for their current location.

5.8 Khalti Payment Diagram

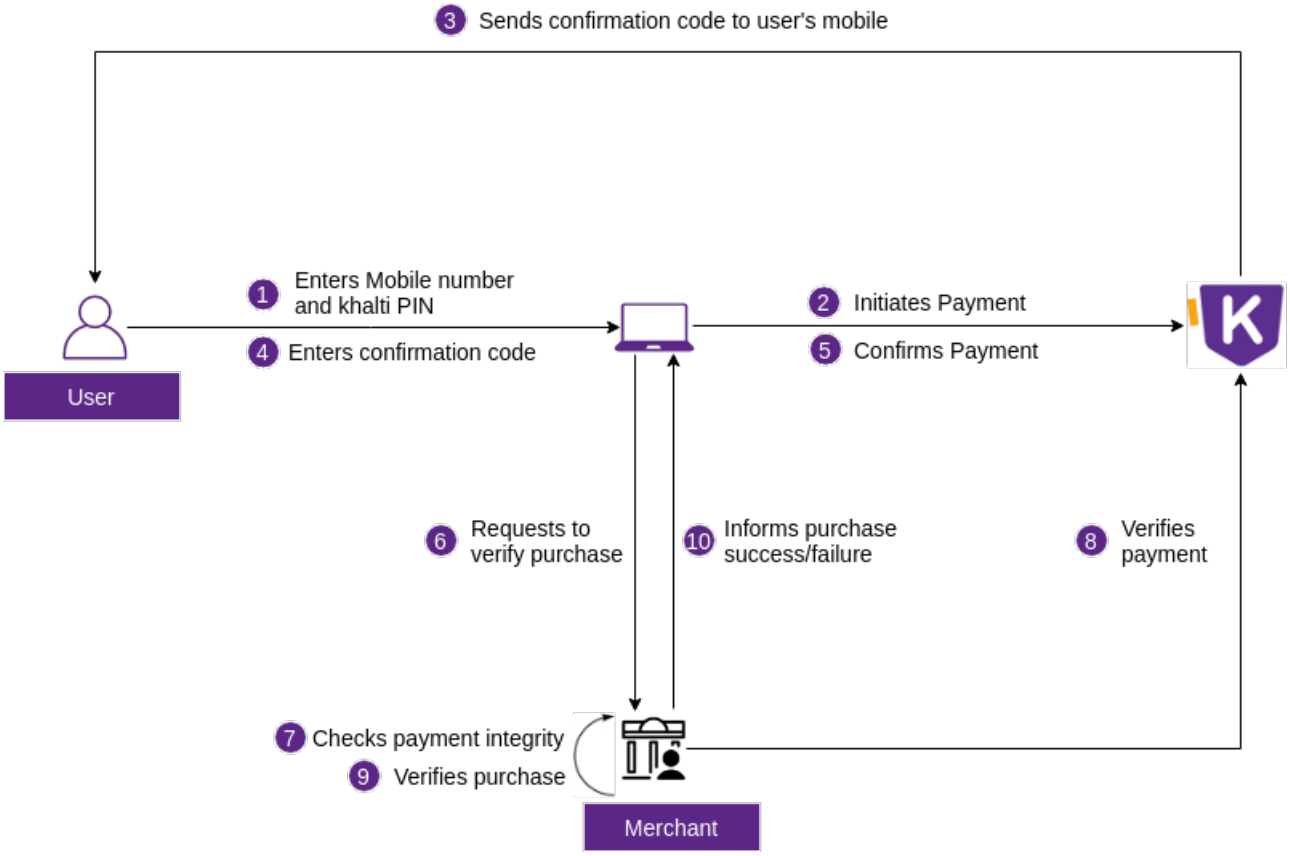


Figure 5.9: Khalti Payment Diagram
(Source:[Online] Available: https://docs.khalti.com/img/khalti-payment-new-overview.png)

5.9 Analysis Model

The construction of the analysis model, which focuses on organizing and formalizing the system requirements, is necessary to produce a model of the system that is accurate, comprehensive, and consistent. Three models make up the analysis model: the dynamic, object, and functional models. Use case diagrams can be used to describe the functional model. Activity diagrams and sequence diagrams are further ways to characterize dynamic models. Using use case and activity diagrams, we have explained the analytical model for this project in terms of the functional model and dynamic models.

5.9.1 Use Case Diagram

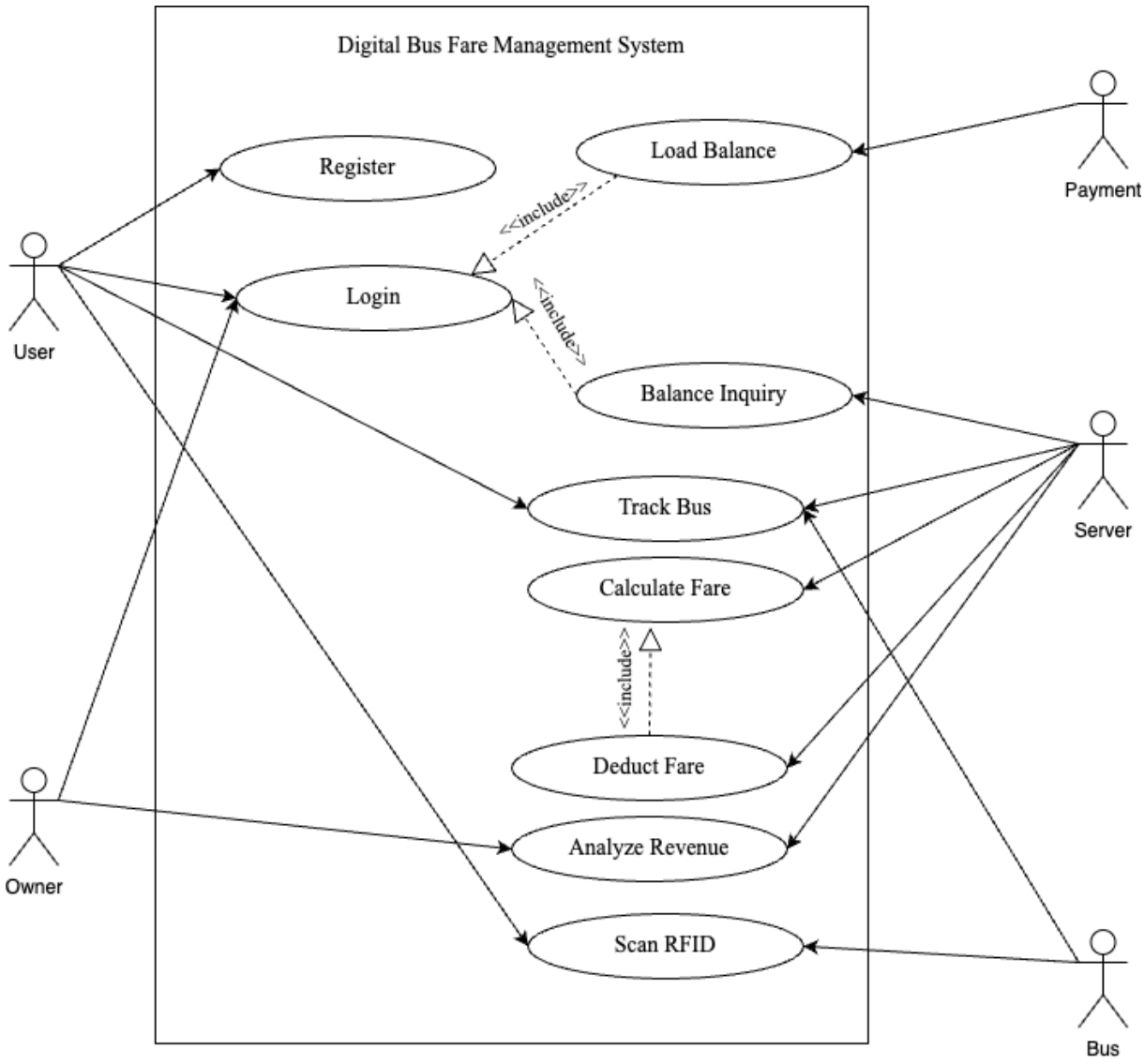


Figure 5.10: Use Case diagram

5.9.2 Activity Diagram

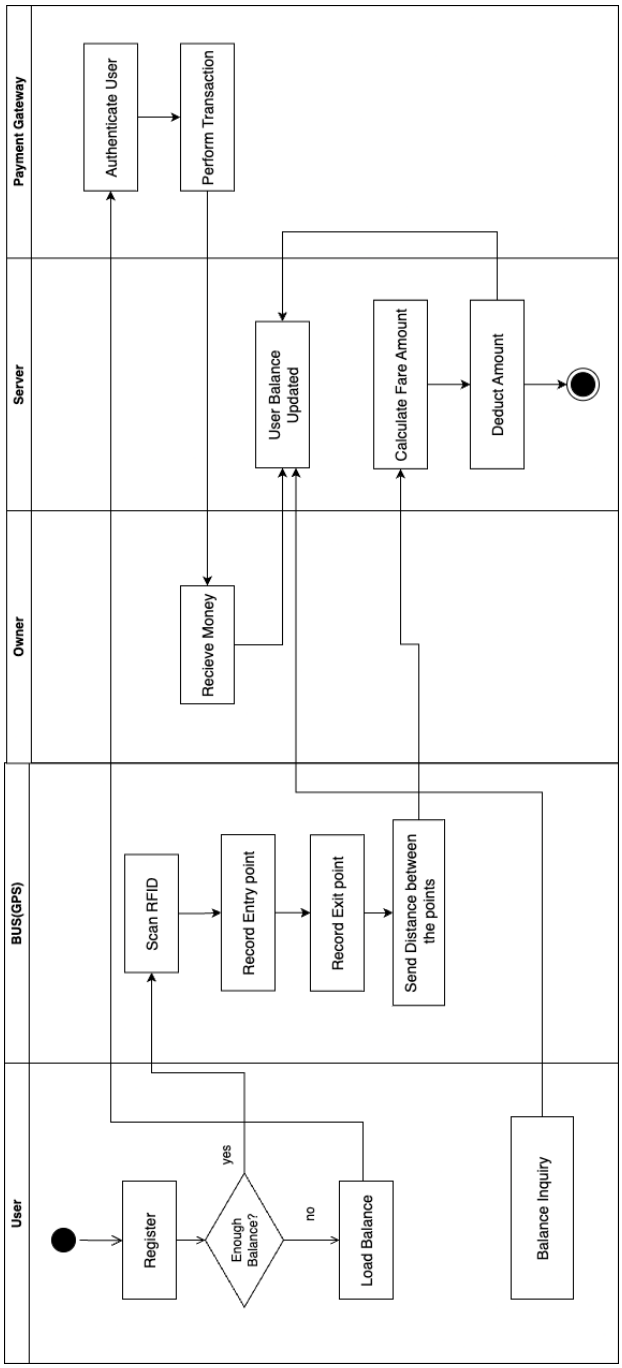


Figure 5.11: Activity diagram

CHAPTER 6: FEATURES AND FUNCTIONALITIES

6.1 Features and Functionalities

Features and functionalities of Digital Fare Management System:

6.1.1 User Authentication and Access Control

- Secure login for admin, owners and passengers.
- Role based access control to ensure appropriate access to system functionalities and data.

6.1.2 Passenger Management

- A new general user or passenger can be registered and then login.
- User can view available balance and can load balance using khalti payment gateway.
- User can view buses locations and know the exact time of arrival of bus.

6.1.3 Owner Management

- A bus owner can be registered and login.
- Bus owner can request to add new bus and remove registered bus.
- User can view daily revenue collected.
- Owners can view their buses live location.
- Owners can analyse their revenue in different (daily, weekly, monthly etc) time frame.

6.1.4 Admin Access and Control

- Admin can login.
- The administrator can verify the general user and the bus owner, and give the general user a unique RFID tag.
- Admin can view all registered general user and bus owner.
- The admin can delete user data account.
- The administrator has the ability to add new buses to or delete buses from the owner's registered bus registry.

6.2 User Manual

6.2.1 User Login

Users are greeted with the Sahaj Yatra homepage when they visit the website. The user needs to register with accurate information if this is their first time visiting the website. And if user already has an account they can login by provide Id and password. Admin login and superadmin login options are also present in the top navigation bar.

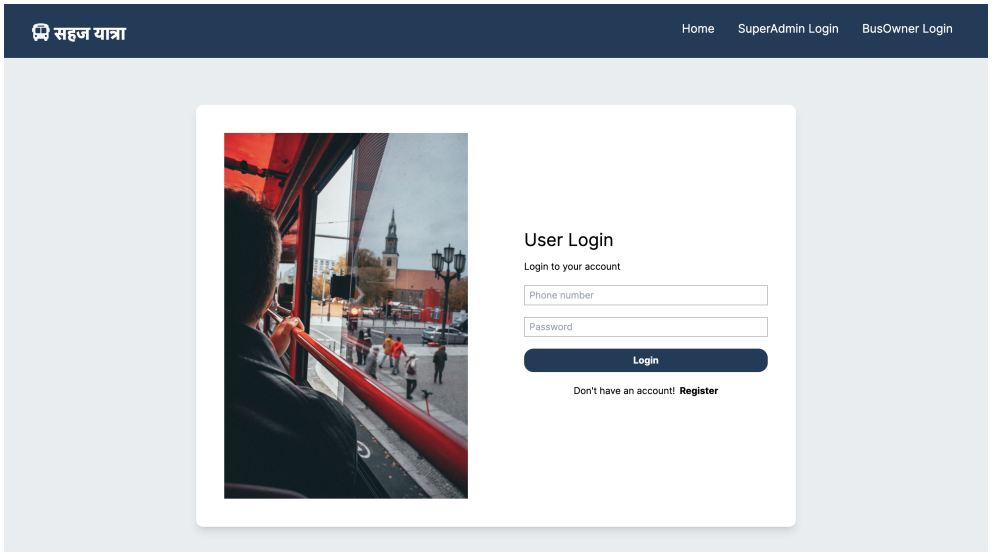


Figure 6.1: Home page

6.2.2 User panel

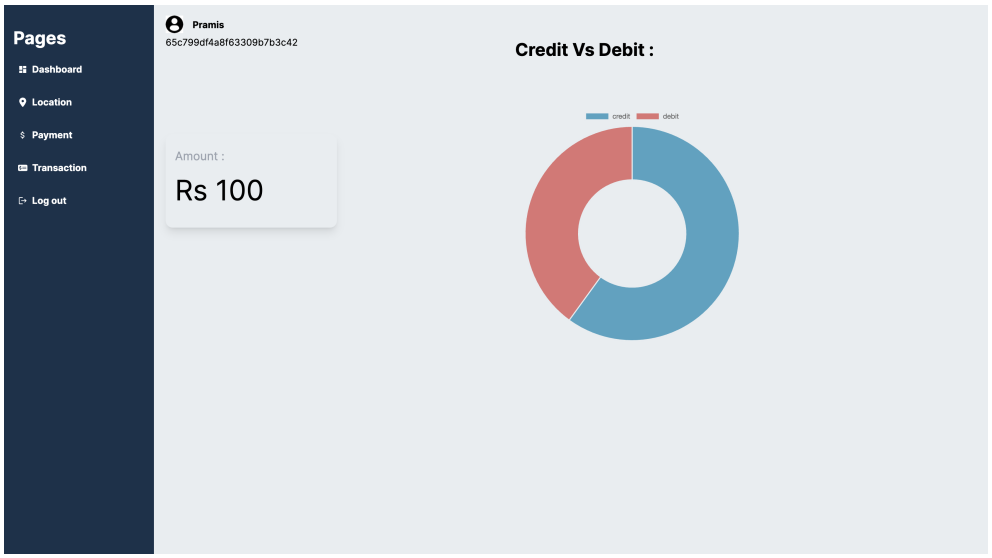


Figure 6.2: User Dashboard

- Dashboard: Upon logging in, users will be directed to the dashboard. The dashboard

provides an overview of essential metrics and functionalities, including the user's balance amount.

- Location Map: Users can follow the whereabouts of buses in real time with the location map feature.
- Payment: Payment feature allows user to load their balance using Khalti.
- Transaction History: The transaction history page provides overview of all transactions, including credits and debits, with details such as transaction amount, type, date, and remarks.
- Logout: After logging out, users will be redirected to the landing page to access the system again.

6.2.3 Owner Panel

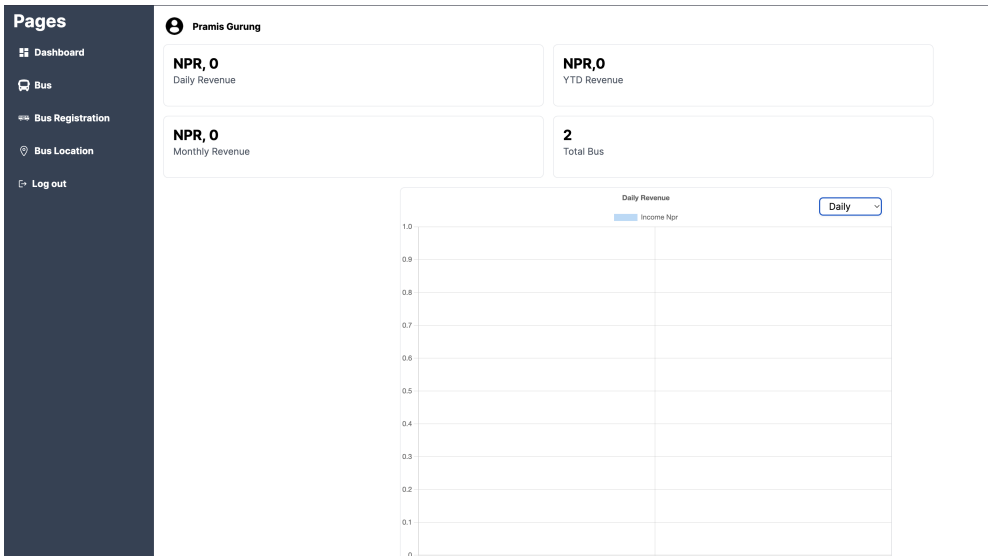


Figure 6.3: Owner Dashboard

- Dashboard: After login, the owner dashboard shows revenue in different time frame, number of buses registered and line graph of revenue over time.
- Bus: The owner can view number of buses registered by clicking bus button in navigation bar.
- Registration: Owner can also register new bus from registration.
- Log Out: Finally owner can log out of account using log out.

CHAPTER 7: EPILOGUE

7.1 Result and Discussion

The implementation of the digital fare management system provided efficient operation by offering seamless fare validation and convenient payment options. Bus owners can easily compare their daily and weekly revenue by logging into the owner dashboard. Because RFID cards are reusable, they are far more useful than the ticketing system that uses paper. Digital fare management systems, as opposed to traditional ones, address a number of issues, including dishonest employees, excessive operating expenses, and fare invasion.
That being said, there are also some limitations and challenges in this system:

- The first challenge of this system is that people might not like to carry an extra card to travel by bus. Therefore, it will take time for people to get used to this system.
- RFID card can be easily cloned by copying unique identification number. Smart card can be better options than RFID.
- Scanner can only be able to read RFID card, if it is placed at a distance less than 5cm from scanner.

7.2 Future Enhancements

There are several rooms for further development and enhancements to maximize the potential of the digital fare management system. Some of the possible future enhancements are as follows:

- After collecting enough data, we can use the data to visualize the movement patterns of passengers. And train a model that assigns the required number of buses to a particular area and time, which overall minimizes the under-utilization and overcrowding of buses.
- RFID cards can be integrated into student ID cards, which helps track the movement of students if needed.
- By keeping record of the capacity of a particular bus, the number of passengers on the bus at a particular time can be monitored, and a warning can be sent to the bus driver.
- RFID can be replaced with a smart card, which has higher security, can store encrypted data, and supports secure authentication mechanisms. Further, this smart card can be used to store personnel details like bank account details and citizenship details and

is compatible with existing contactless payments, which will make it a multipurpose card.

- Instead of nodeMCU, raspberry pi can be used which has few more benefits, it has more powerful processor, has larger storage, can be integrated with other hardware components easily like using keypad, display,etc.

7.3 Conclusions

There are numerous problems with the conventional bus fare collection system, including high operating costs, dishonest employees, bus fare gouging, fare evasion, etc. This method can help solve the issue since it uses an RFID-based payment mechanism, which lowers the amount of labor needed, in the digital fare collection system. With this system the problems such as under utilization and over crowded of buses can be minimized which will benefit the passengers, bus owner and transportation administration. This project will benefits big cities like kathmandu, pokhara etc. Currently there is no existing automatic fare collection system in our country. But we believe developing of this system creates convenience to passenger and bus conductor to collect fare and provide travel service more efficiently. Overall, the above-mentioned chapters' goals and specifications have been successfully attained and fulfilled by this system.

REFERENCES

[1] W. M. M. H. Hassan, Kharim Tarek, "Intelligent transportation system real-time tracking," *Transport Policy, vol. 12*, pp. 2–7, 2015.

[2] P. R. W. M. Bagchi, "The potential of public transport smart card data," vol. 12. Springer, 2005, pp. 464–474.

[3] e. W. Bonneau, "The role of smart cards in mass transit systems," *Card Technology Today*, p. 10, 2002.

[4] R. Clarke, "Person location and person tracking: Technologies, risks and policy implications," *Information Technology People*, vol. 14, pp. 206–231, 2001.

[5] J. D. P. M. Shelfer, "Smart card evolution," *Communications of the ACM*, vol. 45, no. 7, pp. 83–88, 2002.

[6] A. R. J. J. Barry, R. Newhouser, "Origin and destination estimation in new york city with automated fare system data," pp. 500–645, 2002.

[7] R. M. S. U. Bharavi, "Design and development of gsm and gps (global positioning system) tracking module," *Discusses the design and development of a GSM and GPS tracking module, highlighting its potential applications in tracking and monitoring systems for fleet management, and location-based services*, 2017.

[8] S. H. S. S. C. Karthika J, Varshanpriyaa S, "Automatic bus fare collection system by using gps and rfid technology," pp. 1119–1122, 2020.

[9] P. B. C.M. Shield, "The use of smart cards in transportation systems: A european perspective," *8th IFAC/IFIP/IFORS Symposium on Transportation Systems*, pp. 257–262, 2017.

[10] G. Roussos, *Networked RFID: Systems, Software and Services*. Springer London, 2008.

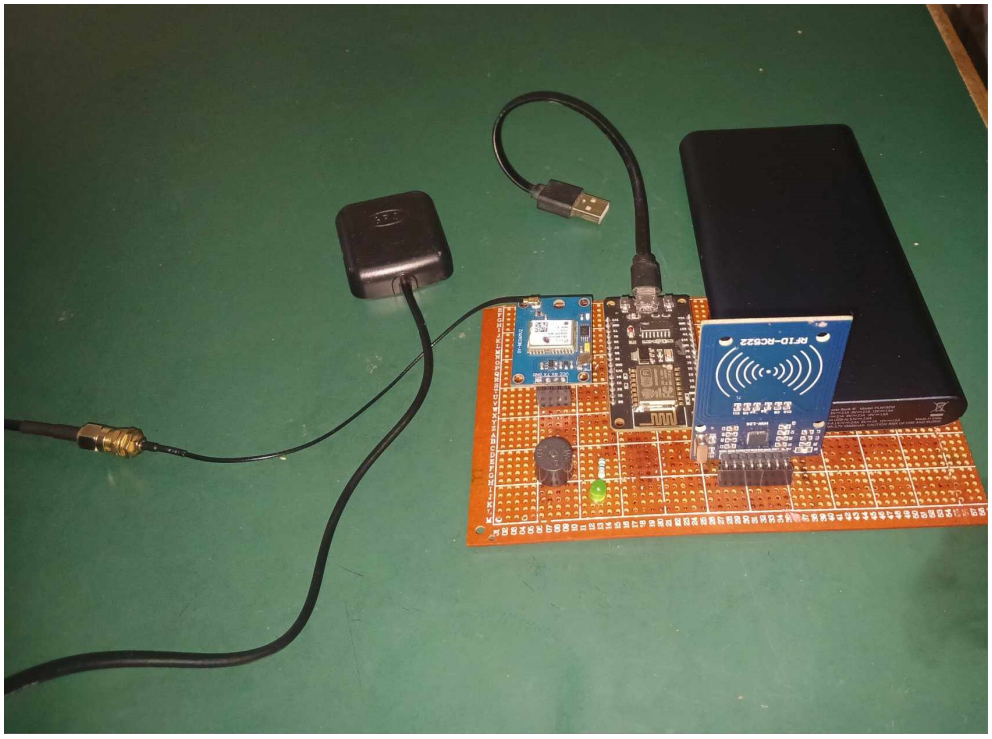
[11] R. V. B. a. I. E. Shazid Bin Zaman, "Bus fare collection system using rfid and gps bus fare collection system using rfid and gps," *EasyChair Preprint*, 2023.

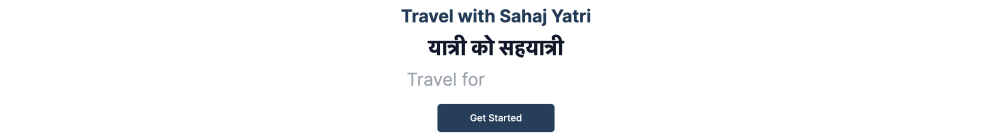
[12] [Online]. Available: https://nextjs.org/docs

[13] [Online]. Available: https://nodejs.org/en/learn/getting-started/introduction-to-nodejs

[14] [Online]. Available: https://www.mongodb.com/docs/

APPENDIX A: Snapshots of Project





About Us :

Our project "Digital Bus Fare Management System" project encompasses the development and implementation of a robust system for the automated payment and hassle-free passenger experience in public transportation. The project aims to create a sustainable solution that addresses the challenges of traditional transport system

Our Objective

Travel Efficiently

To introduce an efficient and convenient digital bus payment system that benefits both passengers and bus operators.

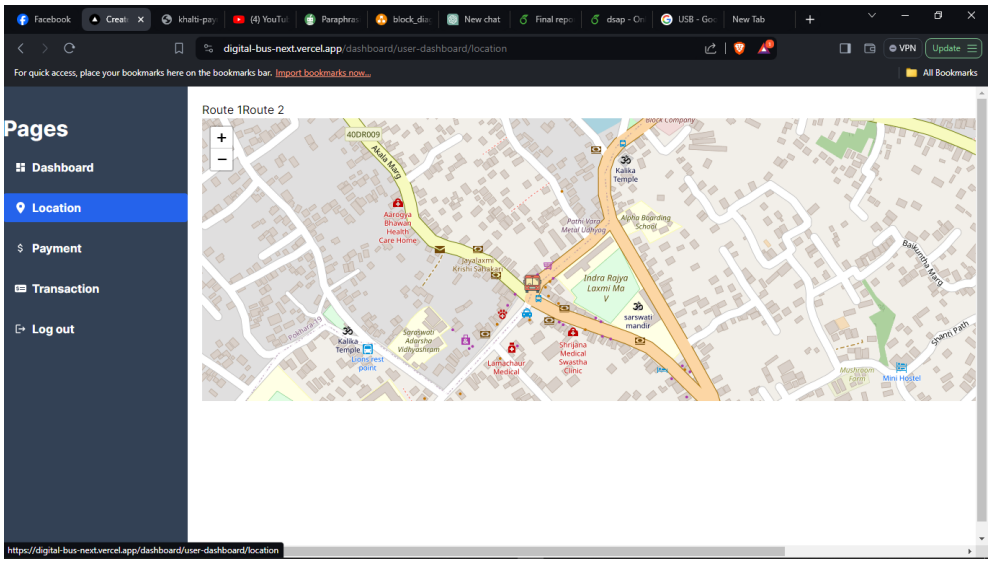
Get Started

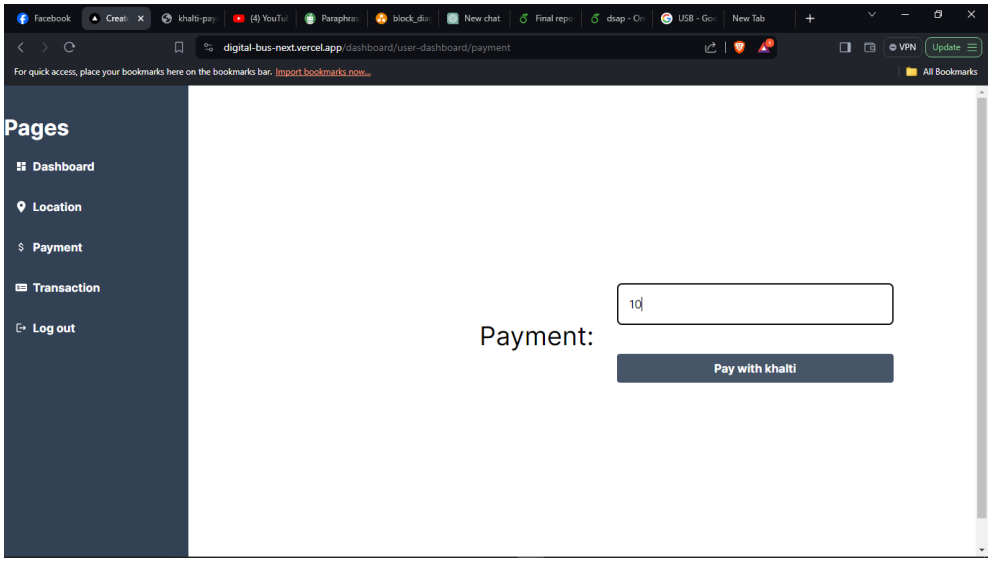
Sahaj Yatra

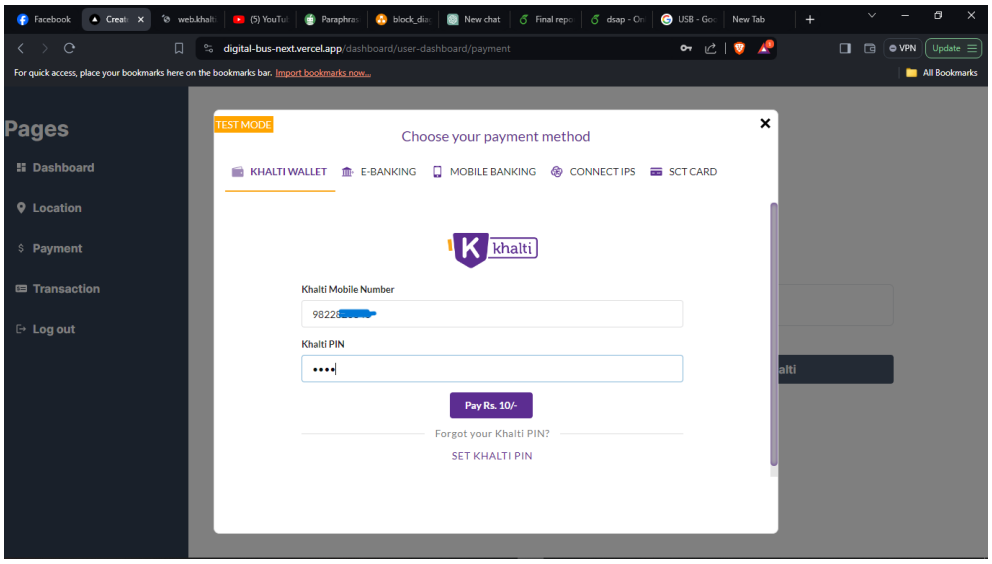
Lamachaur-16, Pokhara

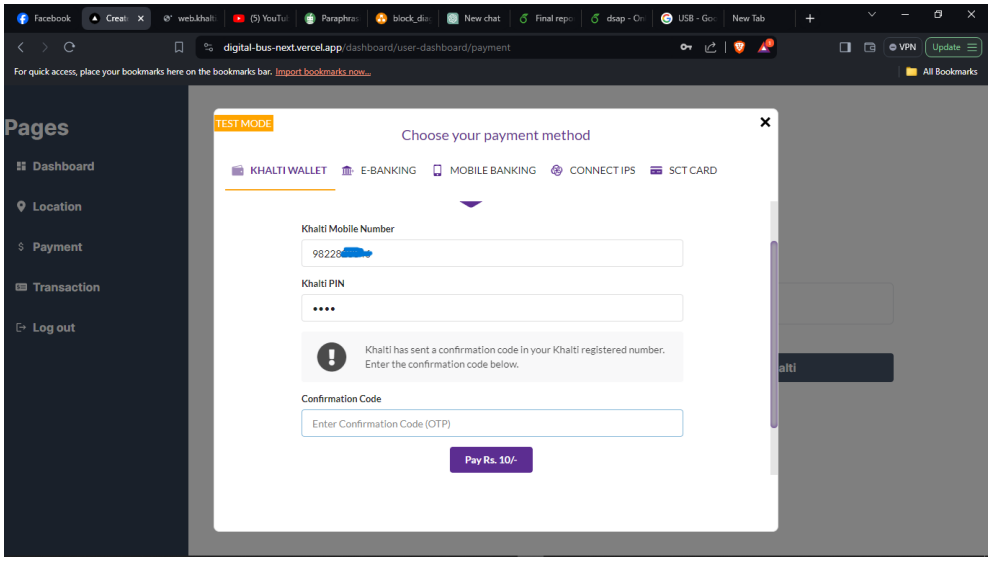
Western Regional Campus(WRC)

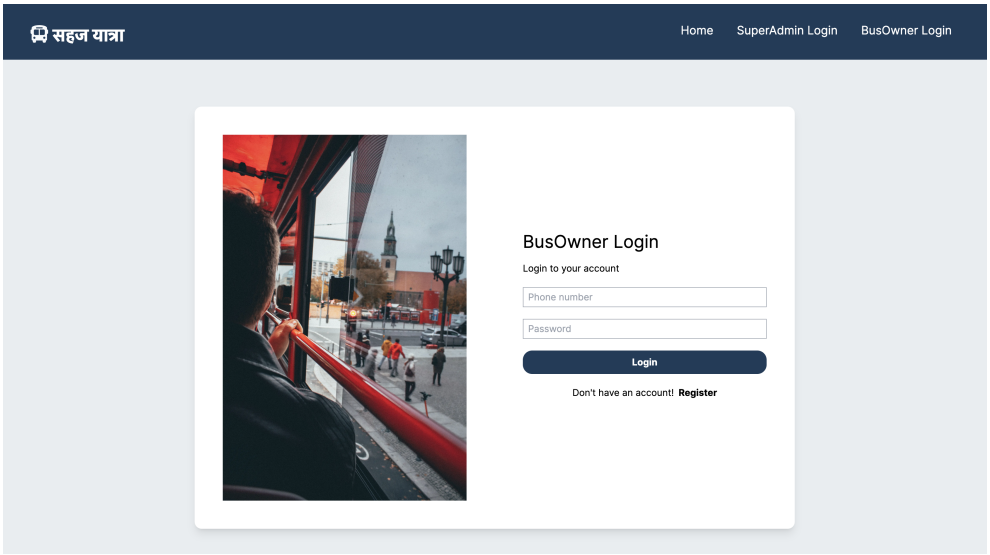
Members	Support	Company
Aayush	Pricing	About
Dikshyant	Documentation	Blog
Pramis	Guides	Jobs
Santosh	API Status	Press
		Careers

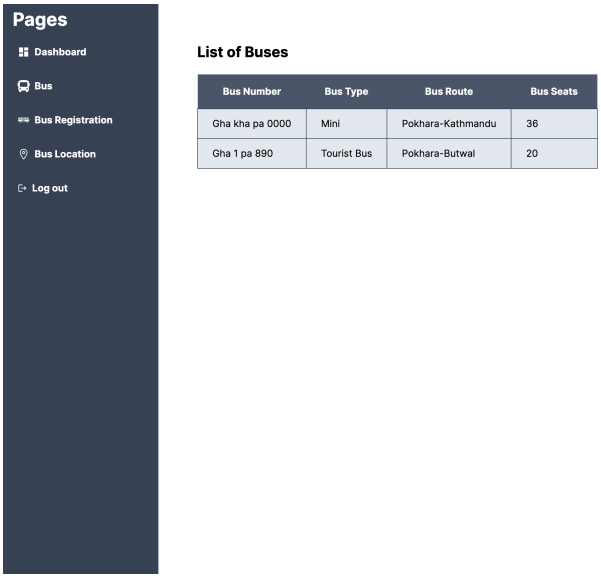


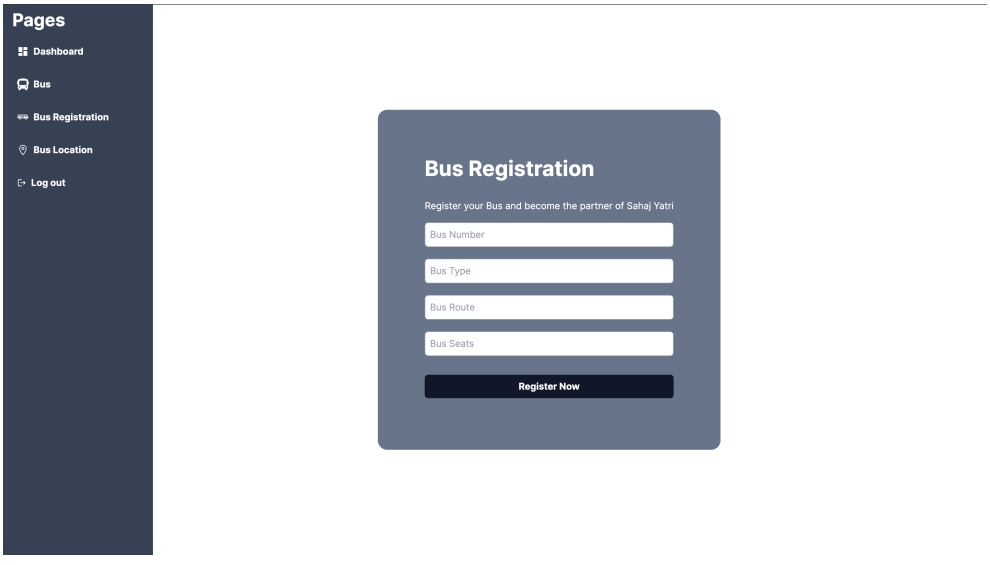

digital-bus-next.vercel.app/dashboard/user-dashboard/payment
Pages
Dashboard
Location
Payment
Transaction
Log out
Payment:
10
Pay with khalti


digital-bus-next.vercel.app/dashboard/user-dashboard/payment
Pages
Dashboard
Location
Payment
Transaction
Log out
TEST MODE
Choose your payment method
KHALTI WALLET
E-BANKING
MOBILE BANKING
CONNECT IPS
SCT CARD
khalti
Khalti Mobile Number
Khalti PIN
Pay Rs. 10/-
Forgot your Khalti PIN?
SET KHALTI PIN

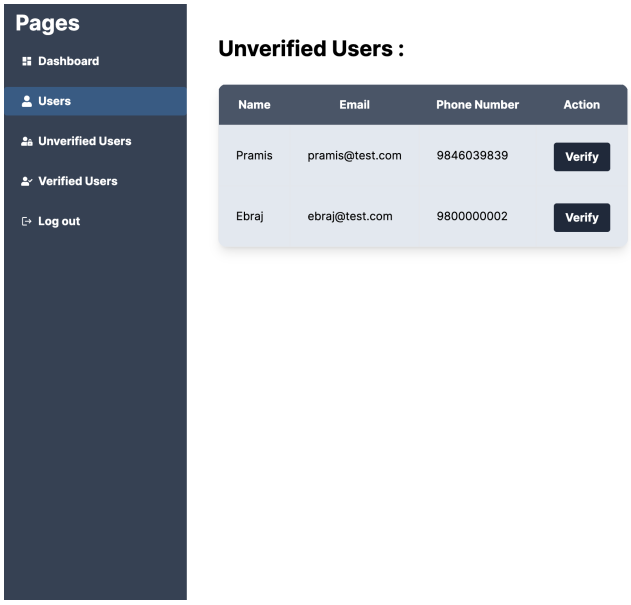

digital-bus-next.vercel.app/dashboard/user-dashboard/payment
Pages
Dashboard
Location
Payment
Transaction
Log out
TEST MODE
Choose your payment method
KHALTI WALLET
E-BANKING
MOBILE BANKING
CONNECT IPS
SCT CARD
Khalti Mobile Number
Khalti PIN
Khalti has sent a confirmation code in your Khalti registered number.
Enter the confirmation code below.
Confirmation Code
Enter Confirmation Code (OTP)
Pay Rs. 10/-

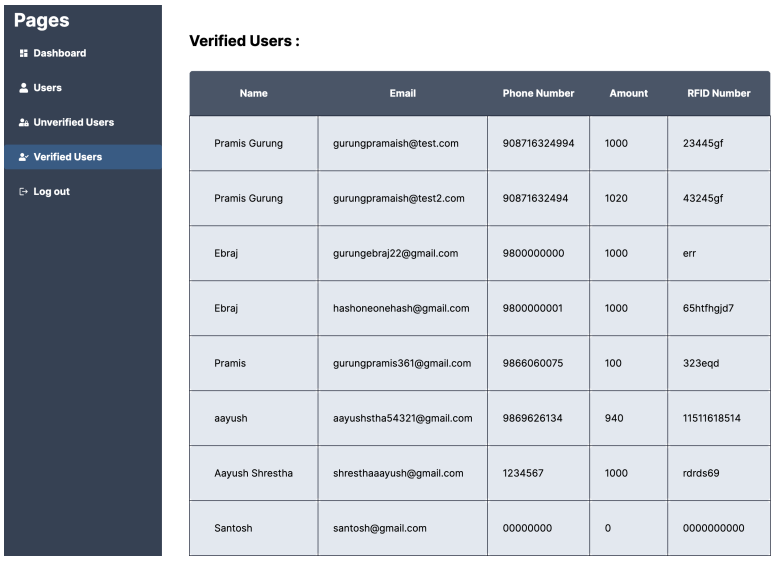

सहज यात्रा
Home
SuperAdmin Login
BusOwner Login
BusOwner Login
Login to your account
Phone number
Password
Login
Don't have an account! Register

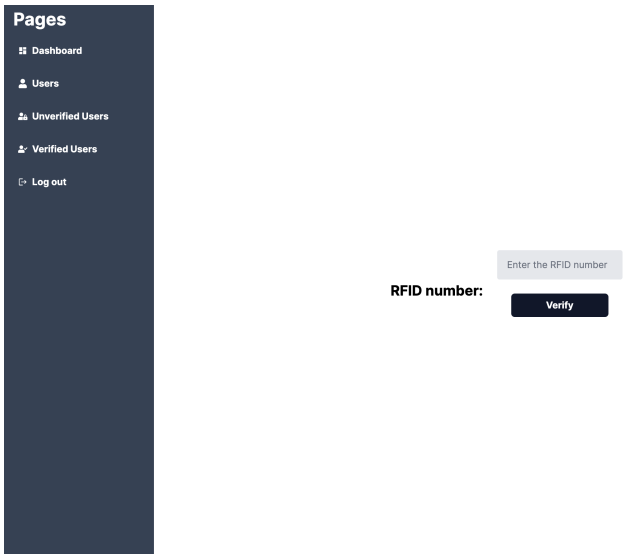

Pages
Dashboard
Bus
Bus Registration
Bus Location
Log out
List of Buses
Bus Number
Bus Type
Bus Route
Bus Seats
Gha kha pa 0000
Mini
Pokhara-Kathmandu
36
Gha 1 pa 890
Tourist Bus
Pokhara-Butwal
20


Pages
Dashboard
Bus
Bus Registration
Bus Location
Log out
Bus Registration
Register your Bus and become the partner of Sahaj Yatri
Bus Number
Bus Type
Bus Route
Bus Seats
Register Now


Pages
Dashboard
Users
Unverified Users
Verified Users
Log out
Users :
Name
Email
Phone Number
Citizenship Number
ID
Info
Pramis Gurung
gurungpramaish@test.com
908716324994
44-01-2-234a
65ba77272fcbd07b9c4b0841
Info
Pramis Gurung
gurungpramaish@test2.com
90871632494
44-01-2-234
65ba77342fcbd07b9c4b0847
Info
Ebraj
gurungebraj22@gmail.com
9800000000
1234567890
65c3cb3ed94f18bba7f6787b
Info
Ebraj
hashoneonehash@gmail.com
9800000001
1234567891
65c3cbf6d94f18bba7f6789c
Info
Pramis
gurungpramis361@gmail.com
9866060075
12345
65c799df4a8f63309b7b3c42
Info
aayush
aayushstha54321@gmail.com
9869626134
123141231
65c874424a8f63309b7b5225
Info
Aayush Shrestha
shresthaaayush@gmail.com
1234567
2987429423
65cdc6f8bec091430c484f20
Info
Santosh
santosh@gmail.com
00000000
739467345
65cef7ed285c088113d230c3
Info


Pages
Dashboard
Users
Unverified Users
Verified Users
Log out
Unverified Users :
Name
Email
Phone Number
Action
Pramis
pramis@test.com
9846039839
Verify
Ebraj
ebraj@test.com
9800000002
Verify


Pages
Dashboard
Users
Unverified Users
Verified Users
Log out
Verified Users :
Name
Email
Phone Number
Amount
RFID Number
Pramis Gurung
gurungpramaish@test.com
908716324994
1000
23445gf
Pramis Gurung
gurungpramaish@test2.com
90871632494
1020
43245gf
Ebraj
gurungebraj22@gmail.com
9800000000
1000
err
Ebraj
hashoneonehash@gmail.com
9800000001
1000
65htfhgjd7
Pramis
gurungpramis361@gmail.com
9866060075
100
323eqd
aayush
aayushstha54321@gmail.com
9869626134
940
11511618514
Aayush Shrestha
shresthaaayush@gmail.com
1234567
1000
rdrds69
Santosh
santosh@gmail.com
00000000
0
0000000000


Pages
Dashboard
Users
Unverified Users
Verified Users
Log out
RFID number:
Enter the RFID number
Verify

APPENDIX B: Source code

```
#include <ESP8266WiFi.h>
#include <SPI.h>
#include <MFRC522.h>
#include <SoftwareSerial.h>
#include <TinyGPS++.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClientSecure.h>


String previousLat, previousLon;
String bus_id = "65e0a3a445a16ee8944e186f";
String lat, lon;
String response;
const char* ssid = "Canada";
const char* password = "lordbro@555";
String Url = "https://sahaj-yatra-api.onrender.com/api/v1";


constexpr uint8_t RST_PIN = D3;     // Configurable, see
   typical pin layout above
constexpr uint8_t SS_PIN = D4;     // Configurable, see
   typical pin layout above

MFRC522 rfid(SS_PIN, RST_PIN); // Instance of the class
MFRC522::MIFARE_Key key;

SoftwareSerial gpsSerial(D1, D2);//rx,tx(D1, D2)
TinyGPSPlus gps; // create gps object

HTTPClient http;
WiFiClientSecure client;


void setup() {
  Serial.begin(115200);
  gpsSerial.begin(9600); // connect gps sensor
  SPI.begin(); // Init SPI bus
```

```
  rfid.PCD_Init(); // Init MFRC522
  WiFi.begin(ssid, password);
  pinMode(D8,OUTPUT);//buzzer
  pinMode(D0, OUTPUT);//led
  while (WiFi.status() != WL_CONNECTED) {
    digitalWrite(D0, HIGH);
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
  digitalWrite(D0, LOW);
  HTTPClient http;
  client.setInsecure();
}

void sendDataToServer(const String& serverUrl) {
    http.begin(client, serverUrl.c_str());

  int httpResponseCode = http.GET();

  if (httpResponseCode>0) {
    //Serial.print("HTTP Response code: ");
    // Serial.println(httpResponseCode);
    String payload = http.getString();

    Serial.println(payload);
     if (payload == "true") {
      Serial.println("Transaction successful!");
      digitalWrite(D0, HIGH);
      digitalWrite(D8, HIGH);
      delay(50);  // Turn LED on
      digitalWrite(D8, LOW);
      delay(40);
      digitalWrite(D8, HIGH);
      delay(100);  // Turn LED on
      digitalWrite(D8, LOW);
    } else if(payload == "false"){
      Serial.println("Transaction failed!");
```

```
      digitalWrite(D8, HIGH);
      delay(200);   // Turn LED on
      digitalWrite(D8, LOW);
      delay(40);
      digitalWrite(D8, HIGH);
      delay(200);   // Turn LED on
      digitalWrite(D8, LOW);
      delay(40);
      digitalWrite(D8, HIGH);
      delay(200);   // Turn LED on
      digitalWrite(D8, LOW);
    }
  }

  else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }

  // Free resources
  http.end();
}

bool sendgps = true;
static void smartdelay_gps(unsigned long ms)
{
  unsigned long start = millis();
  do
  {
    while (gpsSerial.available())
      gps.encode(gpsSerial.read());
  } while (millis() - start < ms);
}


void loop() {

  String tag;
```

```
// ----------------------send gps data every 5 seconds
   -----------------
  smartdelay_gps(1000);

  if (gps.location.isValid())
  {
    //Storing the Latitude. and Longitude
    lat = String(gps.location.lat(), 10);
    lon = String(gps.location.lng(), 10);

   // Send to Serial Monitor for Debugging
    Serial.print("LAT:  ");
    Serial.println(lat);  // float to x decimal places
    Serial.print("LONG: ");
    Serial.println(lon);
  }

  if (lat != previousLat || lon != previousLon) {
  String gpsData = Url + "/bus/location?busId=" + bus_id + "&
     latitude=" + lat + "&longitude=" + lon;
  sendDataToServer(gpsData);
  previousLat = lat;
  previousLon = lon;
  }
  // ---------if rfid available send rfid along with current
     gps location
  if ( ! rfid.PICC_IsNewCardPresent())
    return;
  if (rfid.PICC_ReadCardSerial()) {
  sendRfid_Gps(lat, lon);
  delay(20);
  }
  client.stop();
}


//function to send rfid and gps
void sendRfid_Gps(String lat, String lon){
```

```
    String tag;
    Serial.print("card detected: ");
    for (byte i = 0; i < 4; i++) {
      tag += rfid.uid.uidByte[i];
    }
    Serial.println(tag);
    rfid.PICC_HaltA();
    //send data to server
    String tagData = Url + "/user/deductfare?rfid=" + tag + "&
       busId=" + bus_id + "&latitude=" + lat + "&longitude=" +
       lon;
    sendDataToServer(tagData);
    Serial.println(tagData);
    delay(100);
    sendgps = true;
    tag ="";
    delay(2000);
    digitalWrite(D0, LOW);
}
```