International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Bus Location Tracking System

Ishwar Bagad[1], Samruddhi Pimpale [2], Pooja Patil[3], Jayshree Rane[4]

[1,2,3]Student, Electronics and Telecommunication, Alard College of Engineering and Managements Pune
[4]Professor, Electronics and Telecommunication, Alard College of Engineering And Managements Pune

Abstract
Many technologies have been developed around the world to make people's daily lives easier and better. All users can access Android, which is the newest and fastest-growing technology available today. End-user adoption has increased significantly during the past few years. Based on the most recent GPS technology, the effort gives the bus management team and improved means of tracking bus location in real time, managing the schedule, and monitoring bus functioning. The suggested solution makes use of a database and is an entirely integrated online bus tracking system. It offers the ability to track the whereabouts of that specific bus on Google Maps. They get at the bus on time and are able to view details about the bus, including the schedule. This bus position monitoring system can also be used as an accident detection alarm system, a soldier tracking system, and many more applications with a few small hardware and software modification. Tracking a car entails using latitude and longitude, or GPS coordinates , to determine its whereabouts. The GPS coordinates of a place are what make it valuable. This approach performs admirably in outdoor settings.
Projects involving this kind of vehicle monitoring system are widely uti-
lized to track school buses, taxis and other automobiles.

Keywords: Bus management, Neo6GPS and Buskaro The Raspberry

1. INTRODUCTION

In order to provide real-time information on bus timings and seat reservation, this project aims to develop a web application for tracking bus where about. For tracking the bus location, the online application makes use of Google Maps and Next.js. In order to communicate data to a server and display the bus location on the website, the project additionally makes use of a Raspberry Pi and a NEO 6 GPS module. The data is retrieved via an API.This project main goal is to offer a solution for the issue of erratic public transportation networks. By giving passengers accurate real-time information on bus times and location , this project seeks to improve the effectiveness and convenience of public transit. Users may simply track bus and make seat bookings online thanks to the user-friendly interface of the web application. The team employed a multistep technique to accomplish the project goals .The initial action was to determine the issue and look into potential fixes. The group then made the decision to track bus locations using Google Maps and construct the online application using Next.js. The group also determined that in order to send data to a server and retrieve it via an API, a Raspberry Pi and a NEO6 GPS module were required. The project was carried out effectively, and the outcomes demonstrated how useful the web application is for tracking bus locations and giving current bus timing information. The project also showed how to create a dependable and effective transportation system using contemporary technologies like Next.js, Google Maps, and Raspberry Pi, The sum up, this research offers a workable answer to the issue of erratic public transit

networks. This projects web application can greatly increase commuter's convenience and efficiency with public transportation by giving them access to real-time information on bus locations and schedules.

2. RELATED WORK

2.1 By tracking the bus's location around-the-clock, the proposed system seeks to reduce the amount of time that distant bus users must wait. All of the system's current data is kept on a server, which remote users can access using an online application. Although web-based solutions have long been the standard, normal bus passengers now find it relatively inconvenient to utilize them. Instead, they choose to use more portable Android apps because smartphones are so widely used. Because of this, it is strongly advised to develop an Android app version of the system, which will be considerably more practical and user-friendly for bus riders.

2.2 Real-time bus tracking and information can be developed into an Android app that passengers can download it to their cellphones and simply access while waiting for their bus. Additionally, by using an Android app, the system might benefit from smartphone capabilities like GPS tracking, push notifications, and real-time warnings, all of which would increase its efficacy and efficiency. In the end, the suggested system's Android app version will offer a convenient, approachable, and effective way to cut down on remote bus riders' wait times.

2.3 The suggested solution reduces the amount of manual labor required by the Bus-Management & Monitoring solution by utilizing artificial intelligence with the aid of an RFID module. When a machine passes the machine stop, RFID is used in this situation to track it. Because of this, just an approximate scenario based on the machine stopping is displayed rather than the precise state of the machine. This project was limited since accuracy is crucial in the environment we live in today.

2.4 This paper proposes a machine tracker operation to track a machine with a GPS transceiver. The goal of this effort is to create a system that uses a GPS tracking device and a shadowing device to manage and regulate the transport and provide an SMS with the vehicle's location and list.

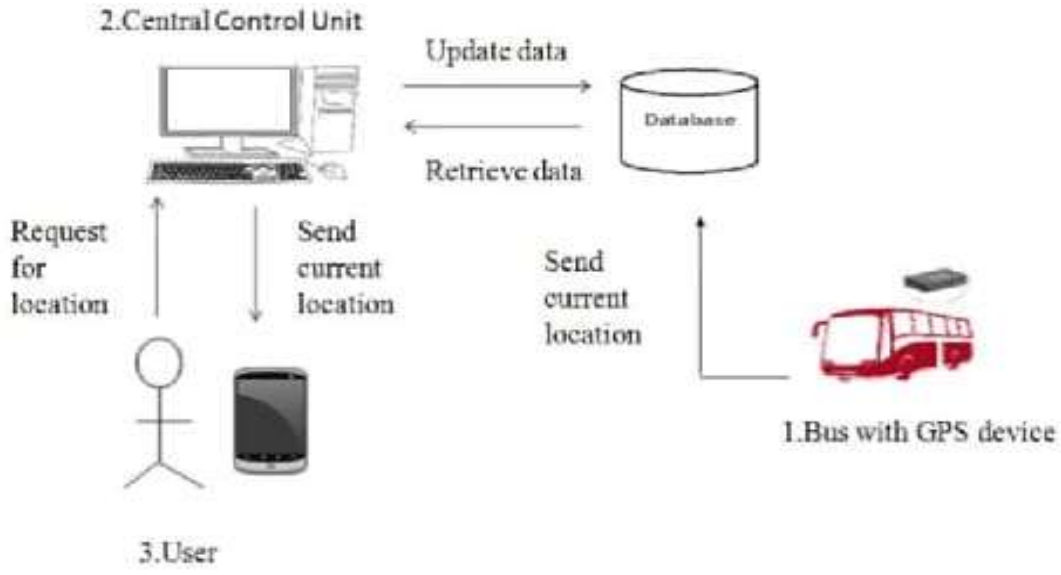


Figure 1: Block Diagram Of Proposed Work

3. HARDWARE PART

3.1 Raspberry Pi: Because of its affordability, portability, and compatibility with a range of operating systems, we decided to use the Raspberry Pi 4 as the primary piece of hardware for our project.

3.2 Neo6 GPS module: This module was chosen in order to track the bus's location in real time.

3.3 Power supply: The Raspberry Pi and GPS module were powered by a 5V/2.5A power source**.**

3.4 Display screen: The bus schedule and location were displayed on the website using a display screen. Furthermore, we ensured adherence to appropriate safety protocols during the handling of hardware components to prevent any possible harm or injury.

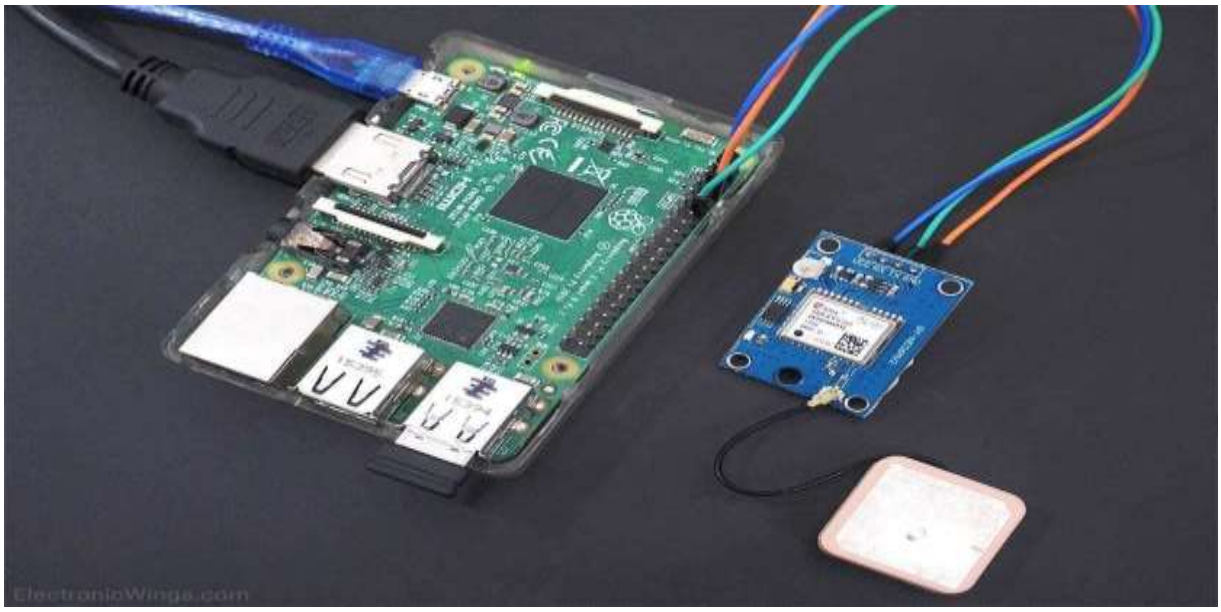


Figure 2 : Hardware Part

4.SOFTWARE PART

4.1 Website

4.1.1 **Next.js:** The web application was developed using the Next.js framework, which is a quick and effective framework that enables server-side rendering and simple deployment.

4.1.2 **Node.js:** Because Node.js is a scalable, lightweight, cross-platform JavaScript runtime environment that can manage a lot of data, we chose it to build the server-side application.

4.1.3 **Google Maps API:** Since Google Maps is a popular, well-documented API that offers precise and dependable location data, we used it to display the bus's location on the website.

4.1.4 **Payment gateway:** For online ticket booking and payments, we incorporated a dependable, safe, and user-friendly payment gateway. For every component, we also took into account alternative solutions such various GPS modules, payment gateways, and display panels. Nevertheless, we decided on the aforementioned elements for the project after giving the budget and project requirements considerable thought.

Furthermore, we made certain that the technologies chosen fulfilled the project specifications and were current. In order to guarantee the project's success, we also took care to adhere to the best practices and recommendations for each technology.

4.2 Android Application

An Android app can add Web View by generating a Web View object either programmatically in the code or in the layout file. Once implemented, developers can load web sites using methods like loadUrl() or show HTML content directly using loadData().In addition, Web View has a number of configuration options to allow you to alter its behavior, including handling URL redirects, enabling JavaScript, and configuring custom user-agent strings.

4.3 Testing

To make sure the web application was error-free, complied with project specifications, and was user-friendly, we carried out a number of testing methods. We tested the application's front-end and back-end using a variety of testing frameworks and tools.

4.3.1 Testing Units: We conducted unit tests of the web application's front-end components using Jest and Enzyme. Enzyme is a testing tool for React that offers a collection of practical APIs for checking the output of React components, while Jest is a well-liked JavaScript testing framework for testing React components.We were able to verify that each front-end component worked as planned by using unit testing, which increased the application's overall dependability.

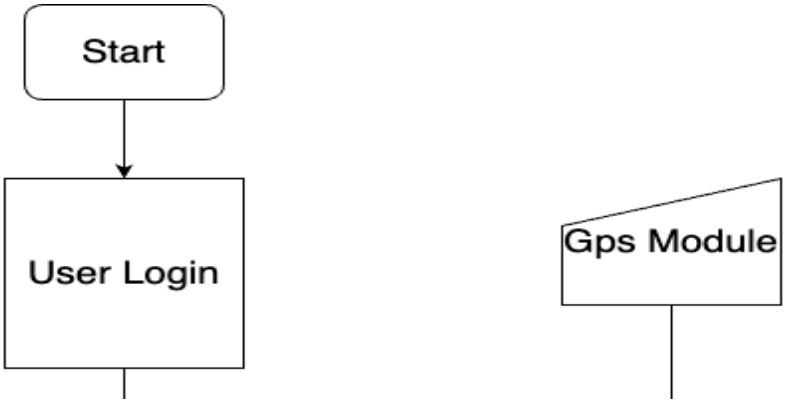
4.3.2 Integration Testing: The back-end elements of the web application were integrated tested using a Super test. Super test is a Node.js testing package that lets us simulate HTTP requests and responses to test endpoints and API responses. We were able to verify that the application's back-end components worked together as anticipated by doing integration testing.

4.3.3 User Acceptance Testing: To make sure the online application was user-friendly and satisfied end users' needs, we carried out user acceptance testing. In order to do user acceptability testing, users have to provide input on the application's overall usability, functionality, and user interface. We were able to find any functional or user interface problems that went unnoticed during the development and testing stage thanks to user acceptability testing.

All things considered, testing was an essential step in making sure the web application was stable, dependable, and easy to use. We were able to find and fix any problems or errors that surfaced during development by carrying out a variety of testing techniques, guaranteeing that the application complied with the project specifications and needs.

4.4 Deployments

Deploying the web application came next, following the completion of the development and testing stages. Moving the web application from a development environment to a production environment where people may use it is known as deployment.



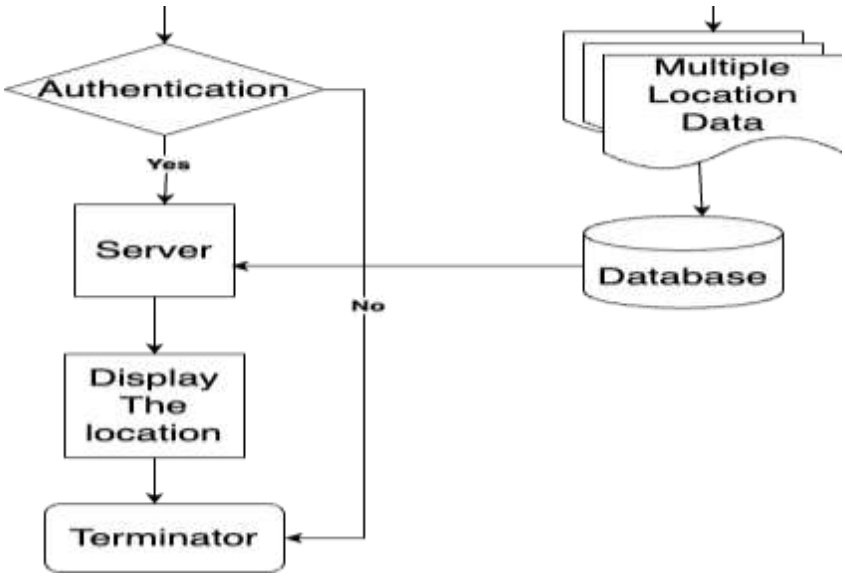


Figure 3: Flow Chart

The procedures we used to launch the web application are as follows:

4.4.1 Infrastructure Provisioning: To provide the required infrastructure for our web application, we turned to Amazon Web Services (AWS). We utilized Amazon Simple Storage Service (S3) to store static assets like images and videos, Amazon Elastic Load Balancer (ELB) to distribute inbound traffic to the web application, and Amazon Elastic Compute Cloud (EC2) to generate and launch virtual servers to execute the web application.

4.4.2 Creation and Deployment Pipeline: To automate the creation and deployment of our web application, we utilized AWS Code Pipeline. Code Pipeline constructed the application and deployed it to the virtual servers automatically, pulling the most recent code from our git repository**.** The continuous integration and continuous deployment (CI/CD) of our web application was implemented through the usage of AWS Code Build.

4.4.3 Configuration Management: To oversee the setup of our virtual servers, we employed Ansible. We can specify and control the infrastructure setup as code with the help of the open-source automation tool Ansible. Using Ansible roles and playbooks, we established the configuration of our virtual servers, which were deployed automatically.

4.4.4 Monitoring and Scaling: To keep an eye on our web application's availability and performance, we employed AWS Cloud Watch. We were able to identify and address any problems that came up during deployment because to the real-time metrics and log data that Cloud Watch supplied. In order to automatically scale up or down the virtual servers in response to the amount of traffic going into the web application, we also employed AWS Auto Scaling.

All things considered, deployment was an essential stage in guaranteeing that the online program was reachable and available to consumers. We were able to automate and streamline the deployment process by utilizing AWS services and tools, which decreased the possibility of errors and made sure the web application was operating effectively and consistently.

5. WORKING OF PROJECT

The project entails developing a web application that lets users read bus schedules, follow buses on map, and purchase tickets online. The system sends bus location data to a server using a Raspberry Pi and a

NEO-6 GPS module. Using an API, the web application retrieves the location data from the server and shows it on a Google Map.

First, cables are used to link the Raspberry Pi to the NEO-6 GPS module. The location data is read by the GPS module and sent as serial data to the Raspberry Pi. The Firebase Real-time Database is utilized to read the serial data and deliver it to the server via a Python script.

The web application can retrieve the location data from the server via an API, which is stored in the database. Next.js, a framework for creating server-side rendered React apps, is used to create the web application. It shows the bus timings using information from the server and plots the bus locations on a map using the Google Maps API.

Online ticket reservations with payment are another option available on the web app. Using JWT authentication, users can register for an account and log in. To store user data, including name, email address, and booking history, the web application employs MongoDB.

Firebase's Payment Gateway is used to create the payment system, enabling customers to safely purchase tickets online.

6. CONCLUSION AND FUTURE WORK

It has worked well for the system's development to send data from a Raspberry Pi to a centralized server using API calls and GPS-based location tracking. Many people can readily access the system due to its cross-device compatibility and user-friendly UI.The initiative has advanced the field of smart transportation systems by illuminating how data-driven technologies may be used to solve practical issues and enhance people's quality of life. The initiative has also brought to light the shortcomings of centralized systems and GPS-based location tracking, which may call for more investigation and study.

7. REFERENCES

1. 1."Application-Based Bus Tracking System," by Shubham Jain, Adarsh Trivedi, and Shweta Sharma, published in the 2019 International Conference on Machine Learning, Big Data, Cloud, and Parallel Computing (COMITCon), Faridabad, India.
2. "IoT based smart school bus monitoring and notification system," by Judy Hyperopia Raj and Jairam Sankar, was published at the IEEE Region 10 Humanitarian Technology Conference (R10-HTC) in 2017.
3. "Transit Bus Electrification Evaluation from GPS Speed Traces," by Andrew J. Kotz, Eric Miller, Andrea Watson, and Kenneth J. Kelly, published in the 2020 IEEE Transportation Electrification Conference & Expo (ITEC)
4. "GPS Based Vehicle Tracking System," by Mohd Hakimi Bin Zohari and Mohd Fiqri Bin Mohd Nazri, published in Volume 10 of the 2021 International Journal of Scientific & Technology Research
5. "Real Time Bus Tracking and Location Updation System," by Ms. A. Deepika Shree, Ms. J. Anusuya, and Dr. S. Malathy, published in the 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)
6. "Global Positioning System" Signals, Measurements, and Performance, Pratap Misra, Per Enge, Ganga-Jamuna Press, Second Edition.