SHUTTLESYNC: REAL-TIME BUS TRACKING AND NAVIGATION

Anushri Jyothi Bhasu[1], K Muhammed Fazil[1], Manas Krishna Sumathi Raghu[1], Leya M S[2]

[1]*Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning)*
[2] *Asst.Prof /Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning)*
Nehru College of Engineering and Research Centre, Pampady, Thrissur, Kerala, India

---

Abstract – ShuttleSync is an innovative, real-time bus tracking and navigation system specifically developed to streamline college transportation services. This system is a browser-based college bus tracking system that delivers real-time GPS monitoring, dynamic ETA estimation, and automated passenger management through role-specific portals for users, drivers, and administrators. Integrated with a Convolutional Neural Network-powered audio classification module, the system autonomously detects on-board emergencies and dispatches instant SOS alerts, providing a cost-effective, hardware-independent intelligent campus transit solution.

Keywords: **GPS (Global Positioning System), campus transportation system, ETA (Estimated Time of Arrival), Convolutional Neural Network, SOS alerts.**

1. INTRODUCTION

An efficient transportation system plays an important role in the management of campus transportation. Most educational institutions use shuttle transportation to enable students to move around the institution and access various learning facilities, such as academic buildings and hostels. Nevertheless, the conventional shuttle transportation system has been challenged by the inability to offer real-time tracking, which enables students to know the exact location of the bus or the expected time of arrival. This has resulted in longer waiting times, which might be inconvenient for students.

This challenge has been addressed by the development of ShuttleSync, which is a real-time bus tracking and navigation system intended to enhance the efficiency of campus transportation. The system has been developed and integrated with a browser-based application that delivers real-time GPS Monitoring, dynamic ETA estimation, and automated passenger management through role-specific portals for users, drivers, and administrators. The system has been developed using Python and the Flask framework, integrated with a Convolutional Neural Network-powered audio classification module for autonomous detection of on-board emergencies and to dispatch instant SOS alerts, providing a cost-effective, reliable, and hardware-independent intelligent campus transportation system.

2. LITERATURE REVIEW

A design of an Intelligent transportation system that makes use of image processing and IoT technologies to automate passenger counting and bus tracking is discussed by Pown M., Bhuvana B. P., Dhilipan R., Ganesh A., and Vicraman S. V [1]. In this system, a Raspberry Pi 4B is utilized as a core processor with a camera module to capture images from a video stream. Using OpenCV and a MobileNet SSD deep learning model, it can count passengers through an object detection algorithm. Individuals are tracked with a centroid tracking method that assigns each person a unique ID to avoid duplication of data. To capture vehicle location data, a NEO-6M GPS module is utilized to send data to Firebase through GSM communication. A Flutter-based application is utilized to send processed data to drivers and administrators. For testing purposes, a local Flask server was utilized. Such a system can improve automation and provide real-time visibility of data.

P. T. Kalaivaani, P. Karthikaa, M. Rathisri, and C. Thameena in [2] proposed a bus tracking and passenger information system using GPS and QR technologies. The system architecture is based on using GPS for tracking, proximity sensors for seat occupancy detection, and QR code integration for quicker access to real-time data for passengers. When a passenger scans a QR code at bus stops or inside buses, they are immediately provided with real-time data about the bus's location, estimated time of arrival, and delay messages. The system is based on Firebase and MySQL for backend processing, and GSM/3G/4G technologies are used for communication between hardware and a cloud database. The system is found to update data in 3-5 seconds, which is a minimum time for such a system to function reliably. This system is effective in providing better passenger experience by providing accurate data for passengers and increasing transparency in bus operations.

Ishwar Bagad, Samruddhi Pimpale, Pooja Patil, and Jayshree Rane [3] propose the development of a highly scalable bus location tracking and online ticketing system. The hardware components include the Raspberry Pi 4 and GPS, which are used to obtain the location of the vehicle. The cloud infrastructure uses AWS cloud services, which include EC2 for hosting, S3 for data storage, and Elastic Load Balancing, which helps handle high traffic volumes. The software

components include Next.js for the user interface and Node.js for the backend operations. MySQL and MongoDB are used for data management of the routes and the users. The integration and deployment process uses AWS CodePipeline, AWS CodeBuild, and Ansible. The system supports the real-time tracking and ticketing of vehicles with the use of JWT authentication. This study presents a highly scalable system with the capability to operate smoothly with high user volumes, providing an efficient solution for smart city transportation systems.

Bhanudas Gadade, A. O. Mulani, and A. D. Harale [4] have proposed an IoT-based school bus monitoring system focusing on the safety of students and their attendance monitoring. This system makes use of an ESP32 module, GPS, RFID, and a Real-Time Clock module to track the movement of the vehicle and the activities of students entering or leaving the vehicle. Each student has an RFID card, and their attendance is recorded automatically by the system. This system makes use of the Internet of Things, enabling the monitoring of the vehicle in real-time and sending messages to the authorities in case of route deviations or delay.

Kalyani Akhade, Krishna Dadge, Rutuja Nikam, Dipti Satpute, and Kunal Shinde [5] proposed a mobile app designed for precise GPS-based location tracking to minimize waiting times for commuters. BUSNAV, developed using Kotlin for the frontend and Java for backend services, utilizes the Google Maps API to provide an interactive user interface for users. Firebase has been employed for the real-time synchronization of location data between the driver and passenger applications. Users can view the route of the buses, the estimated time of arrival, and delay information, allowing them to utilize their time efficiently. BUSNAV has been successfully tested, demonstrating low lag and efficient rendering of the map. The advantage of this system can be identified in its simplicity, scalability, and usability, allowing it to be implemented in institutional campuses or urban transport systems. Real-time location data can improve user convenience, punctuality, and transparency.

Siti Hanisah Lakman and Nur Ziadah Harun [6] proposed a secure Android-based bus scheduling and access control system using biometric fingerprint authentication. The proposed system makes efficient use of Firebase Realtime Database to store and synchronize data, thus facilitating quick access to bus schedules and route information. Moreover, the proposed system replaces password authentication with a fingerprint-based login mechanism to reduce user error and increase security. An OTP-based verification mechanism is also provided to increase security and prevent unauthorized access. The proposed system is user-friendly and is specifically designed to work on Android-based operating systems. Although it is limited to a specific operating system, it highlights the significance of integrating cybersecurity practices into transportation management systems.

T. W. M. P. Premadasa, D. G. S. M. Karunathilake, and W. A. S. Wickramasinghe [7] presented a bus tracking and arrival prediction system that enhances the accuracy of estimated travel time using a combination of routing algorithms and data compression. Continuous location data is stored using a MySQL database. Dijkstra's algorithm is used for calculating the shortest path, and Map Matching algorithms for determining the optimal routes. In addition, the Google Distance Matrix API is used for providing accurate ETA updates. To reduce bandwidth usage, the Google Encoded Polyline Algorithm is used for data compression. The proposed solution is efficient for providing accurate ETA calculations for real-world scenarios and enhancing the accuracy of the bus arrival information provided to the passengers.

An innovative IoT-based application for tracking buses using High Frequency Radio Frequency Identification (HF RFID) technology instead of GPS has been presented by Anjali Jain and Agya Mishra [8]. In this application, RFID tags are fixed on each bus, and RFID readers are fixed at major stops to track the arrival and departure of buses. The data is then received by the NodeMCU ESP8266 microcontroller and sent to the Blynk IoT cloud platform, which can be accessed using mobile devices. The system has a low data latency of around 161 milliseconds, making it a rapid and accurate system. The RFID system does not have GPS inaccuracies and is an energy- efficient system as it consumes low power.

Krishnamoorthy, V. Vijayarajan, G. Sivashanmugam, and N. Visakeswaran, in their work [9], developed an IoT-based bus tracking system utilizing the LinkIt ONE board, which incorporates GPS, GPRS, and GSM technologies. The system allows real-time tracking of the location of the vehicle by transmitting the data to the cloud environment, namely Firebase and Google IoT Cloud, and the data can be easily accessed in real time. A React Native app has been developed, which uses Google Maps to track the current location of the bus. This system enables real-time tracking of the location of the vehicle with zero or little delay and with high precision, thereby making the best use of the capabilities of IoT technology.

K. Ramya, J. Asha Jency, and R. Anu Mangai [10] proposed a GPS and GSM-based tracking system that is especially designed for college bus management systems. In this system, an Android app is placed on every bus that communicates with a server using GPS technology. Once a bus reaches a particular stop, students waiting at the next stop are sent automatic SMS messages regarding the arrival of the bus. In addition to this, route scheduling is also possible in this app. Cloud-based databases are also utilized in this app to provide maximum accuracy and reliability. This app is

successfully implemented in a college environment and is beneficial in increasing the punctuality of the bus.

3. METHODOLOGY

The system is accessible through any browser on a Local Area Network and supports three distinct roles called User, Driver, and Administrator, each of whom interacts with a dedicated portal tailored to their specific needs. Access to the system begins with an authentication step where the user logs in with their credentials, after which the system identifies their role and directs them to the appropriate interface.

The Driver Portal is used by shuttle operators to manage their daily routes. Once logged in, the driver is able to start and end routes, share their live location throughout the journey, and record the boarding and alighting of students at each stop. The driver's real-time GPS position is continuously shared with the system so that all other users can track the bus as it moves along the route. The driver's interface also includes an SOS feature that monitors the surroundings and allows for immediate emergency alerts to be raised when required.

Students can access the User Portal to stay informed about their bus. The portal displays the live location of the assigned bus on an interactive map, provides an estimated time of arrival at the student's boarding stop, and shows the current boarding status. This allows students to plan their time effectively without needing to wait at the stop unnecessarily, as they can track exactly where the bus is at any given moment.

The Admin Portal is intended for institutional coordinators who oversee the entire fleet. It provides a consolidated view of all buses, their operational statuses, and current locations. Administrators can manage driver and student records, monitor emergency alerts as they arise, dismiss resolved alerts, and review historical route statistics and trip performance data. This centralised oversight ensures that any operational or safety issue can be identified and addressed promptly.

When an emergency is detected, either automatically through the audio monitoring system on the driver's device or manually by the driver pressing the SOS button, an alert is immediately raised and made visible to the administrator.

The alert contains details about the bus, its location at the time of the incident, and the nature of the emergency. The administrator can then take appropriate action and mark the alert as resolved once the situation has been addressed. This end-to-end flow from user login to real-time tracking and emergency response forms the operational backbone of the ShuttleSync system.

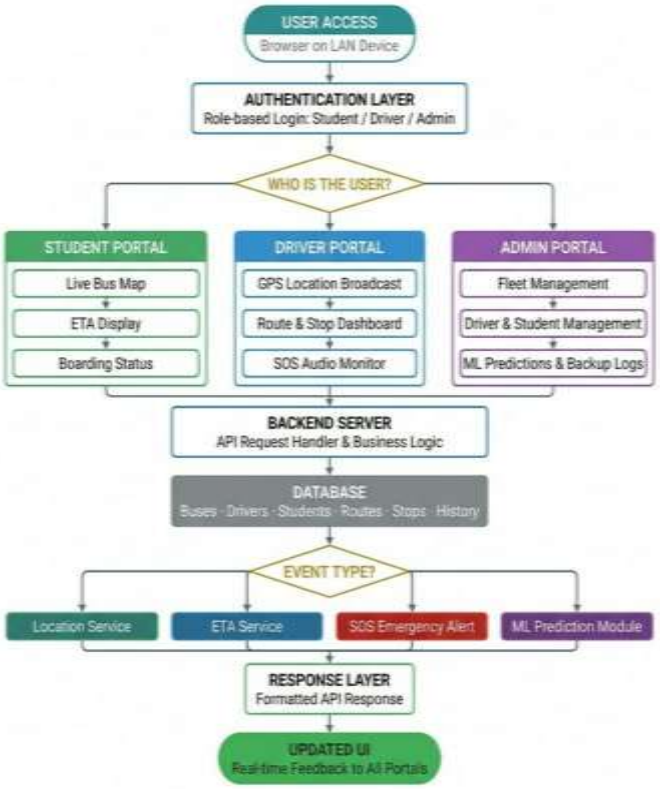

Fig. 1. System Architecture

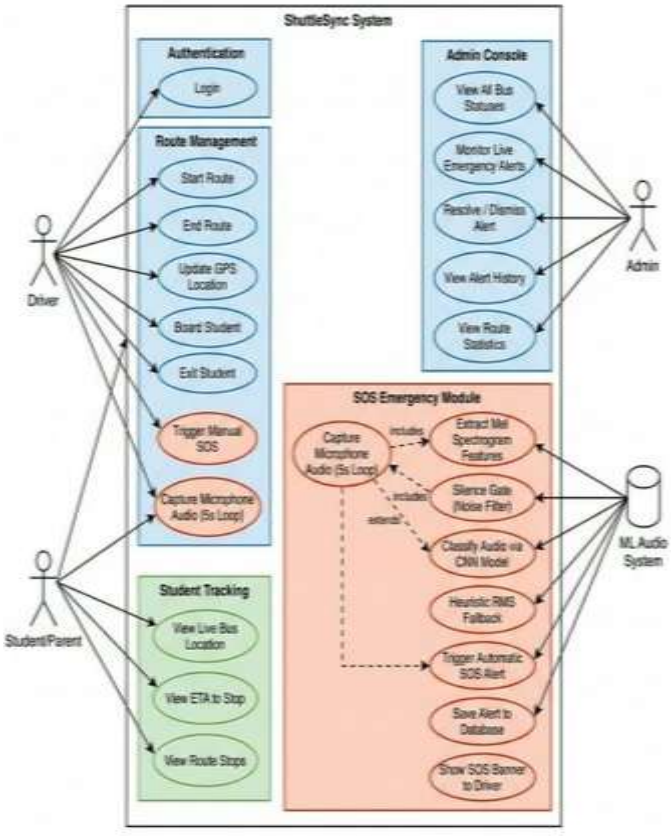

Fig. 2. Use Case Diagram

Fig. 3. ER Diagram of Shuttlesync

4. IMPLEMENTATION

ShuttleSync is implemented as a web-based bus tracking and safety management system built on the Flask microframework in Python, with SQLite as the relational database and a Convolutional Neural Network for on-board emergency audio detection. The backend is structured into three role-based REST API Blueprints, namely Driver, User, and Administrator, registered under a single Flask application entry point. Client interfaces serve as Jinja2-rendered HTML templates integrated with the Google Maps Platform for real-time geospatial visualisation.

The Driver Console is the operational core of the system. After authenticating with a unique driver code and password, the driver initiates a trip via the 'start_route' endpoint, which creates a trip record in the database and marks the corresponding bus as active. The driver's device browser continuously captures GPS coordinates using the Geolocation API and transmits them to 'update_location', where each reading, along with speed and timestamp, persisted to the 'bus_locations' table. Boarding and alighting of users are logged against the active trip through dedicated endpoints, enabling live occupancy tracking. When the route concludes, the driver calls 'end_routes', which finalises the trip and archives it to the 'travel_history' table for future analytics and ETA refinement.

The User Console allows passengers to monitor their assigned bus in real time. Upon login, the user's bus, route, and boarding stop details are fetched from the database. The interface periodically polls 'bus_location' to update the bus marker on the map. ETA is calculated by the 'eta_service' module using the Haversine formula applied iteratively across ordered route stops. When the bus is active, ETA is computed dynamically from the bus's live position to the student's stop and to the destination. When the bus has not yet departed, a scheduled ETA is estimated from historical average speeds in the 'travel_history' table, segmented by day of week and direction, falling back to 30 km/h when insufficient history is available. The user interface also surfaces the bus's emergency status flag, displaying an alert if an SOS event is active for that bus.

The Admin Console presents the real-time location and operational state of all buses, route-level analytics including stop-wise passenger load and weekly trip summaries, and a complete history of emergency events. Administrators can view all unresolved SOS alert type, GPS coordinates, confidence score, and timestamp, and mark them as resolved, which clears the record and resets the bus emergency flag. The SOS system operates through a dual-mode pipeline. In the automated mode, the Driver Console captures five-second audio clips from the device microphone and posts them to 'audio_check'. Each clip is converted to a 128×128 Mel spectrogram using librosa at 22,050 Hz, normalised, and passed into a pre-trained CNN with three convolutional blocks, Batch Normalisation, Max Pooling, a 256-unit Dense layer with Dropout, and a five-class Softmax output classifying audio as normal, engine noise, glass break, crash, or scream. A prediction triggers an emergency only when model confidence reaches or exceeds 0.80 for an emergency class. A 30-second cooldown per bus prevents alert flooding, after which the clip is archived, the current GPS coordinates are retrieved, and an alert record is inserted into 'emergency_alerts' with the bus emergency flag set to 1. In the manual mode, the driver presses a dedicated SOS button that bypasses the CNN and triggers the same pipeline immediately, ensuring coverage for incidents not detectable through audio alone.

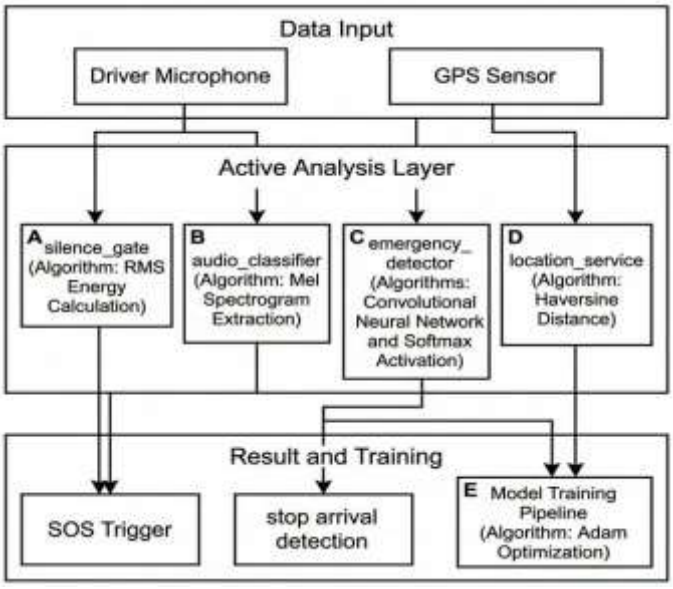

Fig. 4. Algorithm Mapping

5. RESULTS AND DISCUSSION

The ShuttleSync system was evaluated across three primary functional modules, namely, real-time bus tracking, ETA computation, and emergency alert detection, and demonstrated satisfactory performance in each domain under institutional operating conditions. With a lightweight backend and database design, ShuttleSync can provide faster and more private data tracking experience. Based on a comparison of ShuttleSync with most of the cloud-based bus tracking systems, it can provide a more reliable operation, especially in low-bandwidth environments, thus being suitable for a campus environment.

Fig. 5. Shuttlesync Home Page

Fig. 6. User Live bus Tracking and ETA

The live bus map accurately displayed the bus position as it progressed along the designated route, and the boarding and alighting records maintained by the driver were registered against each trip in the database.

Fig. 7. Driver Console

The real-time tracking module performed consistently throughout testing, with GPS location updates from the driver's device being reflected across the student and administrator portals with minimal perceptible delay.

Fig. 8. Fleet Map in Admin Console

Fig. 9. SOS detected by Audio in the driver console

Fig. 10. SOS alert sent to the Admin with Bus Details

The model effectively suppressed false positives arising from routine ambient noise on board the bus, primarily through the silence gate and the confidence gating mechanism. Emergency events that exceeded the threshold were correctly elevated to SOS alerts, with corresponding records created in the database and alert notifications surfaced on the administrator dashboard. The manual SOS trigger also functioned as intended, providing an immediate fallback pathway for situations not captured by the audio monitor.

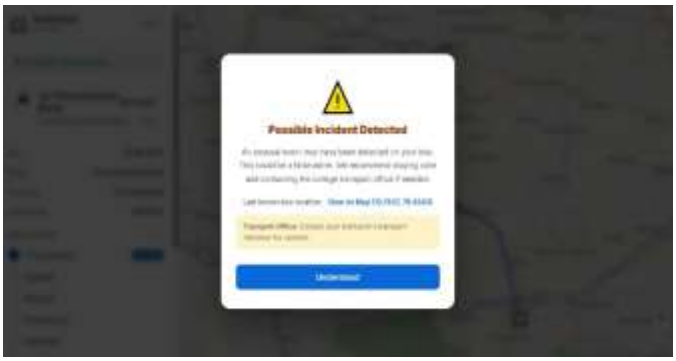

Fig. 11. SOS alert sent to the User

The CNN model was trained on a labelled audio dataset over fifteen epochs with a batch size of thirty-two, an 80% confidence threshold, and a 20% validation split using the Adam optimizer. Future work may focus on incorporating server-side push notifications, expanding the audio training dataset to improve classification robustness, and conducting field trials across a larger fleet to validate performance at scale.

6. CONCLUSION

ShuttleSync is a web-based institutional bus tracking and safety management system that integrates real-time GPS monitoring, data-driven ETA prediction, and machine learning-based emergency detection within a unified platform. The system successfully addresses the limitations of conventional shuttle management approaches by providing role-specific interfaces for drivers, users, and administrators. The incorporation of a CNN for audio-based emergency classification represents a meaningful contribution to passenger safety, enabling automatic detection of events such as crashes and screams without requiring manual intervention from the driver.

The dual-mode SOS mechanism, combining automated audio analysis with a manual trigger, ensures comprehensive emergency coverage across a range of on-board scenarios.

The system demonstrated reliable performance across its core functional modules during testing and offers a scalable foundation for future enhancements like the integration of real-time push notification services, expansion of the audio training dataset to improve model generalisation, and deployment across larger institutional fleets for field-level validation.

ACKNOWLEDGEMENT

We thank the faculty members of the Department of Computer Science and Engineering (AI & ML) for their suggestions and technical assistance during the course of this work. We extend our appreciation to the management of our institution for providing the necessary facilities and resources to complete this project successfully.

REFERENCES

[1] P. M., B. B. P., D. R., G. A., and V. S. V., “Advanced Transportation System Using Raspberry Pi, Open CV and Flutter,” ICDSBS, Chennai, India, 2025, pp. 1–6, doi: 10.1109/ICDSBS63635.2025.11031840.

[2] P. T. Kalaivaani, P. Karthikaa, M. Rathisri, and C. Thameena, “Smart GPS Based Bus Tracking System with Real-Time Updates Using QR Code,” International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), vol. 5, no. 3, pp. 743–746, May 2025, doi: 10.48175/IJARSCT-26393.

[3] I. Bagad, S. Pimpale, P. Patil, and J. Rane, “Bus Location Tracking System,” International Journal for Multidisciplinary Research (IJFMR), vol. 6, no. 2, pp. 1–7, Mar.–Apr. 2024, doi: 10.36948/ijfmr.2024.v06i02.18196.

[4] B. Gadade, A. O. Mulani, and A. D. Harale, “IoT Based Smart School Bus and Student Monitoring System,” Naturalista Campano, vol. 28, no. 1, 2024, E-ISSN: 1827-7160.

[5] K. Akhade, K. Dadge, R. Nikam, D. Satpute, and K. Shinde, “BUSNAV: A Real-Time Bus Tracking Application,” IJISRT, vol. 9, no. 4, pp. 1932–1935, Apr. 2024, doi: 10.38124/ijisrt/IJISRT24APR1696.

[6] S. H. Lakman and N. Z. Harun, “A Bus Schedule Application Using Fingerprint,” AITCS, vol. 5, no. 1, pp. 38–47, 2024.

[7] T. W. M. P. Premadasa, D. G. S. M. Karunathilake, W. A. S. Wickramasinghe, K. G. H. Abeywardhana, A. Athulathmudali, K. W. R. C. W. A. M. G. S. Koswatta, M. H. H. Mohamed, D. C. M. Senavirathna, and P. Abeygunawardhana, “Bus Tracking and Arrival Prediction System,” IJISRT, vol. 8, no. 11, pp. 1855–1862, Nov. 2023, ISSN: 2456-2165.

[8] A. Jain and A. Mishra, “Design of IoT Based Real-Time Bus Tracking App Using HF-RFID,” IJRTE, vol. 9, no. 6, pp. 227–231, Mar. 2021, ISSN: 2277-3878.

[9] A. Krishnamoorthy, V. Vijayarajan, G. Sivashanmugam, and N. Visakeswaran, “Real Time Bus Tracking System Using LinkIt One,” IJRTE, vol. 7, no. 5S3, pp. 16–20, Feb. 2019, ISSN: 2277-3878.

[10] K. Ramya, R. Anu Mangai, and J. Asha Jency, “Real Time E-City Bus Tracking System,” IJETIE, vol. 5, no. 5, pp. 250–255, May 2019, ISSN: 2394-6598.A