SMARTBUS: AN AI-ENABLED GPS-BASED PASSENGER INFORMATION AND TRACKING SYSTEM FOR GOVERNMENT TRANSPORT

A PROJECT REPORT

Submitted by

SHYAM S
MOHAMMED REHAN SHARIEF M T
SATHISH S

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2026

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this project report " **SMARTBUS: AN AI-ENABLED GPS-BASED PASSENGER INFORMATION AND TRACKING SYSTEM FOR GOVERNMENT TRANSPORT"** is the bonafide work of **"SHYAM S, MOHAMMED REHAN SHARIEF M T, SATHISH S"** who carried out the project work under my supervision.

SIGNATURE
Dr. S. Senthil Pandi., M.E., Ph.D.,
Associate Professor
Department of Computer Science
and Engineering,
Rajalakshmi Engineering College,
Chennai - 602 105.

Submitted to Project Viva-Voce Examination held on ____________________

Internal Examiner **External Examiner**

ABSTRACT

The SmartBus project aims to revolutionize public transportation for government-operated bus systems by introducing a real-time GPS-based passenger information and tracking system. Existing public transport systems often lack transparency, provide limited updates, and leave passengers unaware of critical information such as seat availability, accurate arrival times, and emergency support. SmartBus addresses these shortcomings by integrating GPS technology, IoT-based seat occupancy sensors, AI-powered ETA prediction models, and a user-friendly mobile application to ensure safety, convenience, and reliability for passengers.

The system architecture comprises a multi-layered solution involving GPS devices on buses, sensor modules for seat monitoring, cloud-hosted backend services, mobile apps for passengers and drivers, and an administrative dashboard for fleet supervision. Machine learning algorithms are employed to predict ETAs dynamically based on real-time traffic data and historical movement patterns, significantly improving the reliability of arrival estimates. Moreover, the inclusion of emergency panic buttons in the mobile app provides a vital safety feature, allowing passengers to alert authorities instantly during distress situations.

SmartBus is designed to be scalable, cost-efficient, and operational even in low-connectivity zones through SMS/GSM-based fallback systems. This paper presents the design, development, and deployment considerations for SmartBus, alongside its potential impact on public transport modernization in both urban and rural settings.

ACKNOWLEDGMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Dr. S. SENTHIL PANDI , M.E., Ph.D.,** Assistant Professor of the Department of Computer Science and Engineering. Rajalakshmi Engineering College for his valuable guidance throughout the course of the project.

SHYAM S	**2116220701508**
MOHAMMED REHAN SHARIEF	**2116220701515**
SATHISH S	**2116220701526**

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

S. No	ABBR	Expansion
1	AI	Artificial Intelligence
2`	API	Application Programming Interface
3	AJAX	Asynchronous JavaScript and XML
4	ASGI	Asynchronous Server Gateway Interface
5	AWT	Abstract Window Toolkit
6	BC	Block Chain
7	CSS	Cascading Style Sheet
8	DFD	Data Flow Diagram
9	DSS	Digital Signature Scheme
10	GB	Gradient Boosting
11	JSON	JavaScript Object Notation
12	ML	Machine Learning
13	RF	Random Forest
14	SQL	Structure Query Language
15	SVM	Support Vector Machine

1. INTRODUCTION

Public transportation is the backbone of urban and semi-urban infrastructure worldwide, providing affordable and accessible mobility solutions to millions. In many countries, especially in developing regions, government-operated buses constitute a significant portion of daily commuter transport. However, one of the persistent challenges in public transport systems is **uncertainty**—passengers often lack real-time information about bus arrivals, seating availability, route changes, and delays.

The absence of timely, accurate information leads to longer waiting times, overcrowding, dissatisfaction, and ultimately, the decline of public transport usage. Studies have shown that **passenger satisfaction** and **trust** are closely linked to the availability of real-time service information (Williams et al., 2020).

The implementation of a **GPS-based tracking system** integrated with **seat availability detection**, **emergency support features**, and **dynamic routing algorithms** can significantly enhance the commuter experience. By offering clarity, convenience, and safety, such systems empower users to plan their journeys efficiently, reduce wait times, and make informed decisions — contributing to higher adoption rates of public transport.

This literature review aims to explore the evolution of bus tracking systems, review recent technological advancements, compare global efforts, identify existing gaps, and justify the need for a comprehensive, AI-powered SmartBus system for government-operated transport fleets.

1.1 GENERAL

Public transportation is a vital component of urban infrastructure, providing essential mobility services to millions of people every day. However, despite its importance, many government-operated bus systems continue to face significant challenges such as lack of transparency, unpredictable bus arrival times, absence of real-time passenger information, overcrowding, and minimal safety measures. These challenges lead to inefficiencies, reduced passenger satisfaction, and poor adoption of public transport over private means of travel.

The emergence of technologies such as GPS, Internet of Things (IoT), cloud computing, and Artificial Intelligence (AI) has paved the way for transforming traditional transport systems into intelligent mobility solutions. With the proper integration of these technologies, public transportation systems can become more efficient, safer, and commuter-friendly.

The SmartBus project is an initiative aimed at modernizing government-operated bus services by implementing a real-time, AI-powered bus tracking and passenger information system. It integrates multiple components such as real-time GPS tracking, AI-based Estimated Time of Arrival (ETA) prediction, seat availability monitoring using sensors, and emergency alert mechanisms, all accessible via a mobile application. This system ensures better planning, convenience, safety, and communication between commuters and transport authorities.

1.2 OBJECTIVES

The primary objectives of the SmartBus project are:

- To implement a real-time bus tracking system using GPS.
- To provide accurate ETA predictions using AI and machine learning models.
- To monitor and share seat availability data using sensor technology.
- To create a user-friendly mobile application for passengers to access real-time updates.
- To implement a panic alert feature within the app for enhanced passenger safety.
- To provide a web-based dashboard for transport authorities to manage and monitor fleet operations.
- To ensure system functionality even in low or no internet conditions through SMS/GSM fallback support.

1.3 NEED FOR THE PROJECT

Despite significant advancements in digital services, public transportation systems, particularly those operated by the government, often lack real-time intelligence and modern features expected by users today. Most buses do not offer real-time location visibility, and passengers are left waiting without knowing delays or occupancy levels.

The need for SmartBus arises from the following gaps:

- Passengers are unable to plan their trips due to lack of real-time updates.
- Emergency handling is practically non-existent in many public buses.
- No insight is provided into seat occupancy, resulting in discomfort and unsafe travel.
- Transport authorities lack effective tools for route management, incident monitoring, and optimization.

By addressing these pain points, SmartBus can improve the experience for both passengers and administrators while increasing the reliability and efficiency of the government's public transport offerings.

1.4 SCOPE OF THE PROJECT

The SmartBus system is designed to be scalable, adaptable, and cost-effective. It is particularly suitable for deployment in:

- Urban and semi-urban regions.
- College transportation networks.
- Inter-district government buses.
- Emergency transport (e.g., for disaster or relief management).

The system supports:

- Real-time operation through cloud integration.
- Modular design for additional sensors and features.
- Minimal infrastructure setup due to IoT and mobile-first architecture.

2. LITERATURE REVIEW

2.1. INTRODUCTION

The Literature Review provides an analytical overview of existing methods, technologies, and research studies related to the problem of real-time bus tracking and passenger information systems. It evaluates previous developments in public transport systems, highlighting key innovations and identifying limitations. This chapter establishes the knowledge foundation upon which the present SmartBus system is built and justifies the need for a more integrated, intelligent, and user-centric approach.

2.2. REAL - TIME TRACKING IN PUBLIC TRANSPORTATION

Real-time tracking systems have become an integral part of intelligent transport systems (ITS) in metropolitan areas. These systems utilize GPS modules and mobile communication technologies to capture the real-time position of vehicles, providing arrival predictions and location data to passengers and control rooms.
Examples include:

- Real-Time Passenger Information System (RTPIS) by CDAC, India
- MTA Bus Time in New York
- TfL iBus system in London

Despite their success, these systems often lack features such as passenger-specific information, seat availability, and safety alerts.

2.3. INTEGRATION OF IoT IN BUS TRANSPORT

The introduction of Internet of Things (IoT) devices—such as occupancy sensors, environmental monitors, and vehicle health trackers—has expanded the capability of transport systems. IoT-enabled buses can now transmit real-time data not only on location but also on internal status, such as seat occupancy, temperature, and fuel usage.
However, these systems often:

- Require high investment and complex infrastructure
- Lack integration with commuter-facing mobile applications
- Are primarily deployed in private or high-budget public fleets

2.4. PASSENGER INFORMATION SYSTEM (PIS)

PIS aim to enhance the travel experience by offering passengers up-to-date information on schedules, routes, and arrival times. While effective in many urban transport systems, they typically suffer from:

- Static scheduling with poor responsiveness to delays
- No seat availability feedback
- No mobile-accessible emergency alert features

The SmartBus concept proposes to bridge these gaps by integrating PIS with IoT and mobile platforms.

2. 5. AI AND MACHINE LEARNING IN ETA PREDICTION

Recent research has demonstrated the effectiveness of machine learning models—such as Random Forests, LSTMs, and Gradient Boosting Machines—for estimating bus arrival times (ETA). These models outperform traditional average-speed-based estimates by incorporating:

- Real-time traffic data
- Weather conditions
- Road events and detours
- Time-of-day and day-of-week patterns

However, public bus services rarely deploy such models due to limited resources and lack of technical integration.

2.6. MOBILE APP INTEGRATION IN TRANSIT SYSTEM

Mobile applications have transformed the way commuters interact with transport services. Applications like Moovit, Citymapper, and Transit App provide:

- Route planning
- Live bus tracking
- Disruption alerts

These applications, while powerful, often fail to offer personalized features like seat tracking, bus crowding indicators, or in-app safety alerts. Moreover, many of them rely on third-party APIs and are not directly linked to fleet management systems.

2.7. EMERGENCY SUPPORT MECHANISM IN PUBLIC TRANSIT

Passenger safety is a major concern in public transport, yet few systems offer real-time safety features such as:

- Panic buttons
- Emergency notifications to control centers
- Geolocation-based emergency reporting

Where implemented, these systems tend to rely on physical buttons inside the bus, with minimal integration into mobile apps or cloud dashboards.

2.8. RESEARCH GAP IDENTIFIED

From the analysis of the literature, several key gaps emerge:

- Lack of real-time seat availability data for passengers
- Absence of in-app emergency alert mechanisms
- Underutilization of AI/ML models in government bus services
- No reliable fallback system for poor internet zones
- Weak integration between commuter apps and backend monitoring tools

2. 9. SUMMARY OF RELATED WORK

Technology/Method	Strengths	Limitations
GPS Tracking Systems	Live location	No seat data, no emergency features
IoT in Transport	Rich vehicle data	High cost, low app integration
PIS Displays	Real-time arrival info	Station-based only, not mobile
Mobile Transit Apps	Convenience, routing	No sensor data, no safety tools
AI-Based ETA Models	Accuracy	Not widely implemented in gov fleets

2.10. CONCLUSION

The literature establishes the relevance and importance of integrating GPS, IoT, AI, and mobile platforms to modernize public bus systems. While several successful models exist, they are either limited in scope or require high investment. The SmartBus system is proposed as a scalable, intelligent solution that fills the current gaps by offering real-time location tracking, seat availability, AI-based ETAs, emergency alerts, and mobile-first design — all optimized for deployment in low-resource settings such as government-operated fleets in urban and semi-urban regions.

3. SYSTEM DESIGN

3. 1. INTRODUCTION

This chapter outlines the architectural and functional design of the SmartBus system, which integrates multiple technologies to enhance public transportation services. The system is structured to address real-time bus tracking, seat availability monitoring, ETA prediction, passenger safety, and administrative monitoring. A modular, scalable architecture has been developed to ensure the solution is suitable for government fleets with limited infrastructure and budget constraints.

3. 2. SYSTEM ARCHITECTURE OVERVIEW

The SmartBus system architecture is designed as a multi-layered model comprising four core components:

1. Passenger Interface (Mobile App)
2. On-Bus Hardware System (IoT Devices)
3. Cloud-Based Backend Services
4. Administrator Web Dashboard

These layers work together to ensure seamless communication between passengers, vehicles, and transport authorities.

3. 3. SYSTEM COMPONENTS

3.3.1. PASSENGER MOBILE APPLICATION

- Real-time bus tracking via map interface
- Display of estimated arrival time (ETA)
- Seat availability indicators (occupied/vacant)
- In-app panic button linked to control center
- Offline support via SMS alerts in low/no internet zones

3. 3. 2. DRIVER / VEHICLE INTERFACE

- Trip start/end control
- Real-time status updates (bus full, emergency, etc.)
- GPS data relay
- Minimal UI for ease of use

3. 3. 3. IoT HARDWARE ON BUS

- GPS Module: Captures live location
- Seat Occupancy Sensors: Infrared or ultrasonic sensors detect passenger presence
- Microcontroller (ESP32/Raspberry Pi): Aggregates data from sensors
- GSM/4G Module: Sends data to cloud when Wi-Fi is unavailable
- Battery/Power Supply: Ensures consistent operation

3. 3. 4. BACKEND INFRASTRUCTURE

- Web Server: Developed using Node.js + Express.js
- Real-Time Data Stream: Managed using MQTT/WebSocket
- Database:
    - MongoDB: Stores real-time sensor data
    - PostgreSQL: Stores routes, schedules, user profiles
- AI Module: Built using Python (scikit-learn/TensorFlow.js) for ETA prediction

3. 3. 5. ADMIN DASHBOARD

- Visualizes live bus locations
- Displays emergency alerts
- Tracks seat occupancy
- Provides analytics for route optimization

3. 4. FUNCTIONAL MODULES

Module	Description
GPS Tracker	Collects and transmits real-time location
Seat Detector	Updates bus occupancy data to backend
Panic Alert Handler	Sends emergency notification with GPS
ETA Predictor	Uses ML model to forecast arrival time
Notification Engine	Pushes alerts via Firebase/SMS
Dashboard UI	Visual tool for monitoring and management

3. 5. DATA FLOW DIAGRAM (DFD)

Level 0 Overview:

1. Passenger opens app → Requests bus info
2. App requests data from backend
3. Backend processes real-time data from buses
4. AI module predicts ETA
5. Backend returns live data to app and admin dashboard

Level 1 DFD:

- Sensors → IoT Gateway → MQTT/WebSocket → Cloud Server → DB + ML → App/Admin

3. 6. SYSTEM WORKFLOW

1. Bus ignition triggers GPS & sensor module
2. Data sent to IoT gateway for aggregation
3. Gateway transmits to cloud backend

4. Backend processes and stores data
5. AI model updates ETA
6. Passenger and admin interfaces fetch updated values periodically

3. 7. TECHNOLOGY STACK

Layer	Technology
Mobile App	Flutter / React Native
IoT	ESP32, Raspberry Pi, IR Sensors
Backend	Node.js, Express.js
Databases	MongoDB, PostgreSQL
ML	Python, TensorFlow, scikit-learn
Hosting	AWS EC2, Firebase
Communication	MQTT, GSM/SMS
Monitoring	Grafana, Prometheus

3. 8. DESIGN CONSIDERATION

- Scalability: System supports expansion to multiple routes and regions.
- Modularity: Independent components allow easy maintenance.
- Low-Cost Deployment: Hardware chosen for affordability.
- Offline Support: GSM-based fallback ensures accessibility during outages.
- User-Focused UI/UX: Simple interfaces for passengers and drivers.

3. 9. SECURITY AND PRIVACY

- Authentication via Firebase
- Encrypted transmission of location and panic alerts

- Access control for admin portal
- Anonymization of user data for analytics

3. 10. SUMMARY

The SmartBus system is designed to address major challenges in public transportation through a robust, layered, and technologically sound architecture. The use of real-time tracking, AI-enhanced prediction, IoT integration, and mobile interfaces ensures a future-ready solution that is efficient, safe, and scalable.

4. METHODOLOGY

4. 1. INTRODUCTION

This chapter outlines the step-by-step approach followed during the development of the SmartBus system. The methodology is structured into multiple phases, covering requirement gathering, system design, module development, integration, testing, and deployment. Each phase is described with the tools, techniques, and decisions involved to ensure the system's efficiency, accuracy, and usability.

4. 2. METHODOLOGICAL APPROACH

The project adopts an **Iterative Development Methodology**, which supports:

- Frequent testing of components
- Continuous feedback from sample users (students, drivers, admin)
- Gradual integration of modules

The following major phases define the methodology:

4. 3. PHASE 1 – REQUIREMENT ANALYSIS

4. 3. 1. PROBLEM DEFINITION

The primary goal was to address inefficiencies in government-operated public bus systems, particularly:

- Lack of real-time visibility
- Inability to predict arrival times accurately
- Passenger safety concerns
- Lack of occupancy insights

4. 3. 2. STAKEHOLDER INTERVIEWS

Discussions were held with:

- Regular commuters (students, staff)
- Bus drivers
- Transport authorities

4. 3. 3. FUNCTIONAL REQUIREMENT IDENTIFIED

- Real-time bus tracking on mobile app
- Seat availability status per bus
- Panic button with alert routing
- Admin dashboard for live monitoring
- Resilience in low internet conditions

4. 4. PHASE 3 - SYSTEM DESIGN AND PLANNING

System architecture and module-level designs were created during this phase using tools like:

- **Figma** (for UI prototyping)
- **Draw.io / Lucidchart** (for data flow and architecture diagrams)

Key components designed:

- GPS integration logic
- Seat sensor mapping to UI
- Emergency alert routing flow
- AI-driven ETA model pipeline

4. 5. HARDWARE SETUP AND DATA ACQUISITION

4. 5. 1. DEVICE CONFIGURATION

- **ESP32 boards** programmed for sensor and GPS input
- **Ultrasonic sensors** mounted under seats for occupancy detection
- **NEO-6M GPS modules** interfaced with microcontrollers
- **GSM modules** used for SMS backup communication

4. 5. 2. SENSOR CALIBRATION AND TESTING

- Multiple test cases conducted to ensure accurate seat occupancy detection
- Calibration done for real-time response and noise filtration

4. 6. PHASE 4 - SOFTWARE DEVELOPMENT

4. 6. 1 . MOBILE APPLICATION (FRONTEND)

- Built using **Flutter** (for Android and iOS support)
- Features: Live bus tracking, ETA display, panic button, seat info
- Firebase used for user authentication and push notifications

4. 6. 2. BACKEND SERVICES

- Developed using **Node.js + Express.js**
- RESTful API endpoints created for:
  - GPS data storage
  - Seat updates
  - Emergency alerts
  - ETA response requests

4. 6. 3. DATABASE DESIGN

- **MongoDB**: For real-time telemetry data (location, seats)
- **PostgreSQL**: For static data (routes, users, bus metadata)

4. 6. 4. ETA PREDICTION MODULE

- Historical GPS logs collected from sample routes
- Machine Learning model trained using **Random Forest** and **Linear Regression** techniques
- Model deployed using **Flask API** integrated with Node.js backend

4. 7. PHASE 5 - INTEGRATION AND DEPLOYMENT

- IoT devices tested with live buses during dry runs
- Mobile app connected to live cloud backend
- Admin dashboard deployed using React + Firebase Hosting
- WebSocket used for real-time update stream from backend to apps

4. 8. PHASE 6 - TESTING AND VALIDATION

Comprehensive testing included:

- **Unit Testing**: For each software module (app, backend, ML model)
- **Hardware Testing**: Signal strength, GPS accuracy, sensor triggers
- **Integration Testing**: All components working end-to-end in test environment
- **User Testing**: Sample users validated UI usability, accuracy of live updates

4. 9. TOOLS AND PLATFORM USED

Task	**Tools / Platforms**
App Development	Flutter, Dart
Backend	Node.js, Express.js
Database	MongoDB, PostgreSQL
AI/ML	Python, scikit-learn, Flask
Cloud Hosting	AWS EC2, Firebase
Device Programming	Arduino IDE, MicroPython
Testing	Postman, Firebase Emulator, Manual Simulations

4. 10. SUMMARY

The methodology followed a structured, iterative approach allowing parallel hardware-software development. Frequent testing and modular design ensured the system's scalability and real-world applicability. Each phase—from data collection to deployment—was guided by real-world constraints like internet availability, power usage, and ease of use.

5. IMPLEMENTATION

5. 1. INTRODUCTION

This chapter explains the practical implementation of the SmartBus system, covering both the hardware and software components. It elaborates on how individual modules—such as GPS tracking, seat occupancy detection, ETA prediction, mobile applications, emergency alert system, and cloud backend—were developed, tested, and integrated to create a fully functional real-time public bus tracking solution.

5. 2. HARDWARE IMPLEMENTATION

5. 2. 1. GPS TRACKING MODULE

- **Device Used**: NEO-6M GPS Module
- Connected to ESP32 microcontroller using UART interface.
- GPS data parsed using the TinyGPS++ library.
- Latitude and longitude data updated every 5 seconds and sent to the backend server via HTTP/MQTT.

5. 2. 2. SEAT OCCUPANCY DETECTION

- **Sensors Used**: Ultrasonic and Infrared sensors
- One sensor is installed under each seat.
- Sensor readings above a threshold are interpreted as "occupied."
- Data collected via analog/digital pins of ESP32.
- Status sent to backend every 10 seconds for live updates.T

5. 2. 3. IoT GATEWAY

- **Device Used**: ESP32 with built-in Wi-Fi and optional GSM module
- Aggregates GPS and seat sensor data.
- Connects to the internet via 4G or Wi-Fi.
- Includes offline fallback logic: stores data locally and syncs once connected.

5. 2. 4. POWER MANAGEMENT

- DC-DC converter circuit for 12V bus battery to 5V sensor supply.
- Rechargeable Li-ion battery backup in case of power failure.

5. 3. SOFTWARE IMPLEMENTATION

5. 3. 1. MOBILE APPLICATION (FLUTTER)

- Built using Flutter for Android and iOS compatibility.
- Key Features:
  - Real-time bus location on map (Google Maps SDK)
  - ETA display
  - Live seat availability indicator
  - Panic button for emergencies (with location sharing)
  - Firebase login (email/phone-based)

UI Modules

- Home Page: Bus list with live status
- Map View: Live route and location tracking
- Seat Info: Visual seat occupancy display
- Emergency: Panic button triggering alert API

5. 3. 2. BACKEND API SERVER

- Built using **Node.js + Express.js**
- **Core Modules:**
  - /location/update: Receives GPS data from bus
  - /seats/update: Updates real-time seat availability
  - /eta/get: Returns ETA based on AI model
  - /alert/panic: Triggers emergency event with GPS
- APIs secured using Firebase Authentication tokens and API keys.

5. 3. 3. ETA PREDICTION MODEL

- Trained using **Python + scikit-learn** with historical GPS and traffic data.
- Input Features:
    - Bus route, time of day, traffic level, day of week
- Models tried: Linear Regression, Random Forest Regressor
- Final model deployed as a **Flask microservice**, called by backend via REST API.

5. 3. 4. DATABASE INTEGRATION

- **MongoDB**: Stores real-time location and seat sensor updates
- **PostgreSQL**: Stores bus routes, stop coordinates, driver profiles, and user data
- Indexed for fast lookup and minimal latency

5. 3. 5. ADMIN DASHBOARD

- Built using **React.js**
- Features:
    - Live bus tracking on map
    - Real-time seat status for each bus
    - Panic alert log with bus ID and location
    - Analytics: average trip time, peak hours, occupancy trends
- Hosted on **Firebase Hosting**

5. 4. SMS/GSM OFFLINE SUPPORT

- Implemented with **SIM800L GSM Module**
- Sends minimal text-based alerts when internet is unavailable:
    - GPS + seat status every 1 minute
    - Emergency alerts sent immediately
- Server receives and parses SMS messages via a GSM gateway or Firebase cloud function

5. 5. SYSTEM INTEGRATION

- All components tested individually (unit testing)
- Sensors + ESP32 + GSM modules tested in prototype buses
- App connected to live backend; dashboard synchronized via MQTT/WebSocket
- Integration tests conducted for:
    - Simultaneous seat updates + GPS changes
    - Multiple user views on same bus
    - Emergency alert sync across app and admin

5. 6. DEPLOYMENT STRATEGY

- Mobile App uploaded via internal release (Play Store pending)
- Backend hosted on **AWS EC2 Ubuntu instance**
- MongoDB Atlas used for cloud-based NoSQL database
- PostgreSQL hosted via AWS RDS
- Firebase used for:
    - User login
    - Push notifications
    - Hosting admin panel

5. 7. CHALLENGES FACED

Challenge	**Solution**
GPS inaccuracy in test areas	Applied Kalman filtering on location data
Sensor misreads due to vibration	Added median filter + debounce logic

GSM message delays	Buffering with retry mechanism
Flutter build inconsistencies	Switched to stable channel and cleared cache
High latency in AI model	Optimized with model pruning and hosted on a dedicated Flask server

5. 8. SUMMARY

This chapter detailed the complete implementation of the SmartBus system, including sensor setup, mobile application development, AI model integration, backend infrastructure, and dashboard tools. Each module was tested and integrated for real-time performance, accuracy, and usability, making the system ready for live deployment in real-world transport fleets.

6. INTRODUCTION

6. 1. INTRODUCTION

This chapter presents the results obtained from implementing and testing the SmartBus system in a controlled environment. Various components—such as GPS tracking, seat occupancy detection, ETA prediction, emergency alerting, and user interface response—were tested for accuracy, reliability, and efficiency. The performance was analyzed based on predefined metrics to determine the system's viability in real-world deployment scenarios.

6. 2. EVALUATION ENVIRONMENT

The system was tested using:

- 2 prototype buses equipped with ESP32-based IoT hardware
- Simulated GPS path data for various routes
- Android-based mobile application (Flutter build)
- Live cloud backend hosted on AWS
- Admin panel connected to MongoDB and PostgreSQL

6. 3. KEY PERFORMANCE METRICS

Metric	Description
GPS Accuracy	Deviation between actual and reported location
Seat Detection Accuracy	Correct identification of occupied vs. vacant seats
ETA Accuracy	Difference between predicted and actual arrival time

Emergency Response Delay	Time taken to route panic alert to admin dashboard
App Latency	Time delay in UI update from backend
System Uptime	Operational continuity in live testing

6. 4. GPS TRACKING RESULTS

- **Average accuracy**: ±3 meters in open environments
- **Latency**: ~2–3 seconds for position update on user app
- **Map refresh rate**: Every 5 seconds
- **Issue**: Slight location jitter in densely built areas
- **Solution**: Implemented smoothing using Kalman filter

Result: GPS tracking was stable, accurate, and reliable for real-time public use.

6. 5. SEAT OCCUPANCY DETECTION

Total Seats Tested	**Detection Accuracy**	**Missed Detections**	**False Positives**
40 (20 per bus)	95%	2	0

- Sensor values were calibrated against real seat occupation.
- Noise filtering and debounce logic improved reliability.

Result: The seat detection module provided near-accurate data and updated the app every 10 seconds.

6. 6. ETA PREDICTION ACCURACY

- ML model: **Random Forest Regressor**
- Input: Route ID, traffic, day/time, distance
- Sample route ETA: 22 mins
- Predicted ETA: 21.4 mins
- **Average prediction error**: ±1.2 minutes

Metric	Value
MAE (Mean Absolute Error)	1.2 mins
RMSE	1.6 mins

Result: The AI-based model outperformed static estimates and proved valuable in commuter planning.

6. 7. PANIC ALERT SYSTEM

- Emergency alert response tested with both internet and GSM fallback
- Average notification delivery time (with internet): **2.4 seconds**
- Average delay (SMS fallback): **8.9 seconds**
- Panic alert details included bus ID, GPS, and timestamp

Result: Alert mechanism was fast, reliable, and administrator-visible on the dashboard in real time.

6. 8. APP PERFORMANCE (PASSENGER UI)

Feature	Avg. Latency	Max Latency

Live map update	2.3 seconds	4.0 seconds
Seat update	1.8 seconds	3.2 seconds
ETA refresh	2.1 seconds	3.7 seconds
Panic button feedback	2.5 seconds	4.5 seconds

All functions remained responsive, with efficient UI refresh cycles and minimal lag under typical mobile internet conditions.

Result: The app consistently delivered a smooth experience on mid-range Android devices.

6. 9. ADMIN DASHBOARD ANALYTICS

- Real-time fleet view with auto-refresh every 5 seconds
- Panic alerts were logged with time, location, and response acknowledgment
- Bus status chart showed occupancy trends, trip durations, and GPS routes

Result: Admin dashboard allowed effective real-time management and post-trip analytics.

6. 10. SYSTEM RELIABILITY

- **Uptime during test phase**: 98.2%
- Failures: 1 instance of GSM disconnection and 2 Firebase push errors (auto-recovered)
- Offline fallback (SMS) ensured continuous alert flow even without internet

6. 11. SUMMARY OF RESULTS

Component	**Status**	**Remarks**
GPS Tracking	Passed	Real-time and accurate
Seat Detection	Passed	95% accuracy
ETA Prediction	Passed	Avg. error ±1.2 mins
Panic Alert	Passed	Avg. delivery < 3 secs
App Interface	Passed	Smooth UI, real-time sync
Admin Dashboard	Passed	Accurate, responsive
GSM Fallback	Passed	Functional during outage

---

6. 12. CONCLUSION

The SmartBus system achieved high performance and reliability in all test categories. The integration of IoT hardware, AI prediction models, mobile apps, and administrative tools demonstrated the project's feasibility for large-scale deployment in public transport networks. The system's modular design and fallback mechanisms further enhance its real-world applicability, especially in semi-urban and rural government-operated fleets.

7. CONCLUSION

7. 1. CONCLUSION

The **SmartBus** project successfully addresses several longstanding challenges in public transportation systems, particularly those operated by government fleets. Through the integration of **real-time GPS tracking**, **IoT-based seat occupancy detection**, **AI-powered ETA prediction**, and **in-app emergency alert systems**, the project significantly improves the **efficiency, safety, and user experience** for daily commuters.

The implemented system demonstrated strong performance during testing, with:

- Accurate bus location tracking (±3m GPS accuracy)
- Reliable seat detection (95% accuracy)
- Low-latency ETA updates (average error ±1.2 minutes)
- Effective panic alert routing (response time under 3 seconds)

The modular architecture allows each component—hardware, software, AI, and cloud services—to operate independently yet cohesively, ensuring easy maintenance, scalability, and adaptability for future upgrades.

Moreover, the system is designed with cost-efficiency and offline fallback in mind, making it suitable for deployment in **semi-urban, rural, or resource-constrained environments** where public transit remains underserved by modern digital tools.

By empowering passengers with accurate, real-time information and enabling transport authorities to monitor and manage fleets effectively, SmartBus bridges the digital divide in public mobility services and lays the groundwork for next-generation smart transit networks.

7. 2. FUTURE SCOPE

While the SmartBus prototype has proven functional and reliable in controlled and semi-live environments, several enhancements are envisioned for future phases of development and deployment:

7.2.1 FULL - SCALE DEPLOYMENT

- Expand hardware installation to a large number of government buses across multiple routes.
- Collaborate with municipal or state transport departments for real-time field testing and feedback.

7. 2. 2. AI MODEL IMPROVEMENTS

- Integrate live traffic APIs and weather data into the ETA prediction pipeline.
- Train deep learning models (e.g., LSTM, XGBoost) for better time-series forecasting.

7. 2. 3. DYNAMIC ROUTE CONFIGURATION

- Enable buses to reroute automatically during major disruptions based on traffic and road alerts.
- Allow admin dashboards to suggest alternate paths in real time.

7.2.4. PASSENGER FEEDBACK INTEGRATION

- Add in-app feedback/rating system for buses, drivers, and journey experience.
- Use sentiment analysis for service quality assessment.

7. 2. 5. MULTILINGUAL AND ACCESSIBILITY SUPPORT

- Add language options in the app to accommodate diverse user bases.
- Introduce accessibility features like voice commands and screen readers for differently-abled users.

7. 2. 6. INTEGRATION WITH DIGITAL PAYMENTS

- Add ticket booking and payment via UPI or smart cards.
- Sync journey data with payment records for analytics.

7. 2. 7. CLOUD - BASED PREDICTIVE MAINTENANCE

- Use sensor data for early detection of vehicle health issues.
- Predict breakdowns using machine learning to reduce service interruptions.

7. 3. FINAL REMARKS

SmartBus represents a significant leap toward **intelligent, inclusive, and transparent public transport systems**. The project showcases how emerging technologies—when applied thoughtfully—can reshape traditional services to better serve the community. With continued development, institutional collaboration, and user-centric iterations, SmartBus has the potential to become a standard in public transport modernization across the country.

REFERENCES

1. Ariponnammal, S. and Natarajan, S. (1994) *Transport Phenomena of Sm Sel – X Asx*, Pramana – Journal of Physics, Vol.42, No.1, pp.421–425.

2. Barnard, R.W. and Kellogg, C. (1980) *Applications of Convolution Operators to Problems in Univalent Function Theory*, Michigan Math. J., Vol.27, pp.81–94.

3. Shin, K.G. and Mckay, N.D. (1984) *Open Loop Minimum Time Control of Mechanical Manipulators and its Applications*, Proc. Amer. Contr. Conf., San Diego, CA, pp. 1231–1236.

4. Sivanandam, S.N., Deepa, S.N. (2011) *Principles of Soft Computing*, Wiley India Pvt. Ltd., 2nd Edition.

5. Kumar, R., and Gupta, P. (2017) *IoT-Based Real-Time Bus Monitoring and Passenger Information System*, International Journal of Computer Applications, Vol. 164, No. 9, pp.1–5.

6. Singh, A. et al. (2020) *AI-Driven Public Transport Systems: A Review*, IEEE Access, Vol. 8, pp. 182490–182505.

7. Google Transit. (2021) *General Transit Feed Specification (GTFS).* https://developers.google.com/transit/gtfs

8. OpenStreetMap. (2023) *Open-Source Geospatial Data Platform.* https://www.openstreetmap.org