**A TERM PAPER REPORT**

**ON**

**IOT BASED SCHOOL BUS TRACKING AND ARRIVAL TIME PREDICTION**

**Submitted in partial fulfilment of requirements to**

**ACHARYA NAGARJUNA UNIVERSITY**

**For the TERM PAPER work in**

**III/IV B.Tech, in IoT (VI SEMESTER)**

**By**

**CH. PUJITHA (Y21CO011)**

**SK. SALMA (Y21CO048)**

**V.THANUJA (Y21CO054)**

**B. TRINADH (L22CO062)**

****

**APRIL-2024**

**R.V.R & J.C. COLLEGE OF ENGINEERING**

**(AUTONOMOUS)**

**(Affiliated to Acharya Nagarjuna University, GUNTUR)**

**(Approved by AICTE & Accredited by NBA)**

**CHANDRAMOULIPURAM: CHOWDAVARAM, GUNTUR-522019(A.P.)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (IOT)**

**R.V.R & J.C COLLEGE OF ENGINEERING**

**(AUTONOMOUS)**

****

**BONAFIDE CERTIFICATE**

This is to certify that this Term Paper work titled **“IOT BASED SCHOOL BUS TRACKING AND ARRIVAL TIME PREDICTION**” is the Bonafide work of **SK.SALMA (Y21CO048), CH.PUJITHA (Y21CO011), V.THANUJA (Y21CO054), B.TRINADH (L22CO062),** who have carried out the work under my supervision, and submitted in partial fulfillment for the award of the degree, **B.TECH** during the year **2022-2023.**

**Term paper guide** **B.NAGARAJU**

Designation Asst.Prof, Dept of IoT

**ACKNOWLEDGEMENT**

The successful completion of any task would be incomplete without proper suggestion, guidance and environment. The combination of these three factors acts like the backbone to our Term Paper **“IOT BASED SCHOOL BUS TRACKING AND ARRIVAL TIME PREDICTION”.**

We would like to express our gratitude to the Management of R.V.R&J.C COLLEGE OF ENGINEERING for providing us with a pleasant environment and excellent lab facility.

We regard our sincere thanks to our Principal, **Dr.Kolla.Srinivas** for providing support and a stimulating environment.

We are greatly indebted to **Dr.N.NAGAMALLESWARAO**, Professor and Head of the Department Information Technology, for her valuable suggestion during the course period.

We would like to express our special thanks of gratitude to our guide **Mr.B.NAGARAJU**  who helped us in doing the Term Paper successfully.

We would like to thank our Term Paper in-charge **Mr.P.PRUDHVI KIRAN** who gave us the opportunity to do this work.

SK. SALMA (Y21CO048)

CH. PUJITHA (Y21CO011)

V.THANUJA (Y21CO054)

B. TRINADH (L22CO062)

**ABSTRACT**

Nowadays, parents are perturbed about school going children because of the increasing number of cases of missing students. On occasion, students need to wait a much longer time for the arrival of their school bus. There exist some communication technologies that are used to ensure the safety of students. But these are incapable of providing efficient services to parents. This paper presents the development of a school bus monitoring system, capable of providing productive services through emerging technologies like Internet of Things (Iota). The proposed IoT based system tracks students in a school bus using a combination of RFID/GPS/GSM/GPRS technologies. In addition to the tracking, a prediction algorithm is implemented for computation of the arrival time of a school-bus. Ensuring the safety and security of school students is a prime concern for society, the world over. There have been numerous reports such as kidnapping of school students on their way home or to school d and delaying of school bus due to road traffic. Technology can provide a comprehensive solution to this vexing problem. School vehicle tracking systems play a major role in the safety of school children. For tracking a school bus, we propose an IoT-based school bus monitoring system. The mobile app can continuously update the transit times, coordinated with location identification of GPS modules. Through an Android application, parents can continuously monitor the bus route and forecast arrival time of the bus. A smartphone application empowers the end user to monitor the location of school buses, and view the route path, using Google maps.

**CONTENTS**

**S.NO**  **DESCRIPTION**  **PAGE NO**

1. INTRODUCTION

1.1 Overview of Problem Area

1.2 Solution and outline of the project

1.3 System Hardware

1.3.1 GPS Tracking Unit

1.3.2 RFID Tag

1.3.3 RFID Reader

1.3.4 Communication layer

1.3.5 Application layer

1.3.6 Proposed System

2. LITERATURE SURVEY

2.1 A real Time Implementation of Smart Bus

2.2 City Bus Tracking and Smart Ticketing System

2.3 Smart Assistance for Public Transport System

2.4 Smart Bus: A Tracking System for School Buses

2.5 IOT Based Bus Tracking and Arrival Time Prediction

3. SOFTWARE REQUIREMENTS SPECIFICATIONS

3.1 Requirement Analysis

3.2 Functional Requirements

3.3 Non-Functional Requirements

4. DESIGN AND ANALYSIS

4.1 System Architecture

4.2 Activity Diagram

4.3 Data Flow Diagram

4.4 Algorithm

4.5 Class Diagram

5. IMPLEMENTATION

5.1 Platform Selection

5.2 Programming Language Gist

5.3 Module Description

5.4 Coding Standards

6. SOFTWARE TESTING

6.1 Unit Testing

6.2 Integration Testing

6.3 System Testing

6.4 Functional Testing

7. RESULTS AND DISCUSSION

8. CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

8.2 Limitations

8.3 Future Enhancement

9. REFERENCES

9.1 Books

9.2 Website Links

9.3 Reference Papers

**LIST OF FIGURES**

**S.NO Figure DESCRIPTION PAGENO**

1. 1.1 GPS Module 11

2. 1.2 RFID Tag 12

3. 1.3 RFID Reader (125 kHz) 13

4. 1.4 Architecture Diagram 16

5. 4.1 System Architecture 29

6. 4.2 Activity Diagram 33

7. 4.3 Data Flow Diagram 34

8. 4.4 Prediction Modeling 36

9. 4.5 Flow chart 38

10. 4.6 Class Diagram 39

11. 6.1 Unit Testing 53

12. 6.2 Integration Testing 54

13. 7.1 Output Interface 56

14. 7.2 Connection 57

15 7.3 Comparision between predicted and

actual arrival time in morning time 57

16 7.4 Comparision between predicted and

actual arrival time in morning time 58

17 7.5 Bus Location 59

18 7.6 Expected arrival time (ETA) 59

19 7.7 Scheduled and Predicted arrival 60

time of bus

20 7.8 SMS send to Transport InCharge 61

**1.INTRODUCTION**

**1.1 Overview of Problem Area**

Ensuring safety an security of school students is a prime concern for society, the world over. There have been numerous reports such as kidnapping of school students on their way home or to school d and delaying of school bus due to road traffic [1]. A recent study shows that number of missing children across the country, increased by 84 percent between 2013 and 2016[2]. Technology can provide a comprehensive solution to this vexing problem. School vehicle tracking systems play a major role in the safety of school children. For tracking a school bus, we propose an IoT-based school bus monitoring system. IoT is defined as the inter-communication of "connected devices" and "smart devices", and other items embedded with electronics, software, sensors, actuators, and network connectivity, which enable these objects to collect and exchange data. [2]. An in-vehicle device, using Radio Frequency Identification (RFID) tag, identifies each student and tracks the bus via the global positioning satellite (GPS) module. A GSM module is deployed to notify the parents about the entry and exit status of students, and up to the minute reports, of running status of the bus, appended with location and speed of the bus. Short message alerts are sent to parents if the predefined route schedule of the school bus is modified. The information collected by a GPS receiver in the vehicle is communicated to a Cloud server. A smartphone application empowers the end user to monitor the location of school buses, and view the route path, using Google maps. The mobile application also determines the arrival times of a School Bus, using a Kalman filtering-based prediction algorithm. The mobile app has the ability to continuously update the transit times, coordinated with location identification of GPS modules. It uses historical data and real-time information, for computation of the arrival times [3]. Archived bus transit history can be useful to both parents and School Authorities, in future analysis, towards ensuring the safety of children. This paper provides a framework for integration of both hardware (tracking device) and software subsystems (Kalman filtering algorithm and Android smartphone application). The proposed work focuses on the design and performance analysis of the school bus tracking system. The collected data is stored in Ubidots/IoT Cloud platform for processing at a later date.

**1.2 Solution and Outline of the Project**

There is a need to monitor the travel time and driver information. So that, in this project a cost-effective idea is proposed by integration of Radio Frequency Identification (RFID)integrated with ZigBee [4]. There are many RFID and other related technologies giving solutions to the above-mentioned problems [5]. But as per our knowledge from the survey they are costly, less available and hard to deploy in all situations. They are mainly having the following drawbacks:

1. Very high cost- When the read range of RFID devices increases, the cost of that device also increases.

2. Non-economic - Small and Middle range managements of educational institutes may not show interest in using costly technologies.

By considering many drawbacks like these, we proposed a system which employs the wireless communication device, ZigBee along with RFID technology in a cost-effective manner. The basic idea of this project is to develop long distance object detection in an easily deployable manner using RFID with ZigBee technologies.

The purpose of this system is to record the in/out time of the bus along with the driver details of that particular bus. Monitoring [6, 7] this information can give many advantages like reducing accidents led by over speeding and monitoring driver activities even many more. With technological advancement of the world all manual work is replaced by automation. We came up with an affordable and simple handheld device powered up by a 12v battery that worked on the basis of ZigBee, RFID reader and a PC with necessary software. The gate keeper can operate this, and the details of both the bus driver and particular bus are recorded in a database in a formal manner, so that one can check the report and can print it. Even surveys can be done on the logged information by a user understandable graphical user interface and report can be generated. The process is explained graphically in below Fig. 1.1.

When the bus arrives at the gate, the gate keeper will scan the RFID card on bus with the handheld device. Then details are transmitted using ZigBee protocol stored in the database.

**System Hardware**

Radio-frequency identification (RFID) is the wireless use of electromagnetic fields to transfer data, for the purposes of automatically identifying and tracking tags attached to objects. RFID systems can be classified by the type of tag and reader.

**1.3.1 GPS Tracking Unit**

A GPS tracking unit is a device, normally carried by a moving vehicle or person, that uses the Global Positioning System to determine and track its precise location, and hence that of its carrier, at intervals. The recorded location data can be stored within the tracking unit, or it may be transmitted to a central location database, or Internet-connected computer, using a cellular (GPRS or SMS), radio, or satellite modem embedded in the unit. This allows the asset's location to be displayed against a map backdrop either in real time or when analyzing the track later, using GPS tracking software. Data tracking software is available for smartphones with GPS capability.

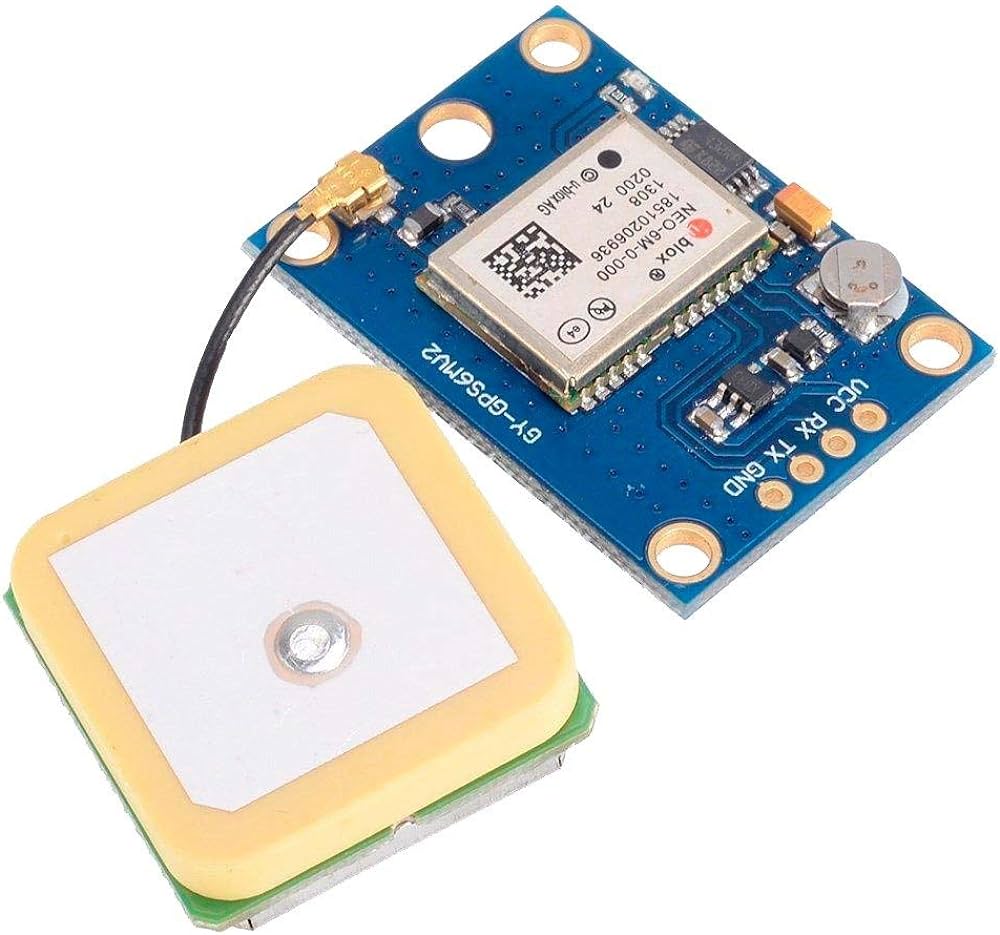


Fig 1.1GPS Module

**1.3.2 RFID Tag**

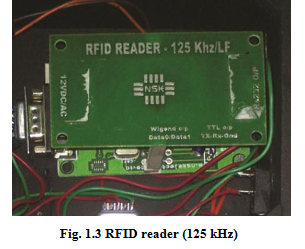
RFID tags contain at least two parts: an integrated circuit for storing and processing information, modulating and demodulating a radio-frequency (RF) signal, collecting DC power from the incident reader signal, and other specialized functions; and an antenna for receiving and transmitting the signal. The tag information is stored in a non-volatile memory. The RFID tag includes either a chip-wired logic or a programmed or programmable data processor for processing the transmission and sensor data, respectively. An RFID reader transmits an encoded radio signal to interrogate the tag. The RFID tag receives the message and then responds with its identification and other information. This may be only a unique tag serial number, or may be product-related information such as a stock number, lot or batch number, production date, or other specific information.



Fig 1.2 RFID Tag

**1.3.3 RFID Reader**

One of the key hardware components in the system is EM 18 RFID Reader. This is the most used RFID reader to read 125 KHz tag. It contains an antenna that can be powered by a 5V power supply. RFID) is an automatic identification technology where digital data is encoded in an RFID tag.



The reader is a radio frequency (RF) transmitter and receiver, controlled by a microprocessor/MCU. The antenna attached inside the reader captures data from tags, then passes the data for processing. Tags are issued to each student, together with their roll numbers. When the tag is placed near the reader, it will get energized and data is transferred to the reader, using radio waves.

tude and speed from NMEA data.

4. Send the message to the recipient/cloud server using AT command through GSM modem.

**Application layer**

**1) Mobile Application:** The latitude and longitude coordinates are transferred to the central server. A mobile application has been developed to access location data and display the vehicles movement in Google map, using Google map API. Users can retrieve information, where users select the bus route number and date, and receive the transit time schedule**.**

**2) Cloud Server:** UBIDOTS is a cloud service to store and analyze information from sensors in real time. It enables creation of applications using IoT. While sending sensor data to Ubidots, it can easily create widgets for viewing in real time, send short message sequence (SMS) or Email Notification. In this project, RFID tag values along with GPS data are sent to Ubidots. The software module is comprised of arrival time prediction and a client-side application. Client-side application allows the user to view the dynamic movement of the vehicle with predicted arrival times. Client-side application was developed on an Android platform.

**1.3.6 Proposed System**

The proposed system consists of both hardware and software module. The hardware layer is composed of a tracking module. The core part of the tracking module is an Arduino microcontroller with various peripherals including RFID reader, GPS receiver and GPRS/GSM modem. On receipt of the data at microcontroller, it is processed and transmitted to Cloud storage for future reference. The software module is comprised of an arrival time prediction and a client-side application. Client-side application allows the user to view the dynamic movement of the vehicle with predicted arrival times. Client-side application was developed on an Android platform. Fig.1 shows the three-layer architecture diagram of the system.

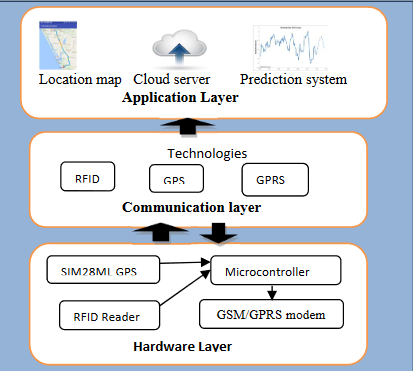


Fig 1.4 Architecture Diagram

An RFID System exists with an RFID tag along with a unique Id. The RFID card reader will read this unique Id and give it to the microcontroller for further processing. The tracking device is placed near the door of the school bus so that an alert message will be sent to parents whenever a student gets onto the bus or leaves the bus. GPS is used for tracking processes. To estimate the accurate arrival time of the bus, a Kalman filtering dynamic algorithm is used.

**2. LITERATURE SURVEY**

**2.1. Shared S, Bagavathi** **Sivakumar P, Anantha Narayanan V,” The** **Smart Bus for a Smart City - A real-** **time implementation”,** [**2016 IEEE**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7938533)[**International Conference on Advanced Networks and Telecommunications**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7938533)[**Systems (ANTS).**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7938533)

The need for a real-time public transport information system is growing steadily. People want to plan their city commutes and do not like waiting for long hours, nor take a long route to reach their destination. The proposed hardware solution in this paper computes the shortest path to reach the destination in real time and gives that information to the bus driver. Artificial Neural Networks (ANN) is used to give an accurate estimate of the arrival time (ETA) to the commuter by means of an application. ETA to the next stop is communicated to the commuter MQTT (MessageQueuingTelemetryTransport) protocol, by the hardware mounted on the bus.The prototype thus developed makes sure commuting in cities is pleasant,and hassle free. The existing platforms and applications that are used to assist commuters plan their travel use mobile data for the connectivity and communication and GPS to get the real-time location of the bus (or other means of transport) relative to the commuter. There are solutions that offer limited accuracy in metropolitan cities. However, these solutions are not available to the other cities, and they rely on historical data to provide information. The Intelligent Transport System (ITS)-based solutions can be studied to overcome these pitfalls, which would help the commuter to effectively utilize public transport which includes lower waiting time. There are many implementations of Intelligent Transport System all around the world, each solution designed to address a specific demographic region. There are existing solutions like tram TRACKER by Yarra Technologies in Melbourne, Australia and Google Maps is always there to cater the needs of the metropolitan commuters. The components of ITS Technologies are wireless communication like Wi-Fi, WiMAX, RFID, etc. and computational technologies like AI, Real time data processing, etc.

**2.2.Ajay Shingare, Ankita Pendole, Nikita Chaudhari and ParikshitDeshpande, Prof. Samadhan** **Sonavane,”GPS Supported City Bus Tracking & Smart**

**Ticketing System”,** [**2015 International Conference on**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7370053)[**Green Computing and Internet of Things (ICGCIoT)**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7370053)Now-a-days increasing density of vehicles on the road is becoming a problem for traffic control. Ultimately obstacles arise in the managing and tracking of the vehicle. Because of the problem state, it is necessary for every organization and individual to track the vehicle. People will monitor and track their vehicles for the safety concerns with the help of our Android app. public transport and private buses tracked to citizens with traffic and transportation details like location, crowd, etc. The proposed system will be used for the positioning of the bus from remote location. The Smart Card based ticketing module which swaps the card to the Smart handheld device for the transaction purpose. The smart ticketing device will also contain the dynamic routes as per the bus depot. The smart device has been enhanced with the GSM and GPS technology and made available with required necessary configurations which makes it more efficient than that of the existing system. The location of the bus can be observed continuously using the GPS system. The GPS satellites transmit signals to a GPS receiver. These receivers statically receive signals. GPS satellite transmits data that indicates the location and current time of the vehicle. The Smart Card provides identification, application processing along with data storage.

These smart cards are capable of recharging. By integrating both GPS technology and smart cards we are going to design a whole bus ticketing system. Whenever the passengers enter the bus, he/she will be asked by the conductor whether he/she wants to buy a ticket by using a smart card or money.

 If a smart card is used, then the conductor will swipe the smartcard. Then validity and of smart card will be checked with server and then the ticket will be issued. According to Source and destination the distance covered by passenger is get calculated and according to that bus fare amount will be reduced from smart card. Smartcards will also be useful for conductors for fast issuing the tickets to the passengers.

**2.3.** [**Reshma Rathod,“Smart assistance for public transport system**](https://ieeexplore.ieee.org/document/7830206/)**”**

[**2016International Conference on Inventive Computation Technologies**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7811903) **(ICICT)**

In this paper we have provided public Smart Assistance in Public Transport System. The project is to be implemented for public buses (for ex: PMTs in Pune). It has the entire smart assistance system required for public security and safety. The smart system includes safety for women as well. It has accident detection and monitoring facility. It also has user friendly application for users to track buses on their smart phones. The smart system can be designed for both online (GPS) and offline (GSM) for user friendly service. Here, GPS system is used to get real-time co-ordinates for offline (GSM) system. It allows the user to save time by acknowledging no. of persons present in the bus as well as no. of seats available in the bus along with the current and next stop acknowledgement with its arrival timings. It also has ramps for handicapped people to provide them with ease to use the bus service. It also has a driver authentication system using RFID tag.

The system also has many additions feature to make public transport system an intelligent and easy to use system, so that the public can take smart advantage of it. The system is specially designed for Smart Cities as it is trending now-a-days. The basic idea of producing a smart public transport system is to be developed on an ARM system using GPS/GSM technology. The basic need to develop this system is to minimize public time issues related to public transport system. Features to be implemented in the system are vehicle tracking (online/offline), availability of no. of seats in the vehicle (bus), engine heat monitoring in the bus, women safety, accident detection, and various other features.

A Smart Assistance for Public Transport System is to be designed. The Public transport selected is Public Bus. The issues related to public transport buses are taken into consideration. The issues such as bus arrival time prediction, no. of persons available in the bus, safety for women’s, accident detection and safety, alcohol detection for driver, speed limiter and indicator, ramp for handicap people, driver authentication using RFID tag and bus report to public through online/offline options are available. In this paper we are focusing on the offline system. The basic methodology used is GPS/GSM.

Smart Bus Tracking System Literature Survey Department of IS&E, BMSIT&M 12 2017-18 (online/offline), availability of no. of seats in the vehicle (bus), engine heat monitoring in the bus, women safety, accident detection, and various other features. A Smart Assistance for Public Transport System is to be designed. The Public transport selected is Public Bus. The issues related to public transport buses are taken into consideration. The issues such as bus arrival time prediction, no. of persons available in the bus, safety for women’s, accident detection and safety, alcohol detection for driver, speed limiter and indicator, ramp for handicap people, driver authentication using RFID tag and bus report to public through online/offline options are available. In this paper we are focusing on the offline system. The basic methodology used is GPS/GSM.

**2.4.**[**Majd Ghareeb,**](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Majd%20Ghareeb.QT.&newsearch=true)[**Athar Ghamlous,**](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Athar%20Ghamlous.QT.&newsearch=true)[**Hawraa Hamdan,**](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Hawraa%20Hamdan.QT.&newsearch=true)**AllBazzi,Samih Abdul-Nabi, “Smart bus: a tracking system for school buses”, 2017 Sensors**

**Networks Smart and Emerging Technologies (SENSET).**

An increased concern for parents is the safety of their children on the way back home from school and the timing of their arrival. Waiting for school buses in the morning and then in the afternoon to return the kids is a time-wasting daily mission for parents, especially with the increasing traffic jams at these hours. In this paper we present a mobile and web application that is designed to address this issue. The system will help parents, the school and the bus to communicate automatically and easily via the application in order to detect kids’ arrival time. The bus application side will notify parents a few minutes before its approaching to their home. Furthermore, the system will allow parents to inform the school and hence bus application side about the absence of their kid. The system has been efficiently and dynamically designed and implemented so it can be hosted and used by any school administration without the need for any major modifications. The objective of this project is to facilitate this task on parents and save their time by automatically notifying them few minutes before the arrival of their kid, so they can go out to receive him from the bus. This is also applied in the morning tour of the bus to inform parents that the bus is approaching, and their kid should go out to take the bus to school. This system can be used by any Lebanese school to increase the safety measures for its student sand to relieve parents from the responsibility of waiting school buses each day.

**2.5.**[**R.C.Jisha,**](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.R.%20C.%20Jisha.QT.&newsearch=true)[**AiswaryaJyothindranath,**](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Aiswarya%20Jyothindranath.QT.&newsearch=true)[**LSajithaKumary,**](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.L%20Sajitha%20Kumary.QT.&newsearch=true)**“**[**IoT based school bus tracking and arrival time prediction**](https://ieeexplore.ieee.org/document/8125890/)**”,** [**2017International Conference on Advances in Computing, Communications**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8119306)[**and Informatics (ICACCI).**](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8119306)

While opting for public transport, time and patience are of more concern. We can also say passengers travelling on public transport found their loss of time due to waiting at the bus stops. We are using the GPS and GPRS modules, the GPS module will locate the buses via the satellite, and the GPRS module will collect all data and send it to the website. The buses will be monitored live using coordinates with this system. Also, by implementing geofence, user get notify once bus entered in his/her predefined area. We are developing an Android application which will give the real-time schedule of buses. Also, it provides quick and real-time replay for inquiry, via the server. Also, in case of any unexpected activities or breakdown, the alert will be sent to the system, with the Bus location. The transportation system provides as the heart in the social and economic growth of the country. As the population in India is increasing, rapid explode in rate of vehicles which results in an overload on traffic management. Public transport is becoming an important part of transport system in urban areas, advance in easily available technology Can be enforced which help the passenger who recalculate between a rural and urban to-get the travelling information and it helps the passengers to comfort them with the final real time location. Public transport, mainly the bus sluice, has been properly developed in many parts of the world. For reducing the fuel-usage, snobby usage of car and comfort traffic crowding we can use the bus services. Passengers require the exact schedule of the bus. The anxiety of passengers increases while waiting for a long time at the bus stop and changes their mind to opt for the Department of IS&E, BMSIT&M.

**3.SOFTWARE REQUIREMENT SPECIFICATIONS**

**3.1Requirement Analysis**

**Real-Time Tracking:**

* The system should be able to track the school bus in real-time using GPS or other location tracking technologies.
* Tracking data should be accurate and updated at regular intervals.

**Mobile Application**:

* A mobile application should be developed for parents, school administrators, and bus drivers to track the bus's location.
* The application should be user-friendly and compatible with both Android and iOS devices.

**Geofencing:**

* Geofencing capabilities should be implemented to notify parents and school authorities when the bus enters or exits predefined geographical zones such as school premises or designated bus stops.

**Arrival Time Prediction:**

* The system should utilize historical data and real-time traffic information to predict the estimated time of arrival (ETA) of the bus at each stop.
* Predictions should be accurate and updated dynamically based on changing traffic conditions.

**Communication System:**

* A robust communication system should be established between the school bus, central server, and mobile applications.
* Communication protocols should ensure data integrity, security, and low latency.

**Emergency Alerts:**

* The system should have provisions for sending emergency alerts to parents, school administrators, and authorities in case of any emergencies or unexpected delays.
* Alerts should be delivered via push notifications, SMS, or email.

**Data Privacy and Security**:

* The system should adhere to data privacy regulations and ensure the security of sensitive information such as student data and location information.
* Encryption protocols should be implemented to secure data transmission and storage.

**Dashboard for Administrators:**

* A web-based dashboard should be provided for school administrators to monitor the overall performance of the system, view bus routes, analyze historical data, and generate reports.

**Scalability and Maintenance:**

* The system should be scalable to accommodate a growing number of buses and users.
* Regular maintenance and updates should be provided to ensure smooth operation and optimal performance.

**Cost-Effectiveness:**

* The solution should be cost-effective in terms of hardware, software, and maintenance expenses.

**Feedback Mechanism:**

* A feedback mechanism should be incorporated to gather input from parents, drivers, and school authorities for continuous improvement of the system. Feedback can be collected through the mobile application or web portal.

**3.2 Functional Requirements**

1. The system should track the school bus's location in real-time using GPS or other location tracking technologies.
2. Location data should be updated at regular intervals and displayed on the mobile application's map interface.
3. Geofencing capabilities should trigger notifications to parents and school authorities when the bus enters or exits predefined geographical zones such as school premises or designated bus stops.
4. The system should predict the estimated time of arrival (ETA) of the bus at each stop based on historical data, real-time traffic conditions, and distance from the current location. Predictions should be accurate and continuously updated as the bus progresses along its route.
5. Emergency alerts should be sent via push notifications, SMS, or email and include relevant information such as the nature of the emergency and the bus's current location.
6. Users should receive notifications for important events such as bus delays, changes in schedule, or emergency situations.
7. Notifications should be customizable, allowing users to specify their preferred communication channels and frequency of updates.
8. School administrators should have access to a web-based dashboard for managing bus routes, viewing real-time bus locations, generating reports, and sending alerts.
9. Bus drivers should have access to a mobile application or device interface for viewing assigned routes, receiving instructions from administrators, and updating their status (e.g., starting, pausing, or completing a route).
10. The interface should display relevant information such as upcoming stops, ETA, and any special instructions or announcements.

**3.3 Non-Functional Requirements**

**Performance:**

* The system should be capable of handling a large number of concurrent users and bus tracking requests without significant degradation in performance.
* Response times for fetching bus location data, calculating arrival predictions, and sending notifications should be minimized to ensure a smooth user experience.

**Reliability:**

* The system should be highly reliable, with minimal downtime and disruptions in service.
* Redundant components, failover mechanisms, and backup systems should be implemented to ensure continuous operation even in the event of hardware failures or network outages.

**Scalability:**

* The system should be designed to scale horizontally to accommodate an increasing number of buses, users, and concurrent requests.
* Scalability should be achieved through distributed architectures, load balancing, and cloud-based infrastructure to support growth without compromising performance or reliability.

**Availability:**

* The system should be available 24/7, allowing users to access real-time bus tracking and arrival time predictions at any time.
* High availability should be ensured through redundant servers, automated monitoring, and rapid incident response procedures.

**Security:**

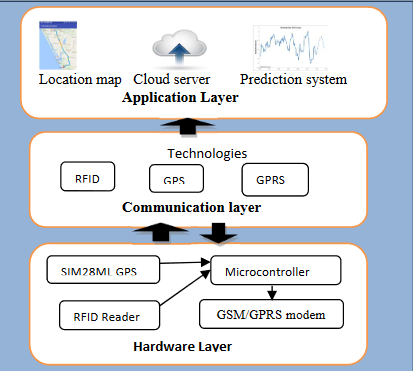
* The system should adhere to industry-standard security practices to protect against unauthorized access, data breaches, and cyber-attacks.
* Security measures should include encryption of sensitive data, secure authentication mechanisms, regular security audits, and compliance with relevant regulations such as GDPR or HIPAA.

**Usability:**

* The user interface of the mobile application and web dashboard should be intuitive and easy to use, requiring minimal training for users to navigate and understand.
* Accessibility features should be implemented to ensure that the system is usable by individuals with disabilities.

**4. DESIGN AND ANALYSIS**

**4.1 System Architecture**



**Fig 4.1 System Architecture**

**A. Hardware Layer**

The system hardware is an Arduino Mega microcontroller (MCU), GPS receiver, RFID reader module and a GPRS/GSM Modem. The RFID reader module is used to read the RFID tag data. Here we use a passive reader. Once the data is received by the reader via the serial port of Arduino, GPS receiver will retrieve the National Marine Electronics Association (NMEA) data set. Then this data is conveyed to MCU for further processing. The microcontroller will extract the relevant data from GPRMC string. After processing the data from GPS and RFID reader, microcontroller will transmit the data to cloud storage through GPRS modem. The various hardware components of the device are explained below.

**1) Microcontroller**: The Arduino mega microcontroller serves as the core part of the tracking module to control the school bus tracking system. A C language program, saved in the microcontroller’s memory, controls the modules. The Arduino Mega 2560 has 54 digital input/output pins, 16 analog inputs, 4 UARTs. The proposed system uses 3 UARTs-one for RFID, one for GPS Receiver, and the third for GPRS/GSM Modem.

**2) RFID Reader**: One of the key hardware components in the system is EM 18 RFID Reader. This is the most commonly used RFID reader to read 125 KHz tag. It contains an antenna that can be powered by a 5V power supply. RFID) is an automatic identification technology where digital data is encoded in an RFID tag. The reader is a radio frequency (RF) transmitter and receiver, controlled by a microprocessor/MCU. The antenna attached inside the reader captures data from tags, then passes the data for processing. Tags are issued to each student, together with their roll numbers. When the tag is placed near the reader, it will get energized and data is transferred to the reader, using radio waves.

**3) GPS Receiver**: The proposed system uses a SIM 28 ML GPS receiver which has excellent low power consumption characteristics. The module has complete signal processing from antenna in NMEA messages. This module requires 12V power supply. The host port is configurable to UART. Transmitter pin of modem is connected to receiver pin of Microcontroller. The Global Positioning System (GPS) helps to provide geographic coordinates of object anywhere on Earth

with exact Universal Time Coordinated time (UTC). GPS offers great accuracy and generates real-time tracking location data. GPS satellite transmits location data. GPS receiver receives the signals in NMEA format. There are different NMEA signals. We are focused on GPRMC signal, since it contains location information along with the speed. Format of GPRMC is given below.

$GPRMC,123519.000, A,7791.0381, N, 06727.4434, E,022.4,084.4,230394,003.1, W\*6A

**4) GSM/GPRS Modem:** SIM900 is designed with a very powerful single-chip processor. Transmitter and receiver pins of modem are connected to receiver and transmitter pins of microcontroller. Modem requires 12 V power supply. The GPRS module is responsible for establishing connections between an in-vehicle device and a cloud for transmitting location information of the vehicle [10].

**B. Communication Layer**

Communication layer is used to establish a connection with underlying hardware, the MCU and the upper layer applications. Various underlying technologies used to develop the application are RFID, GPS and GPRS. Various technologies used for communication are explained below. The proposed architecture uses the following algorithm at the hardware level.

Transmission and Reception Algorithm at

the Microcontroller

Input: UART 1 Rx pin (RFID Reader OUT), UART 3 Rx Pin

(GPS Receive OUT)

Output: UART 2 TX Pin (GSM IN)

1. Read the 12-character RFID data from EM 18 using Microcontroller.

2. Read NMEA data from the GPS.

3. Extract Date, Time, Latitude, Longitude and speed from NMEA data.

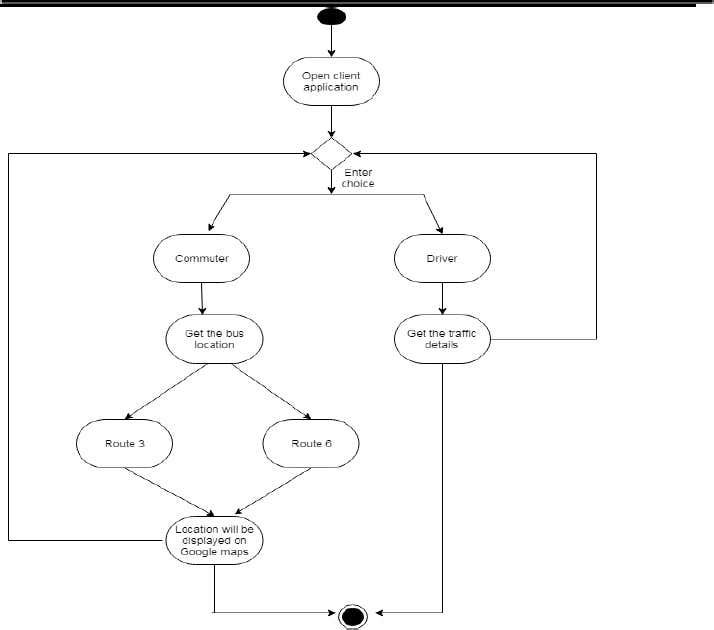
4. Send the message to the recipient/cloud server using AT command through GSM modem

**C. Application Layer**

**1) Mobile Application**: The latitude and longitude coordinates are transferred to the central server. A mobile application has been developed to access location data and display the vehicles movement in Google map, using Google map API. Users can retrieve information, where users select the bus route number and date, and receive the transit time schedule.

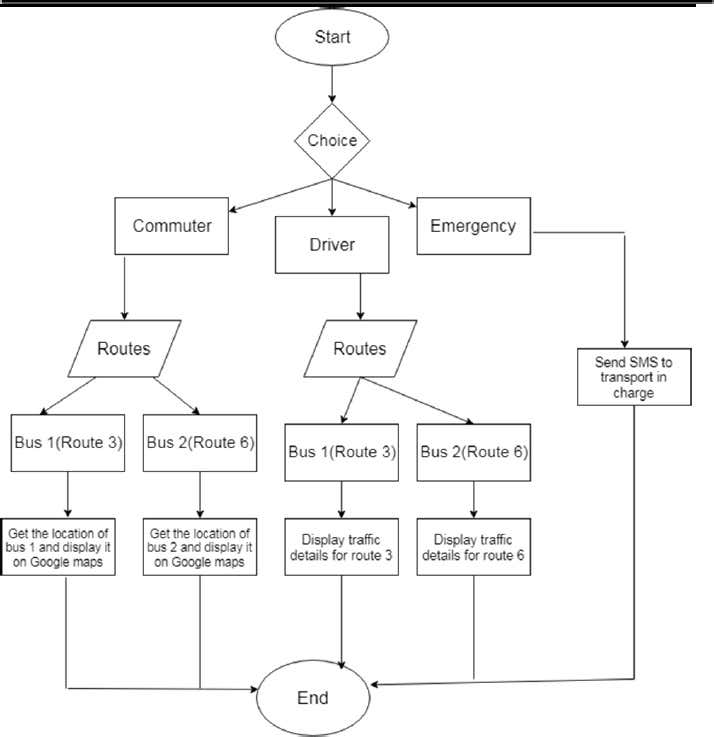
**2) Cloud Server**: UBIDOTS is a cloud service to store and analyze information from sensors in real time. It enables creation of applications using IoT. While sending sensor data to Ubidots, it can easily create widgets for viewing in real time, send short message sequence (SMS) or Email Notification. In this project, RFID tag values along with GPS data are sent to Ubidots. The software module is comprised of arrival time prediction and a client-side application. Client-side application allows the user to view the dynamic movement of the vehicle with predicted arrival times. Client-side application was developed on an Android platform.

**4.2 Activity Diagram**



**Fig 4.2 Activity Diagram**

**4.3 Data Flow Diagram**



**Fig 4.3 Data Flow Diagram**

**DESCRIPTION:**

* **Data Sources:**
  + Commuter
  + Driver
  + Emergency
* **Data Destinations:**
  + Bus Route 3
  + Bus Route 6
* **Data Processes:**
  + The system can receive data from commuters, drivers, and emergency responders.
  + It can then process this data and send it to either Bus Route 3 or Bus Route 6.
* **Data Flows:**
  + The data from the commuters, driver, and emergency responders flows into the system.
  + The system then processes this data and sends it to the appropriate bus route.

**4.4 Algorithm**

**KALMAN FILTERING**

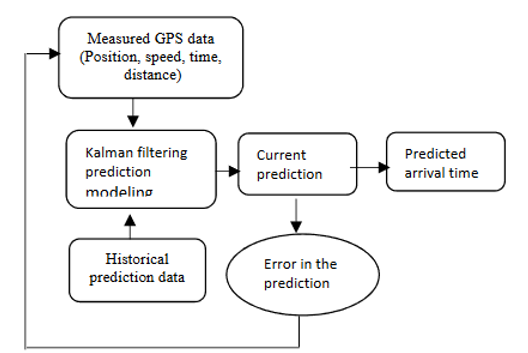
In the proposed framework, we used Kalman filtering algorithm to build the prediction model. Kalman filtering is a powerful method for predictions, in real time systems, since it estimates the state of a system from measured values, which include errors [11]. This is an iterative algorithm, which involves two stages that first predict the system, and then correct the same using measurements.

Kalman filtering has two steps:

1. Prediction

2. Measurement update

In the initial prediction equations, the current estimate is calculated using estimates of the previous state and known information and error co-variances. In the measurement-update equations, an estimate is obtained by combining new measurements from the system. Thus, in Kalman filtering process, initial parameters are predicted, and they are adjusted with every new measurement to make the arrival time prediction more appropriate. It does not require all previous data to be stored and reprocessed every time.



**Fig** **4.4 Prediction Modeling**

ALGORITHM

The following steps describe the algorithm

1. **Step1.** There are three different options commuter, driver and emergency. The students/staff can click on the commuter option to get to the bus location.
2. **Step2**. In step 2, the respective students/staff can click on their routes. By clicking on their routes, they get the information of where exactly their bus is located on Google maps.
3. **Step3.** In the same way, the driver can click on the driver option to get congestion details. When the bus 1 driver clicks on his route he gets the traffic details. In the same way Bus 2 driver can click on his route.
4. **Step4**. The last option is emergency, the students/staff and driver can click on the emergency button if there is any bus breakdown or accident and so on. By clicking on that option an SMS will be sent to transport in charge along with the location.
5. **Step5.** Click on end option to terminate

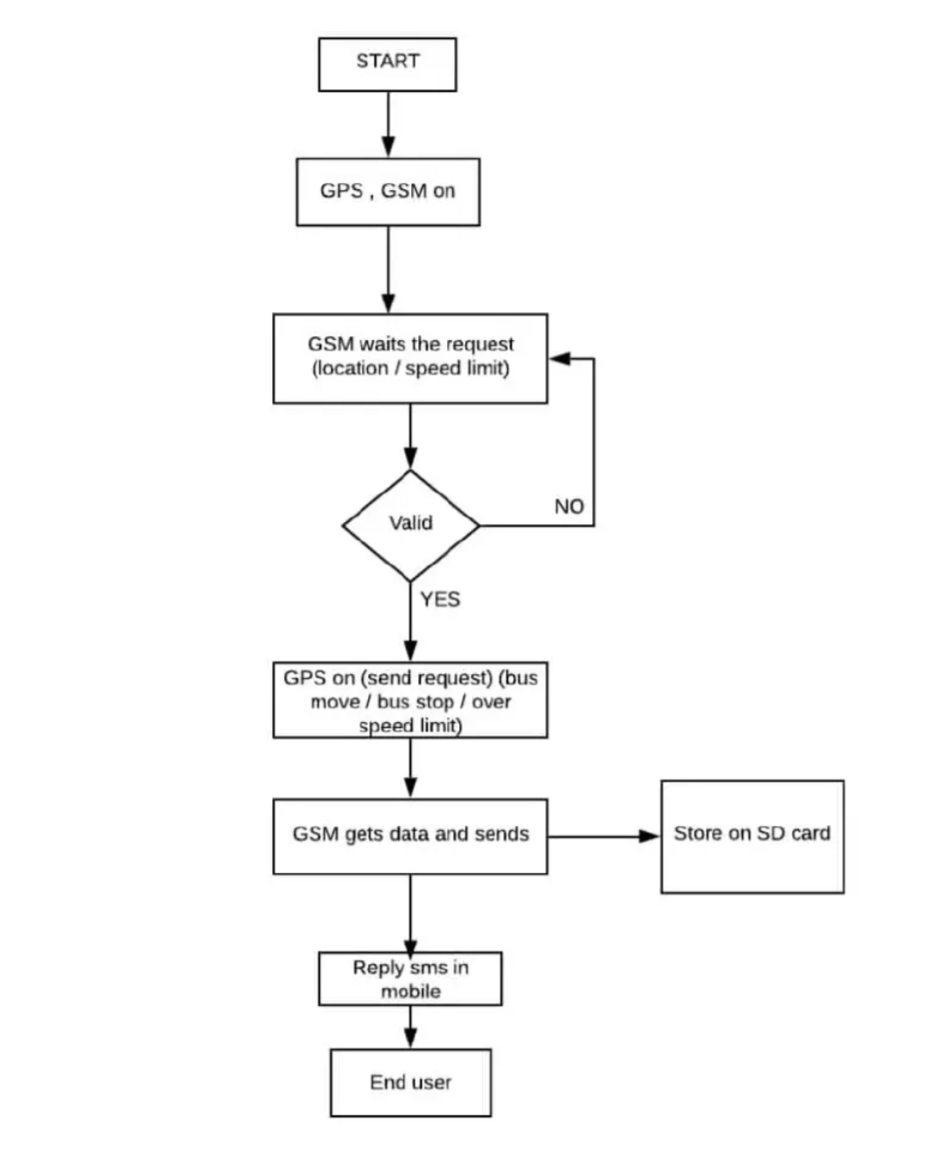
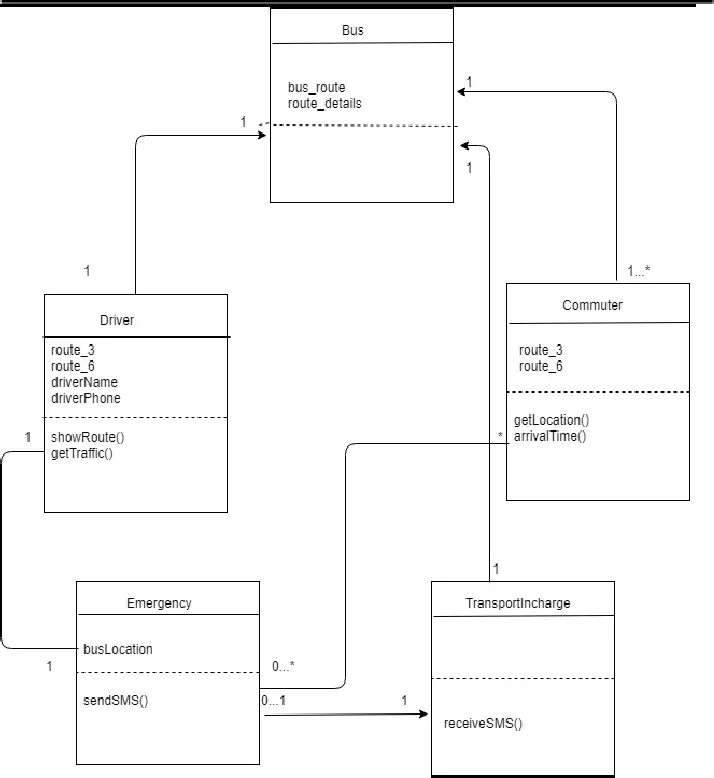


Fig 4.5 Flow chart

**4.5 Class Diagram**



**Fig 4.5 Class Diagram**

**DESRIPTION:**

* **Driver:** The driver has access to a system that shows them their route (route 3 or route 6), route details, and their name and phone number. They can also use the system to display the current bus location and get traffic updates, including estimated arrival times.
* **Commuter:** The commuter interacts with the system by using a function called getLocation(), which presumably provides their location data to the system.
* **Transport in-charge:** The transport in-charge can receive SMS updates from the driver, presumably about delays or emergencies, and can also send SMS messages to the driver.

**5. IMPLEMENTATION**

**5.1 Platform Selection**

Android Studio

It is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designe specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development.

The following features are provided in the current stable version:

* Gradle-based build support
* Android-specific refactoring and quick fixes
* Lint tools to catch performance, usability, version compatibility and other problems
* Progard integration and app-signing capabilities
* Template-based wizards to create common Android designs and components
* A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
* Support for building Android Wear apps
* Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine.
* Android Virtual Device (Emulator) to run and debug apps in the Android studio. Android Studio supports all the same programming languages of IntelliJ, and PyCharm e.g. Python, and Kotlin and Android Studio 3.0 support "Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects back port some Java 9 features.

**5.2 Programming Language Gist**

**Java**

* Android applications are developed using the Java language
* Some of the Java’s important core features are:
* It’s easy to learn and understand
* It’s designed to be platform-independent and secure, using virtual machines
* It’s object-oriented
* Android relies heavily on these Java fundamentals.

The Android SDK includes many standard Java libraries (data structure libraries, math libraries, graphics libraries, networking libraries and everything else you could want) as well as special Android libraries that will help you develop awesome Android applications.

**5.3. MODULE DESCRIPTION**

**5.3.1. Commuter module**

Project is designed in four parts where each part is responsible for different aspects. Essentially, the main activity handles instantiate methods and pre-configuration processes for Location Manager class such as manage the previous location, current location and status check whether GPS is on or off.

**5.3.1.1. Extraction of information from remote XML**

In order to extract the latitude and longitude from the XML, we use the Xml Pull Parser interface. XML Pull Parser is an interface that defines parsing functionality provided in XMLPULL V1 API. This is a simple interface

– parser consists of one interface, one exception and one factory to create parser. An XML file consists of 4 major components:

* **Prolog: The** first line that contains the information about a file is prolog. Typically, this is the line:
* **Events: Events** in an XML file include simple start and end tags and more
* **Text: It's** simple text in between two tags. Example: My Text</Random Tag
* **Attributes: Attributes** are the additional properties of a tag that are present within the tag. Example: Some Text or nested tags.

Steps required to parse a XML using XML pull parser are,

* Get instance of XMLPULL factory
* By default, factory will produce parsers that are not namespace aware; to change setNamespaceAware() function must be called
* Create an instance of the parser
* Then set parser input
* Start parsing. Typical XMLPULL application will repeatedly call next() function to retrieve next event, process event until the event is END\_DOCUMENT. We will be able to get the latitude and longitude and store them in the variable's lati and longi

try { xml Pull Parser Factory=

Xml Pull Parser Factory. new Instance();xml Pull Parser Factory .set Name space Aware(false); parser=xml Pull Parser Factory. New Pull Parser();}

catch(Xml Pull Parser Exception e) {

e. print Stack Trace();}

Private String get Loaded Xml Values(Xml Pull Parser parser)

throws Xml Pull Parser Exception, IO Exception {

Int event Type = parser. Get Event Type();

String name =null;

Entity m Entity =new Entity();

While (event Type != Xml Pull Parser. END\_DOCUMENT ){

if(event Type == Xml Pul l Parser. START\_TAG ){

name = parser. get Name();

if(name. equals("l at")){

M Entity. lati= parser. Next Text();}

else if(name. equals("l on")){

M Entity. Long i= parser. Next Text();}}e

Vent Type = parser. next();}

Return m Entity. lati+", "+ m Entity . Long i;}

public class Entity{ public String l at i;

Public String long i;

}

}

**5.3.1.2. Set up connection to internet**

Android AsyncTask is an abstract class provided by Android which gives us the liberty to perform heavy tasks in the background and keeps the UI thread light thus making the application more responsive. Here we use the BackgroundAsyncTask function to make a connection to the internet as a background task.

private class Back ground Async Task extends Async Task<String, Void, String> {

@Override

Protected String do In Background(String ...params) {URL url =null ; String returned Result ="";

try

{ url = new URL(params[0]);}

catch(Malformed URL Exception e) {

e. print Stack Trace();}

Http URL Connection conn =null;

try

{conn = (Http URL Connection)url . Pen Connection();

Conn. Set Read Timeout(10000);

Conn. set Connect Timeout(20000)

;conn .set Reques t Method ("GET");

Conn. Set Do Input(true);

Conn. connect();

Input Stream is = conn. Get Input Stream();

Parser. Set Input (is, null);

Returned Result = get Loaded Xml Values( parser);}

catch(IO Exception e)

{e. Print Stack Trace();}

Catch (Xml Pull Parser Exception e) {

e. print Stack Trace();}

Return returned Result;}

**5.3.1.3. Display the coordinates of bus on Google maps**

Android uses URI string as the basis for requesting data in a content provider and for requesting actions. Intents let you start an activity in another app by describing a simple action you'd like to perform (such as "display a map" or "show directions to the airport") in an Intent object. The Google Maps app for Android supports several different intents, allowing you to launch the Google Maps app and perform actions

**Use of geo:** Intent to display a map at a specified location and zoom level. q defines the place(s) to highlight on the map.

public void display Map(double clat, double clong){

try{

String ge Code ="geo:0,0?q="+ clat +","

+ clong ;

Intent send Location To Map =new Intent(Intent. ACTION\_VIEW ,

Uri. parse(geo Code));

Start Activity(send Location To Map);}

Catch (Exception e){}

}

**5.3.1.4. Use Location Manager to get current location**

This class provides access to the system location services. These services allow applications to obtain periodic updates of the device's geographical location, or to fire an application-specified Intent when the device enters the proximity of a given geographical location. It checks for permission for access of location using the

Permission. ACCESS\_COARSE\_LOCATION and uses get Latitude and get Longitude to get the user’s

current location.

private void get\_ l at\_ long\_ details() {

// turn GPS On();

M loc Manager= (Location Manager)

get System Service (Context. LOCATION\_SERVICE );

**Driver Module**

To get traffic details using navigation:

String geoCode =

"https://www.google.com/maps/dir/?api=1&origin=BMSCE,Bengaluru&destination=BMSIT,Bengaluru&waypoints=Lalbagh+West+Gate%2CBengaluru%7CMallya+Hospital%2CBengaluru%7CHalasuru+Police+Station%2CBengaluru%7CColes+Park%2CBengaluru%7CTV+Tower%2CJayamahal+Main+Road%2CBengaluru%7CRT+Nagar+Police+Station%2CBengaluru&travelmode=driving&dir\_action=navigate";

This API lets us use Google maps in order to display the traffic details. Here we specify theorigin and destination and the mode of travelling

**5.3.3. Emergency Module**

To send SMS : Sms Manager class is responsible for sending SMS from one emulator to another or device. MY\_PERMISSION\_REQUEST\_SMS and SEND\_SMS checks for permission and allows the application to send the SMS. A Geocoder class uses either a location name or a location's latitude and longitude values to get further details about an address.

Geocoder geocoder;

List <Address> addresses;

geocoder = new Geocoder (this, Locale. get Default());

addresses = geocoder. Get From Location (c\_lat,c\_long,1);

String address = addresses. get(0).get Address Line(0);

String city = addresses. get(0).get Locality();

String state = addresses. get(0).get Admin Area ();

String country = addresses. get(0).get Country Name();

String postal Code = addresses. get(0).get Postal Code();

String known Name = addresses. get(0) .get Feature Name();

address = address + city + state + country + postal Code + known Name;

Toast. make Text ( get Application Context(),"Address is: "+ address, Toa st. LENGTH\_ SHORT ).show(); if(Context Compat. check Self Permission(this,

Manifest. permission. SEND\_SMS )!= Package Manager. PERMISSION\_GRANTED ) {

if (Activity Compat. should Show Request Permission Rationale (this,

Mani fest. Permission. SEND\_SMS )) {

}else{

// permission is already granted

Activity Compat. request Permissions (this,

New String[]{Mani fest. Permission. SEND\_ SMS },

MY\_PERMISSIONS\_REQUEST\_SEND\_SMS);} }

else{

Sms Manager sms Manager = Sms Manager. get Default ();

sms Manager. send Text Message("+918105730024",null,"Help Needed. Location: "+ address ,null, null);

Toast. make Text (get Application Context (),

"SMS Sent Successfully",

Toast. LENGTH\_ LONG). show();}

private voidget\_lat\_long\_details() {

// turnGPSOn(); mlocManager= (LocationManager)

getSystemService(Context.LOCATION\_SERVICE );

**5.4 Coding Standards**

* This project is developed under the coding standard of the Java and user library used in it.
* Naming conventions of java language is followed for method and variable names.
* Begin method names with a strong action verb (for example, location).
* If the verb is not descriptive enough by itself, include a noun (for example, getLatitudegetLongitude). adScroll for next page
* Add adjectives if necessary to clarify the noun(forexample,getLastLocationRecord)
* Use the prefixes get and set for getter and setter methods. Getter methods merely return the value of an instance variable; setter methods change the value of an instance variable.
* For example:
* use the method names getAddress and setAddress to access or change the instance variable address
* Abbreviations are not used in declaring the name of method, class, activity, adapter, fragment and variables.
* Control structures, conditions, loops etc. are used wisely.
* Repetitive code must be written is methods.
* Avoid using the same name of inbuilt keywords.

**6. SOFTWARE TESTING**

Software testing is a process used for verifying the correctness, completeness and quality of the developed software. Software is built out of sub-systems that are composed of modules, which in turn are composed of procedures and functions. The sequence of testing activities performed for the tracking system is as below:

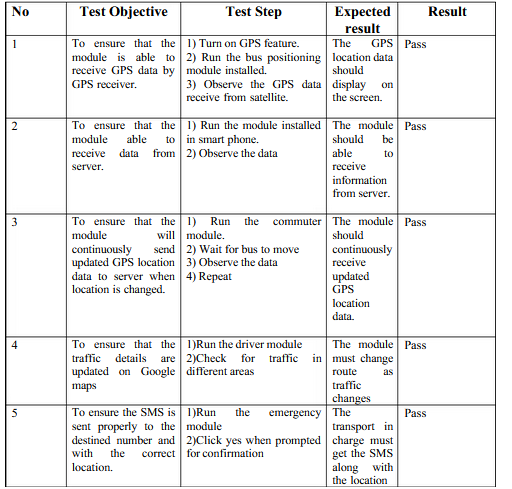
* + Unit Testing
  + Integration Testing
  + System Testing
  + Functional testing

**6.1 Unit Testing:**

There exist a number of components in every sub-system. Every component is tested using respective test procedures. Each component is tested individually based on their needs. Unit test focuses verification effort on the smallest unit of the software design component. Each module can be tested using the following two Strategies:

**Black Box Testing:** In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program.

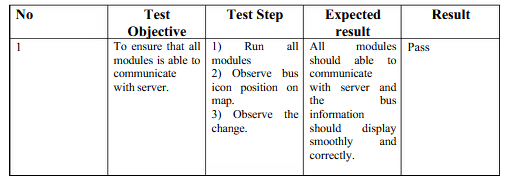
**White Box testing:** In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases.



**Table 6.1 Unit Testing**

**6.2 Integration Testing:**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration testing ensures that software and subsystems work together a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together.



**Table 6.2 Integration Testing**

**6.3 System Testing:**

System testing is the stage of implementation. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding a yet undiscovered error. System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

**6.4 Functional Testing**

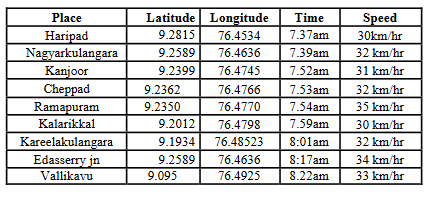
Involves in-house testing of the entire system before delivery to the user. Its aim is to satisfy the user the system meets all requirements of the client's specifications. Functional testing means testing the application against business requirements. Functional testing is executed using the functional specifications given by the client or by the design specifications according to use cases given by the design team. The role of functional testing is to validate the behavior of an application.

* Functional testing means testing the application against business requirements.
* Functional testing is executed using the functional specifications given by the client or by the design specifications according to use cases given by the design team.
* The role of functional testing is to validate the behavior of an application.

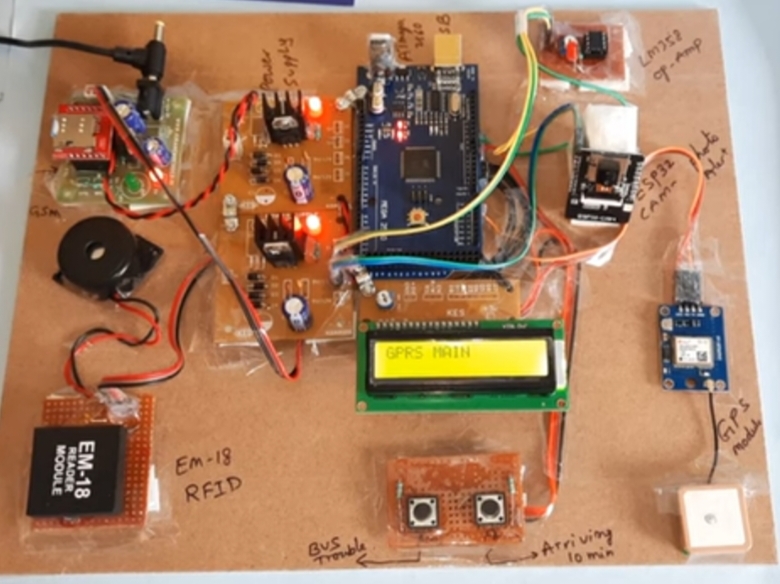
**7. RESULTS AND DISCUSSION:**

**Results and Discussion:**

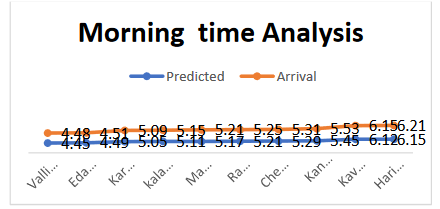
The proposed school bus tracking system integrates both hardware and software modules to provide real-time tracking and arrival time predictions for parents. The hardware layer utilizes Arduino microcontroller along with RFID, GPS, and GPRS/GSM modules for data collection and transmission. The software module includes an Android application for user interface and prediction algorithms for estimating arrival times.



**Fig 7.1 Output Interface**

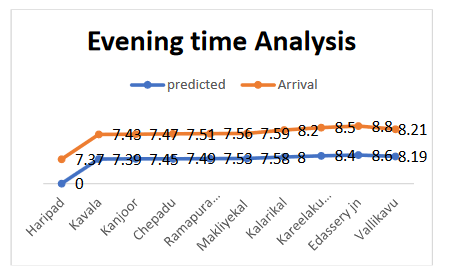


**Fig.7.2 connection**



**Fig.7.3: Comparison between predicted and actual arrival time in**

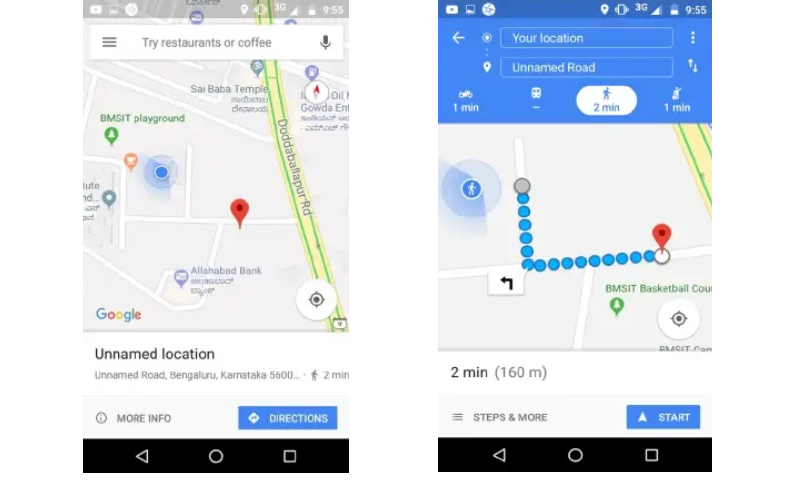
**morning time**



**Fig.7.4: Comparison between predicted and actual arrival time in**

**Evening time**

**Screenshots**



**Fig 7.5 Bus Location Fig 7.6 Expected Arrival Time**

**(ETA)**

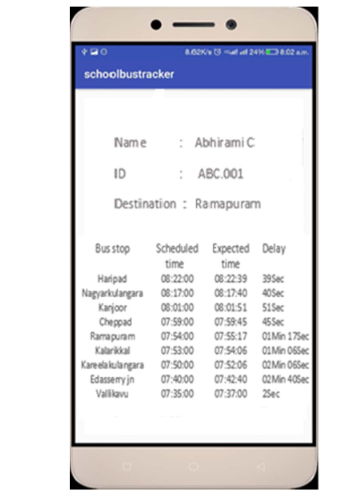


Fig.7.7: Scheduled and predicted arrival time of bus

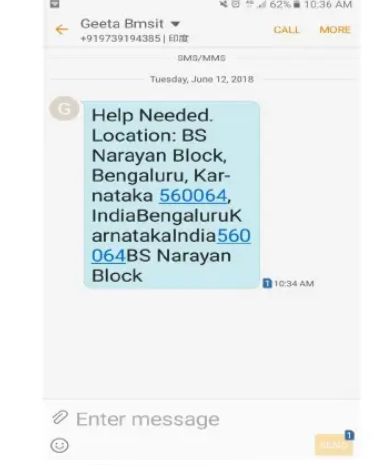


Fig.7.8 SMS send to Transport InCharge

**Data Collection and Analysis:** GPS data collected from school buses are transmitted in real-time to the cloud server. Historical data, along with real-time data, are used to predict arrival times

at each bus stop. The Kalman filtering algorithm is employed for prediction modeling, which estimates the state of the system based on measured values and errors. The algorithm iteratively predicts and corrects the arrival time, resulting in more accurate predictions.

**Result and Analysis:**

The Android application provides parents with real-time information about the school bus's current location and expected arrival times at each bus stop. The system's effectiveness is evaluated by comparing predicted arrival times with actual trip data. The analysis shows that the predicted arrival times have an error rate of 11.31%, with the accuracy assessed using the root mean square error (RMSE). Sample plots comparing predicted and actual arrival times demonstrate the system's performance.

**8.** **CONCLUSION AND FUTURE ENHANCEMENT**

**8.1 CONCLUSION:**

We developed a school bus tracking device, in collaboration with RFID, GPS and GSM/GPRS technologies. Whenever a child enters/exits the bus, the tracking system can notify the child’s parents, through SMS alerts. An Android application has been developed to display vehicle location on Google maps and display arrival times of a school bus, at each of its designated bus-stops. The implementation is very cost-effective, as it is based on easily accessible electronic devices. To predict the expected arrival time of the school bus, at each node (stop), we resorted to Kalman filtering dynamic algorithm. The accuracy of the algorithm calculated using RMSE was shown to be self-evident as the algorithm’ predictions returned the most accurate results. We found that a single algorithm cannot predict 100% accurate results. We assert that the accuracy of the proposed system can be enhanced by unifying the algorithm with other machine learning algorithms, such as Artificial Neural Network (ANN), Support Vector Machine (SVM).

**8.2 LIMITATIONS:**

**Dependency on GPS Signal:** The system heavily relies on GPS signal reception for accurate location tracking. In areas with poor GPS reception or signal interference, the system's reliability may be compromised, leading to inaccurate location data and arrival time predictions.

**Hardware Reliability:** The performance of the system is contingent upon the reliability of the hardware components, including the Arduino microcontroller, RFID reader, GPS receiver, and GPRS/GSM modem. Malfunctions or failures in any of these components could disrupt the tracking and communication process.

**Power Supply Requirements:** The hardware components, especially the GPS receiver and GPRS/GSM modem, require a stable power supply, typically 12V. In cases of power outages or insufficient power sources, the system's functionality may be affected, leading to data transmission delays or failures.

**Limited Coverage Area**: The system's effectiveness is limited to areas with sufficient network coverage for GSM/GPRS communication. In remote or rural areas with limited network infrastructure, the system may encounter difficulties in transmitting data to the cloud server, impacting real-time tracking and prediction accuracy.

**Data Latency:** Due to the reliance on GPRS/GSM communication for data transmission to the cloud server, latency issues may arise, especially during peak usage times or in areas with network congestion. This could result in delays in updating vehicle locations and arrival time predictions, reducing the system's responsiveness.

**Prediction Accuracy:** While the Kalman filtering algorithm is employed for arrival time prediction, it may not account for all variables affecting travel time, such as traffic conditions, road closures, or unforeseen delays. As a result, the system's predicted arrival times may not always align perfectly with actual arrival times, leading to discrepancies and potential dissatisfaction among users.

**Maintenance and Upkeep:** Regular maintenance and upkeep of the hardware components, as well as software updates for the Android application and cloud server, are essential to ensure the system's continued functionality and performance. Failure to maintain the system adequately could lead to decreased reliability and accuracy over time.

**Cost Implications:** While the system is described as cost-effective, there are still initial hardware and development costs associated with its implementation. Additionally, ongoing expenses for maintenance, data storage, and server usage may accrue, potentially posing financial constraints for organizations or institutions implementing the system.

**8.3 FUTURE ENHANCEMENT:**

* Another feature that could enhance the IoT-based school bus tracking and arrival time prediction system is a two-way communication system between parents and bus drivers.
* This feature would allow parents to communicate directly with bus drivers through a messaging interface available on the mobile app. Parents could use the app to send messages to the bus driver regarding any concerns, such as delays, changes in pickup/drop-off locations, or student absences. Similarly, bus drivers could communicate with parents to provide updates on the bus status, traffic conditions, or any other relevant information.
* Another feature that could enhance the IoT-based school bus tracking and arrival time prediction system is a real-time video streaming feature for parents.
* This feature would allow parents to access live video feeds from cameras installed on the school bus. Parents could view their child's journey in real-time, ensuring their safety and providing peace of mind.
* Another feature that could enhance the IoT-based school bus tracking and arrival time prediction system is a driver behavior monitoring and feedback system.
* This feature would involve installing sensors in the school bus to monitor various aspects of the driver's behavior, such as speed, braking, acceleration, and adherence to traffic rules. The system would provide real-time feedback to the driver, as well as generate reports for school administrators to review driver performance.

**Integration of Machine Learning Algorithms:** Incorporating advanced machine learning algorithms, such as Artificial Neural Networks (ANN) or Support Vector Machines (SVM), could improve the accuracy of arrival time predictions. These algorithms can analyze historical data patterns and factors influencing travel times, leading to more precise predictions.

**Dynamic Routing Optimization:** Implementing dynamic routing optimization algorithms can help optimize bus routes in real-time based on traffic conditions, road closures, and other factors. This enhancement can improve efficiency, reduce travel time, and enhance overall service quality.

**Mobile Application Features:** Enhancing the mobile application with additional features, such as real-time notifications for parents about bus delays or route changes, can improve user experience and engagement. Integration with school management systems for attendance tracking and student management could also be considered.

**Predictive Maintenance:** Implementing predictive maintenance techniques for hardware components can help anticipate and prevent potential failures. Utilizing sensor data and machine learning algorithms, the system can detect anomalies in hardware performance and schedule maintenance proactively, reducing downtime and ensuring continuous operation.

**Alternative Communication Protocols:** Exploring alternative communication protocols, such as LoRaWAN or NB-IoT, can provide more reliable and cost-effective communication options, especially in areas with limited cellular coverage. These protocols offer long-range and low-power capabilities, ideal for IoT applications like vehicle tracking.

**Crowdsourced Data Integration:** Leveraging crowdsourced data from commuters and other vehicles can enhance the accuracy of traffic and road condition information. Integrating such data sources into the prediction algorithms can provide real-time insights into traffic patterns and help adjust arrival time predictions accordingly.

**User Feedback Mechanism:** Implementing a user feedback mechanism within the mobile application can allow parents and stakeholders to provide feedback on the accuracy and reliability of arrival time predictions. This feedback can be used to continuously improve the system and address any issues or concerns raised by users.

**Environmental Impact Analysis:** Conducting an environmental impact analysis to assess the system's carbon footprint and energy consumption can help identify areas for optimization and sustainability improvements. Implementing energy-efficient hardware components and optimizing data transmission processes can minimize the system's environmental impact.

**9.REFERENCES**

**9.1 REFERENCE PAPERS:**

[1] S. Shah and B. Singh, “RFID Based School Bus Tracking and

Security System,” pp. 1481–1485, 2016.

[2] J. Zambada, R. Quintero, R. Isijara, R. Galeana, and L. Santillan, “An

IoT based scholar bus monitoring system,” 2015 IEEE 1st Int. Smart

Cities Conf. ISC2 2015, 2015.

[3] A. F. Abidin and M. Kolberg, “Towards Improved Vehicle Arrival

Time Prediction in Public Transportation: Integrating SUMO and

Kalman Filter Models,” Proc. - UKSim-AMSS 17th Int. Conf.

Comput. Model. Simulation, UKSim 2015, pp. 147–152, 2016.

[4] S. Lee, G. Tewolde, and J. Kwon, “Design and implementation of

vehicle tracking system using GPS/GSM/GPRS technology and

smartphone application,” 2014 IEEE World Forum Internet Things,

pp. 353–358, 2014.

[5] T. Bojan, U. Kumar, and V. Bojan, “Designing vehicle tracking

system - An open-source approach,” 2014 IEEE Int. Conf. Veh.

Electron. Safety, ICVES 2014, pp. 135–140, 201

[6] L. Singla, “GPS Based Bus Tracking System,” 2015.

[7] T. Liu, J. Ma, W. Guan, Y. Song, and H. Niu, “Bus Arrival Time

Prediction Based on the k-Nearest Neighbor Method,” 2012 Fifth Int.

Jt. Conf. Comput. Sci. Optim., pp. 480–483, 2012.

[8] A. Sai, S. Salim, P. K. Binu, and R. C. Jisha, “A Hadoop Based

Architecture Using Recursive Expectation Maximization Algorithm

for Effective and Foolproof Traffic Anomaly Detection and

Reporting,” vol. 10, no. 55, pp. 2101–2106, 2015.

[9] M. Prathilothamai, A. M. Sree Lakshmi, and D. Viswanthan, “Cost

effective road traffic prediction model using Apache spark,” Indian J.

Sci. Technol., vol. 9, no. 17, 2016.

[10] H. Yaling, “The Design of Monitoring System Based on GPRS,”

2016 Int. Conf. Robot. Intell. Syst., pp. 432–435, 2016.

[11] N. R. Nair, “2-D Airborne Vehicle Tracking Using Kalman Filter,”

2016.

**9.2 WEBSITE LINK:**

**LINK: https://ieeexplore.ieee.org/Xplore/home.jsp**