INTERNET OF THINGS (IoT) SMART VEHICLE TRACKING WITH MONITORING SYSTEM

COURSE TITLE: PROJECT & THESIS
COURSE CODE: ME400

NAME OF STUDENTS :

1. **MD ALIUR RAHMAN**
2. **MD. BIPUL HASAN**
3. **MD RAJU ISLAM**
4. **MD SALAUDDIN PIASH**
5. **MD SHAKIL HOSSAIN**

DEPARTMENT OF MECHANICAL ENGINEERING
SONARGAON UNIVERSITY (SU)
DHAKA- 1215, BANGLADESH.

JANUARY 2025.

INTERNET OF THINGS (IoT) SMART VEHICLE TRACKING WITH MONITORING SYSTEM

COURSE TITLE: PROJECT & THESIS
COURSE CODE: ME400

NAME OF STUDENTS:

1. **MD ALIUR RAHMAN** **ID ME 2102024287**
2. **MD. BIPUL HASAN** **ID ME 2102024322**
3. **MD RAJU ISLAM** **ID ME 2102024406**
4. **MD SALAUDDIN PIASH** **ID ME 2101023299**
5. **MD SHAKIL HOSSAIN** **ID ME2102024132**

A Graduation Exercise Submitted to the Department of Mechanical Engineering Partial Fulfillment of the Requirement for the Degree of Bachelor of Mechanical Engineering.

DEPARTMENT OF MECHANICAL ENGINEERING
SONARGAON UNIVERSITY (SU)
DHAKA- 1215, BANGLADESH.

JANUARY -2025

DECLERATION

We, hereby, declare that the work presented in this project is the outcome of the investigation and research work performed by us. We also declare that no part of this project and thesis has been or is being submitted elsewhere for the award of any degree.

.....
Md Aliur Rahman
ID: ME2102024287

.....
Md. Bipul Hasan
ID: ME2102024322

.....
Md Raju Islam
ID: ME2102024406

.....
Md Salauddin Piash
ID: ME2101023299

.....
MD Shakil Hossain
ID: ME2102024132

APPROVAL

This is to certify that the project on **Internet of think Smart Vehicle Tracking with Monitoring System** by Md Aliur Rahman (ID: ME2102024287), Md. Bipul Hasan (ID: ME2102024322), Md Raju Islam (ID: ME2102024406), Md Salauddin Piash (ID: ME2101023299), and MD Shakil Hossain (ID: ME2102024132) has been carried out under my supervision. The project has been carried out in partial fulfillment of the requirements of the degree of Bachelor of Science (B.Sc.) in Mechanical Engineering of years of 2024 and has been approved as to its style and contents.

.....

Md. Mostofa Hossain
Professor & Head
Department of Mechanical Engineering
Sonargaon University (SU)

ACKNOWLEDGEMENT

First of all, we are grateful to Allah, the almighty for giving us the courage and enthusiasm to complete the thesis work. The authors express their gratitude to **"Md**. **Mostofa Hossain sir"** for his constant & meticulous supervision, valuable suggestion and encouragement to carry out this work. For all this, the authors acknowledge their sincere gratitude to him. We are also grateful to all our thesis & project working team of SU for their help in construction of the project work and give their valuable knowledge and time for completing the experiment. Finally, we would like to thank everybody who supported us in any respect for the completion of the thesis.

Authors,

Md Aliur Rahman
Md Shakil Hossain
Md Raju Islam
Md. Bipul Hasan
Md Salauddin Piash

ABSTRACT

Without vehicles, life is unimaginable and the number of automobiles has also grown quickly to meet the needs of enormous population.

This paper introduces a smart vehicle tracking system using Internet of Things (IoT) technology. The system integrates GPS for real-time location tracking.

It allows users to monitor a vehicle's speed and location through a user-friendly IoT interface or mobile app.

The collected data is sent to the cloud via a NodeMCU microcontroller, enabling speed alerts and IoT fuel level control.

By reducing the response time in emergencies and controlling over speeding, the system aims to minimize accidents and save lives.

This innovative solution is particularly useful in densely populated areas like Bangladesh, where road safety is a major concern.

TABLE OF CONTENTS

Name of the Contents		Page No
Declaration of Candidates		ii
Board of Examiners		iii
Certificate		iv
Acknowledgement		v
Abstract		vi
Table of Contents		vii
List of Figure		viii
List of Table		ix
List of Tables		x
Chapter-1: Introduction		
1.1	Introduction	01
1.2	Problem Statement	02
1.3	Background of the Project	02
1.4	Methodology	03
1.5	Project Management	04
1.6	Objective of the Project	05
Chapter 2: Background and Motivation		
2.1	Introduction	06
2.2	Motivation for Project	07
2.3	Internet of Things (IoT) Basic Concept	08
2.8	Literature Survey	09
2.9	Time Plan	10
Chapter 3: The Design Methods and Procedure		
3.1	Introduction	12
3.2	Hardware Procedure	13
3.2.1	Block Diagram of This Project	14
3.2.2	Circuit Diagram of This Project	15
3.3	Software	16

3.3.1	Software Tools	17
3.3.2	Programming	18
3.3.3	Arduino Program Development	18
3.3.4	Android Apps	19
3.3.5	Proposed Project in Proteus 8.13	19
3.4	Over All Flow Chart	20
3.5	The list of Devices used in the Project is Given Below	20
3.6	Node MCU	20
3.6.1	Technical Specifications	21
3.7	Buck Module	21
3.8	Battery	22
3.8.1	Specifications	22
3.9	Motor Driver Module	23
3.9.1	Specifications	23
3.10	Gear Motor	23
3.11	GPS Module	24
3.12	Relay Module	25
3.14	Ultrasonic Sensor	27
3.14.1	Technical Specifications	28

Chapter 4: Result Analysis

4.1	Introduction	29
4.2	Result Analysis	30
4.2.1	Location Monitoring	30
4.2.2	Car Oil Level Monitoring	32

Chapter 5: Conclusion and Recommendation

5.1	Introduction	34
5.2	Advantage	34
5.3	Application	34
5.4	Limitation	35
5.5	Future Scope	35
5.6	Conclusion	35

References 36
Appendix 37

LIST OF FIGURES

Figure No	Figure Contains	Page No
Figure 1.1	Methodology	03
Figure 1.2	Project Management	04
Figure 2.1	IoT Basic Concept	08
Figure 3.1	Block Diagram	14
Figure 3.2	Circuit Diagram	15
Figure 3.3	Program Installation Process	18
Figure 3.4	Flowchart of the Compiling Process	18
Figure 3.5	Blynk Android Webapps	19
Figure 3.6	User Interface of Proteus 8.13	20
Figure 3.7	Node MCU	21
Figure 3.8	LM2596 Voltage Converter	22
Figure 3.9	Lipo Battery	22
Figure 3.10	Motor Driver Module	23
Figure 3.11	Gear Motor	24
Figure 3.12	GPS Module	24
Figure 3.13	Circuit Diagram of GPS Module	25
Figure 3.14	Relay Module	25
Figure 3.15	Stematic of Relay Module	26
Figure 3.16	Pinout of Relay Module	27
Figure 3.17	Ultrasonic Sensor	27
Figure 3.19	Project Picture	29
Figure 4.1	Location Monitoring System	32
Figure 4.2	Car Oil Level Monitoring	32

LIST OF TABLES

Table No	Table Contains	Page No
Table- 01	Time Scheduled Table for Project Introduction	10
Table- 02	Time Plan Table for Project Introduction	10
Table- 03	Time Scheduled Table for Project	11
Table- 04	Time Plan Table for Project	11

CHAPTER- 01

INTRODUCTION

1.1 Introduction

In urban areas accidents are most common phenomena where many of such accidents can be taken care easily but some accidents occur during night time when the visibility is quite low, during such cases it will be difficult for an ambulance driver to identify the accident spot with the help of phone calls made by the citizens. If the driving force knows the precise spot of the accident the time period between the spot and the hospital is going to be significantly reduced. The main objective of this paper is to help reduce the time factor in case of accidents. There are many cases where an accident occurs during the night and the person met with the accident is unconscious then it would take hours for someone to find out and inform the authorities about it. So saving such precious time will indeed save lives. In connection with this concept, an experimental setup is constructed that can detect accidents automatically without any human help.

In terms of innovation, the internet of Things (IoT) is seen as the internet's successor. All items may exchange data and communicate via data-detecting devices thanks to the Internet of Things (IoT), a sophisticated system that links them all to the Internet. It makes it possible to precisely recognize, discover, follow, observe, and supervise. It is a system that enhances communication between people, between people and things, and between things. Systems in the Internet of Things will be connected to a lot of things. According to a current paradigm of communication, in the not too distant future, commonplace items will be equipped with microcontrollers, electronic communication devices, and reasonable convention stacks that will allow them to communicate with one another and with customers, becoming a vital part of the Internet. Everyone relies heavily on vehicles and the road transportation network to get around in this densely populated mechanical world. Since the dawn of time, transportation has been an integral part of human development. The number of vehicles on the road increases in tandem with the population. When it comes to public transit, time and patience are crucial. In other words, many individuals who utilize public transit buses have wasted time because they have to wait at bus stops. Millions of students must travel from home to college and vice versa every day.

This article describes an intelligent vehicle tracking and speed monitoring system based on the Internet of Things. Our newly generated technology allows for the Internet based reporting of vehicle location information. It enables anybody to instantaneously check the speed and location of vehicle on the internet, hence reducing arrival and departure times. Using various sensors, the system monitors the location as well as speed and delivers real- time information. No devices are mentioned in a report that advises monitoring two tracking indicators. Creating a system that can calculate the speed of vehicle using or sensor using interrupt method and location tracking using the GPS system is proposed. This system is able to achieve its objectives by utilizing technologies such as the Internet of Things (IOT) and the Cloud. The core concept of the "internet of things" is that every device should be connected to the internet and any other relevant gadgets. The data generated by Internet of Things devices may be readily uploaded to the cloud and subsequently delivered across the internet to end

users. Vehicle monitoring is an essential use of the notion behind the Internet of Things. This entails sensing and collecting a information from vehicle and utilizing it for triggering alerts and interacting with users through user interface.

1.2 Problem Statement

In todays fast-paced world, managing and monitoring vehicles efficiently is a critical challenge for individuals and organizations. Traditional vehicle tracking systems often lack real-time updates, remote accessibility, and integration with modern IoT solutions. This leads to inefficiencies in fleet management, increased fuel costs, and heightened security risks such as vehicle theft.

Moreover, the absence of a comprehensive monitoring system makes it difficult to assess vehicle health, driving behavior, and environmental impact. Without actionable insights, users cannot optimize routes, reduce maintenance costs, or ensure the safety of passengers and goods. There is a pressing need for an IoT-enabled smart vehicle tracking and monitoring system that provides real-time location updates, vehicle diagnostics, and alerts through a user-friendly platform accessible via the internet. Such a system would empower users to make data-driven decisions, enhance vehicle safety, and contribute to more efficient and sustainable transportation practices.

1.3 Background of the Project

The Internet of Things (IoT) has revolutionized the way devices interact, enabling seamless communication and data sharing between physical objects and digital platforms. In the context of transportation, IoT technology has introduced innovative solutions for tracking and monitoring vehicles, addressing challenges in fleet management, personal vehicle security, and urban transportation systems.

Traditional vehicle tracking systems relied heavily on manual intervention and basic GPS technology, which often provided limited functionality, such as static location tracking. However, the integration of IoT has enhanced these systems by incorporating real-time data analytics, cloud computing, and advanced sensors. These advancements allow for continuous monitoring of a vehicle's location and fuel consumption.

Smart vehicle tracking systems have become increasingly important due to the growing need for efficient logistics, reduced operational costs, and heightened security measures. Industries such as logistics, public transportation, and personal vehicle owners are adopting these systems to improve operational efficiency and ensure the safety of assets. We are attached some other systems to this system, vehicle can be located over the Internet, how much oil is in the vehicles, vehicles, can be turned off if desired through the Internet. In this way, this system can speed up the rescue operation, minimize the casualties after the accident, and numerous important human lives can be spared.

Governments and organizations are also leveraging IoT-enabled vehicle tracking to support initiatives like smart cities, which aim to reduce traffic congestion, lower carbon emissions, and improve public safety. The rise of mobile and web-based applications has further

simplified access to these systems, making it easier for users to receive alerts, generate reports, and make informed decisions in real time.

The IoT Smart Vehicle Tracking with Monitoring System represents a modern solution that combines the power of IoT, GPS, and cloud technologies to address current transportation challenges while paving the way for future advancements in mobility and connectivity.

1.4 Methodology

A GPS tracking unit is a gadget that makes use of the Global Positioning System (GPS) to pinpoint the exact location of a person, vehicle, or other asset that it is linked to and to record the asset's position over time. A cellular (GPRS), radio, or satellite modem included within the tracking device can be used to transfer the recorded location data to a central location database or an internet-connected computer. This makes it possible to use GPS tracking software to analysis the track afterwards and display the asset's location against a map backdrop in real time. In several nations throughout the world, vehicle tracking systems are becoming more well-liked and prevalent. It offers a tones of benefits for users, but it benefits car users the most since it makes it simpler for them to track their vehicles. The smartphone has overtaken all other forms of communication as the most widely utilized technology available today.

Therefore, it is now evident from the aforementioned study how cell phones have become a significant and fundamental part of our modern lives. For this reason, this vehicle monitoring system is text message oriented so that we may take care of our own vehicles with just a single touch of our hands. With the use of an internet connection, a smart phone can track the whereabouts of our automobile in real time. This tracking system was created in such a way that customers could retrieve their vehicles using a simple and user-friendly interface.

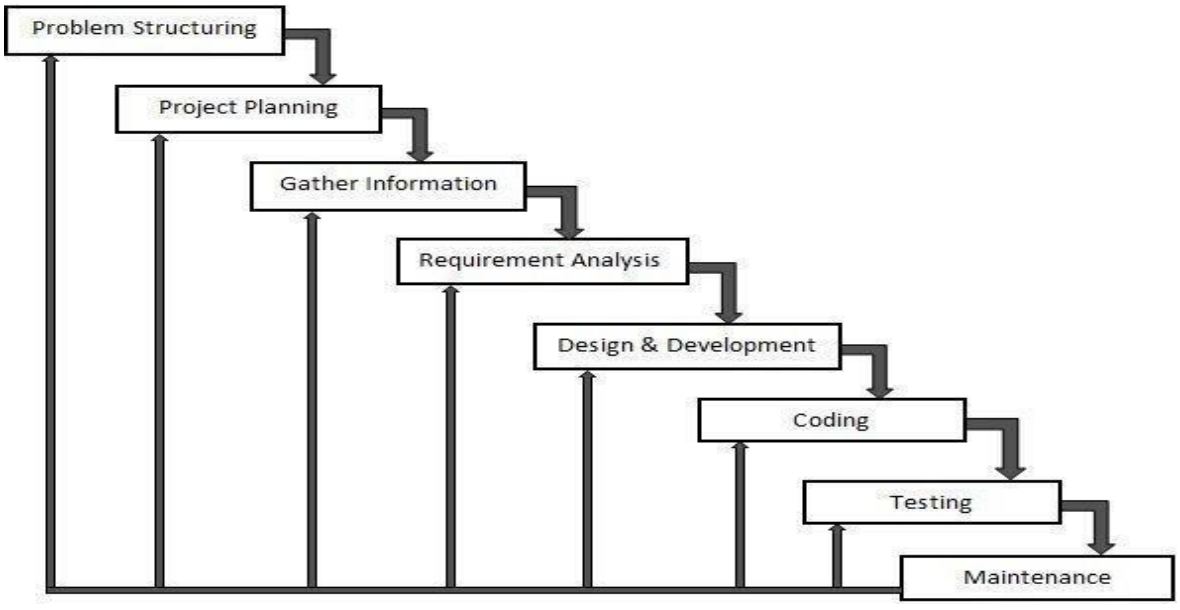


Fig- 1.1: Methodology

1.5 Project Management

Effective project management is essential for successfully developing and deploying the IoT Smart Vehicle Tracking with Monitoring System. The project management process involves systematic planning, execution, monitoring, and control to ensure the project's objectives are achieved within the defined scope, time, and budget constraints. Below is a comprehensive approach to managing this project:

1. Project Planning
2. Resource Allocation
3. Project Execution
4. Monitoring and Control
5. Project Closure

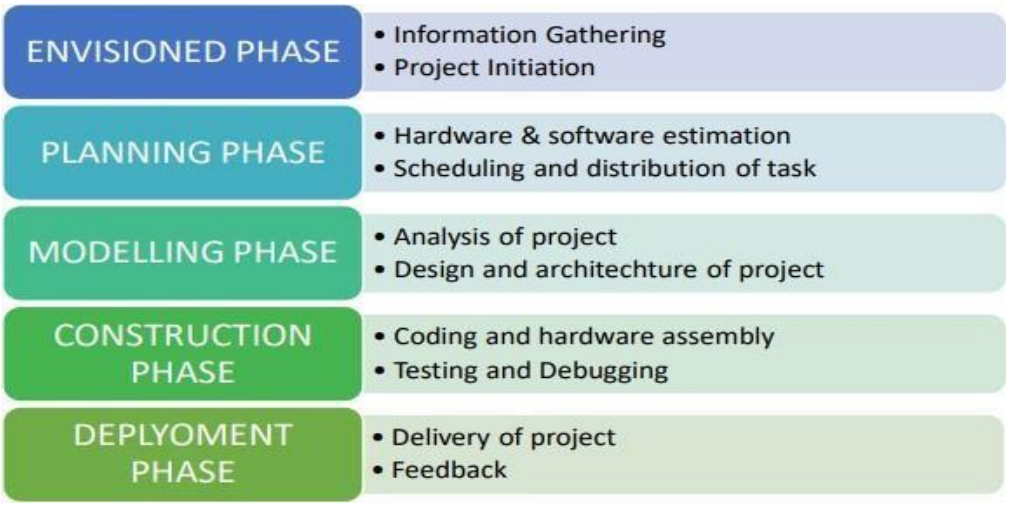


Fig :Project Management

1.5 OBJECTIVE OF THE PROJECT

To design IoT Smart Vehicle Tracking System is to enhance real-time tracking, monitoring, and management of vehicles using IoT technologies, ensuring accurate location updates, vehicle health monitoring, fuel level and improved safety**.**

- To design and implement IoT for smart vehicle GPS tracking System.
- IoT Monitoring System.
- IoT vehicle Controlling System.
- IoT fuel level control.

CHAPTER- 02

BACKGROUND & MOTIVATION

2.0 Introduction

Nowadays, the rate of accidents has increased rapidly. Due to employment, the usage of vehicles like cars, bikes has increased, and because of this reason accidents can happen due to over speed. People are going under risk because of their over speed, due to the unavailability of advanced techniques, the rate of accidents can't be decreased. Around 80 per cent of accidents contribute to the loss of many lives. Mostly, the growing countries are being targeted by the day to day road accidents. The major reason is the lack of infrastructure, lack of traffic control and accident management also there can be multiple causes of road accidents, some of them are, driver negligence due to drowsiness, driving while intoxicated, over speeding, etc. Out of all the developing countries, Bangladesh has been listed as the country with a higher number of accidents in urban areas accidents are most common phenomena where many of such accident can be taken easily but some accidents occur during night time when the visibility is quite low, during such cases it will be difficult to identify the accident spot. The most prominent reason for the loss of a life during an accident is the unavailability of immediate help that can save a person's life by a few seconds. The moment an accident has occurred, the life of all passengers travelling in the vehicle is at stake. It all depends on response time that can save their lives by a few minutes or seconds. According to the statistics, reducing accident delay time by even 1 minute can save 6 per cent of lives. Hence, this response time is very crucial, and it needs to be reduced or at least either improved to save their live Tracking and monitoring vehicle are coming vastly utilized based on Global Positioning System (GPS). In this paper a real time tracking system is proposed. The proposed framework would make great utilization of new innovation that basis on embedded board denotation Arduino Intel Galileo. This system acts on Global System for Mobile Communication (GSM), Global Positioning System (GPS) and General Packet Radio System (GPRS) which are utilized for vehicle tracking and monitoring. The Wi-Fi Module is applied which incorporates three technic to be specific GPS, GPRS and GSM. GPS gives the vehicle location coordinates, GPRS transmits these data to the server and finally the GSM transmits warning message to the vehicle owner phone. This project exhibits the evolution of the vehicle tracking system prototype which is used in the vehicle. In particular, the framework will use GPS to acquire a vehicle location coordinates and send it utilizing GSM modem to the owner phone and to the web server. After that, the browser can load the PHP webpage which uses Google maps to show the location in a real-time. To define the location accuracy of the suggested system, we compared the system proposed results with the different commercial GPS device .With the advancements in technology, automated vehicle tracking systems (VTS) have become very prominent. The implementation details of a VTS that can display the vehicle position on Google maps, is presented in this project. The GPS, GSM/GPRS modules controlled by an Arduino microcontroller are placed inside the vehicle. The vehicle position obtained from the GPS module is sent to a remote web server using GSM. The Arduino microcontroller board is used as a web server and it displays received vehicle coordinates on the Google maps. The vehicle position is updated every 10sec as the vehicle is moving and tracking. Google maps API is used to locate the vehicle in a map using the smart phone application. Developed a web application using PHP and MySQL for GPS tracking using low priced mobile phones. The system proposed in this paper helps in monitoring the vehicle location from anywhere in the world through the IoT based web server. Displaying the coordinates on Google maps enables the users to understand the location of vehicle easily.

.

.

2.1 Motivation for Project

The motivation behind developing an IoT Smart Vehicle Tracking with Monitoring System stems from the growing challenges in vehicle management, safety, and efficiency, coupled with the transformative potential of IoT technologies. Below are the key drivers for this project:

1. Need for Enhanced Vehicle Security
2. Demand for Efficient Fleet Management
3. Advancements in IoT Technology
4. Importance of Real-Time Data
5. Focus on Safety and Compliance
6. Environmental Responsibility
7. Addressing Urban Challeng

Internet of Things (IoT) Basic Concept

The intercommunication between gadget to gadget or AI associated through the web with inserted innovation frameworks utilizing remote sensors, actuators which is distantly controlled, screen and upgraded by the client for mechanization is alluded as Internet of things (IoT). Here the expression "Things" signifies actual gadgets, for example, chips, cameras, sensors and other such gadgets.

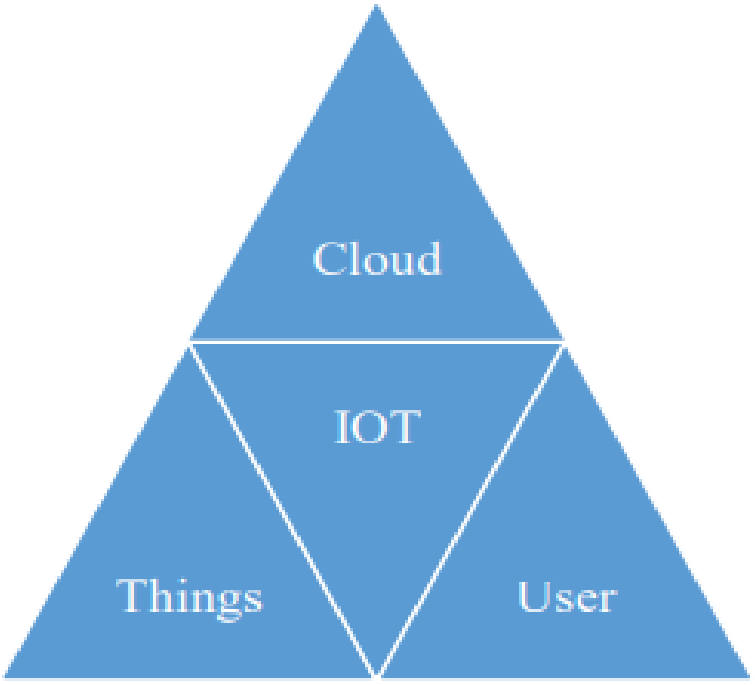


Fig-2.1: IoT Basic Concept

These actual gadgets are capable to impart, gather data and trade information by interfacing an organization. The implanted innovation of these actual gadgets makes this trade of data each other conceivable. There are an arrangement of modern robotization incorporates that can help making life at mechanical progressively invaluable and less difficult to administer, especially for involved, immense families. Assume you could therefore oversee and control the contraptions that you usually turn on and off every day. With a modern computerization structure, you can oversee apparatuses when you're out of the house and contemplating whether you made sure to kill light or not, savvy framework will be there to address the inquiry. The creating proximity of the Internet of Things in person's lives has made turn of events and progression in the clever mechanical space, empowering customers to relate their devices through the web to their telephones and tablets, and improve than at any other time benefits for nuclear families. Also, as house proprietor are changing their devices to one central application, device or focus, they further comprehend the value these modern robotization things can bring to a family. It is the network separated from the regular gadgets utilizing web, for example, work area, PCs, Smartphone, tablets and so on.

Literature Survey

In a survey, states that considering the spike rate in the crime regarding vehicles during transportation, and even the latest reports by insurance association of the Malaysian committee confirms approx. 60 vehicles are stolen every day. In our research work, we found that Arduino based vehicle tracking system using technologies like GPS and GSM, helps to display the exact coordinates and the current location of the vehicle. It also helps calculate distance travelled by the vehicle and stores in the database for further analysis. The data can be viewed in real time or on the demand. An algorithm which predicts the arrival and departure time is also used. [1]

In 2019, Mohammed Hazim Alkawaz put forth a vehicle tracking and reporting system using robust algorithm. In this system, raspberry pi was used. The issue of keeping track of stolen vehicle was solved in this system [2].

In 2020, Bernard Akindade Adarmola put forth vehicle tracking system using Arduino microcontroller. The GPSGSM based tracking system for vehicle was constructed based on optimum material consideration [3].

In 2019, Kunpeng Dai implemented Multiple Vehicle Tracking Based on Labelled Multiple Bernoulli Filter Using Pre-Clustered Laser Range Finder Data using Raspberry Pi as mainboard [4].

J.Pelegri, et.al. Proposed a vehicle speed monitoring system by using GMR magnetic sensors. Here a Microcontroller process the signal of sensors to obtain the speed and length of the car in real time [5].

A PC stores the data permanently, and after present it with a software. The signal transfer is based on the FSK modulation.

Rajesh Kannan et.al. Proposed a wireless sensor network for vehicle speed monitoring and traffic routing [6]. This system is monitored with wireless sensors in order to identify the speed of vehicles and also to regulate the traffic.

Ravi Kishore et.al. Proposed an over speed monitoring system [7]. This system calculates the speed and GPS coordinates which are able to find out the vehicle maximum speed in the respective area. If the speed of the vehicle exceeds the speed limit, the driver is alerted through a buzzer. If the driver still does not drive within the speed limit, a SMS will be sent to the concerned authorities.

G. Kirankumar et.al. Proposed a vehicle speed monitoring system using Rubee protocol [8]. This approach involves a fixing device called Rubee in all vehicles and by installing two receivers at a specific distance. Here receivers receive and store the license plate details. Ahmed Al-Gindy et.al. Proposed a RFID speed monitoring system which is able to identify the objects in certain range [9]. A vehicle speed can be approximated by travelling between two RFID readers. These RFID readers are placed at known and predetermined distances from each other.

Previous studies have demonstrated that various wireless technologies, including RFID, IR, GPS, Bluetooth, and Wi-Fi, manage and monitor smart public transportation systems.

This is done by merging information technology, cutting-edge methodologies, and smart sensing systems. A smart bus transportation system is developed and

Introduced by Parmeshwar et al. (2017) [10] using GSM, GPS, and Arduino. Passengers waiting for the next bus receive a notification from GSM and GPS. In this system, passengers were automatically counted. Using Wi-Fi and a smartphone application for real-time bus tracking, Kiran et al. (2017) [11] developed a smart bus tracking system. The programmer sends data to the cloud using the address of the bus terminal, where it is then retrieved and displayed to the user via a mobile app. The tracking system includes installing GPS, RTC, and Arduino UNO in a bus as well as an Android App that can be used to follow buses and determine the distances between stations along their routes [10].There is no mechanism that provides users with information on the location and speed of the bus.

2.1 Time Plan

The following tables define the main tasks in the project introduction and project itself.

T1	Project Definition	1 Week
T2	Collecting data	11 Weeks
T3	Analysis	7 Weeks
T4	Theoretical calculation	4 Weeks
T5	Documentation	10 Weeks
T6	Prepare for presentation	2 Weeks

Table- 01: Time Scheduled Table for Project Introduction

The time of the project introduction is scheduled over 16 weeks, table 2 shows how the work was scheduled over this time:

Week / Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
T1	■															
T2		■	■	■	■	■	■	■	■	■	■	■				
T3				■	■	■	■	■	■	■						
T4						■	■	■	■							
T5					■	■	■	■	■	■	■	■	■	■		
T6															■	■

Table-02: Time Plan Table for Project Introduction

The following table defines the main tasks in the project:

T1	Collecting data	3 Week
T2	Implementation	10 Weeks
T3	Analysis	5 Weeks
T4	Building and testing the system	8 Weeks
T5	Documentation	10 Weeks
T6	Prepare for presentation	2 Weeks

Table-03: Time Scheduled Table for Project

The time of the project is scheduled over 16 weeks, table 4 shows how the work was scheduled over this time:

Week / Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
T1	■	■	■													
T2			■	■	■	■	■	■	■	■	■	■				
T3				■	■	■	■	■								
T4							■	■	■	■	■	■	■	■		
T5					■	■	■	■	■	■	■	■	■	■		
T6															■	■

Table-04: Time Plan Table for Project

CHAPTER- 03

THE DESIGN METHODS AND PROCEDURES

3.1Introduction

As we know the usage of vehicles are increasing drastically, the hazards due to vehicles is also increasing. The main causes of accidents are high speed, drink and drive, diverting minds, over stress and due to electronic gadgets.

This project intentionally deals with accident which occurs due to carelessness of the person who is driving the vehicle. In most of the cases family members or the ambulance and police are not kept informed on time. This results in delaying in getting help.

Most of the time, we might not be able to locate the accident because no one knows where the accident will take place.
Without vehicles, life is unimaginable and the number of automobiles has also grown quickly to meet the needs of enormous population.
A smart bus speed and location tracking system has been developed in this work. IoT technology can be used to quickly locate arrival times, current bus positions, and bus routes on a map.
Google Maps and the Global Positioning System are used for navigation and display services. The speed of the vehicle can be tracked based on the rotations made by the vehicle wheels and the monitored data is also passed to the server.
So that the over speed indications is tracked and alerted to reduce speed of the vehicle and avoid accidents.
The Node MCU microcontroller is used to receive data from speed monitoring unit and global positioning system and passes that information to node MCU Wi- Fi module to transfer that information to IoT cloud. The user can use IoT cloud user interface or app to track vehicle and its speed.

3.1 Hardware Procedure

Designing a Vehicle Accident Detection and Location Monitoring System involves integrating various hardware components to detect accidents and track the location of the vehicles involved. Here's a generalized hardware procedure for such a project:

Microcontroller Unit (MCU): Choose a suitable microcontroller (MCU) like Arduino, Raspberry Pi, or a specialized microcontroller for embedded systems. The MCU will be the brain of the system, responsible for processing data from sensors and making decisions.

Accelerometer and Gyroscope: Integrate an accelerometer and gyroscope to detect sudden changes in acceleration and orientation. These sensors help in identifying abrupt movements or impacts that may indicate an accident.

GPS Module: Include a GPS module for location tracking. This module will provide real-time coordinates of the vehicle's position. Popular GPS modules include the NEO-6M or NEO- M8N.

Collision Detection Sensor: Use collision detection sensors or impact sensors to detect physical collisions or crashes. These sensors can be accelerometers, force-sensitive resistors (FSRs), or dedicated collision sensors.

Communication Module: Integrate a communication module like GSM, GPRS, or LTE for transmitting data to a central monitoring system. This module enables the system to send accident alerts and location information to emergency services or designated contacts.

Power Supply: Choose an appropriate power supply solution, such as a rechargeable battery or a vehicle's power system. Ensure the power supply is stable and has sufficient capacity to support continuous operation.

User Interface (Optional): Incorporate a user interface, such as an LED display or an LCD screen, to provide visual feedback or information to the driver.

Emergency Button (Optional): Include an emergency button that drivers can manually press in case of emergencies to trigger an immediate alert.

Enclosure and Mounting: Design a suitable enclosure to protect the electronics from environmental factors like dust, water, and vibration. Ensure secure mounting within the vehicle to prevent damage during normal operation and accidents.

Central Monitoring System (Optional): Develop a central monitoring system that receives and processes data from multiple vehicles. This can be a web-based application, mobile app, or a dedicated server that tracks and manages the entire fleet.

Testing and Calibration: Conduct thorough testing of the entire system to ensure accurate accident detection and reliable location tracking. Calibrate sensors and fine-tune algorithms for optimal performance.
Compliance with Regulations: Ensure that the system complies with relevant regulations and standards for vehicle safety and monitoring.
Keep in mind that specific requirements may vary based on the project's scope, intended use, and the type of vehicles involved. Additionally, it's essential to consider privacy and data security aspects when designing such systems.

3.1.1 Block Diagram of This Project

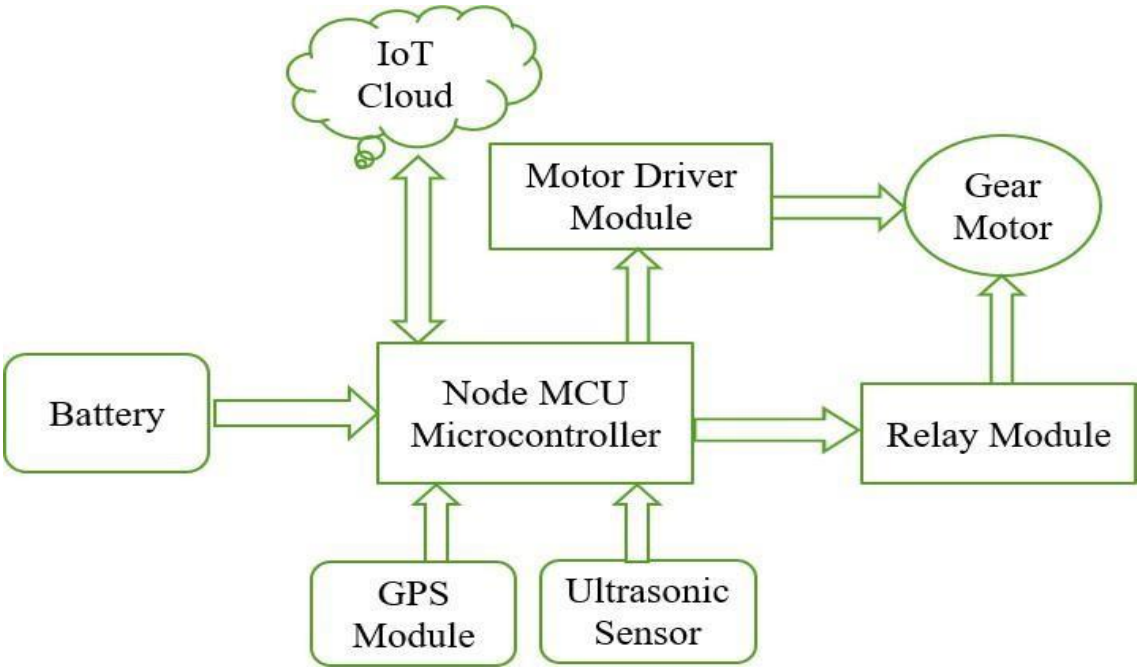


Fig- 3.1: Block Diagram

3.12 Circuit Diagram of This Project

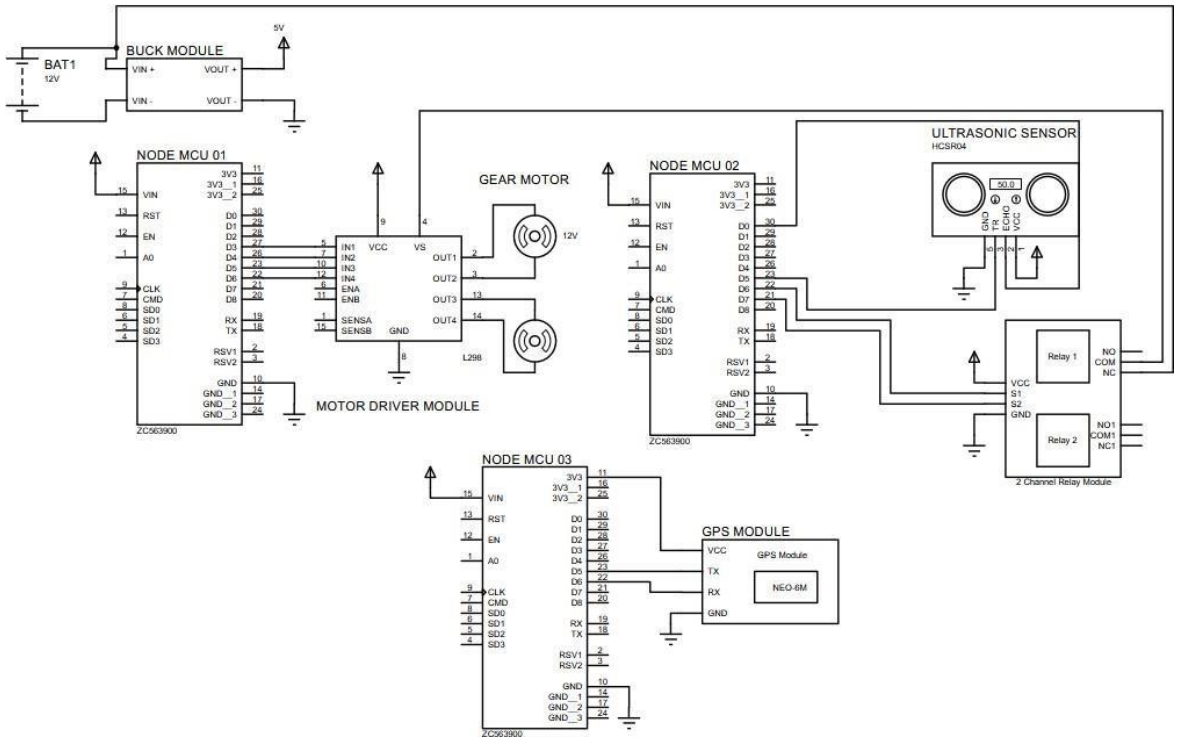


Fig- 3.2: Circuit Diagram

3.2 Software

Designing the software for a Vehicle Accident Detection and Location Monitoring System involves creating algorithms for accident detection, location tracking, and implementing a user interface for feedback and communication. Below is a generalized software procedure for such a project:

Algorithm for Accident Detection: Implement an algorithm that analyzes data from the accelerometer, gyroscope, or collision sensors to detect abrupt changes indicative of an accident. Set thresholds for acceleration, deceleration, or impact to trigger the accident detection.

GPS Data Processing: Develop a module for processing GPS data from the GPS module. Extract real-time coordinates and possibly other relevant information like speed and heading.

Integration of Sensors and Modules: Write code to integrate data from various sensors and modules, ensuring smooth communication and data flow between them. Implement error-checking mechanisms to handle sensor failures or communication issues.

Communication Module Implementation: Code the communication module to send accident alerts and location information to a central monitoring system or emergency contacts. Utilize protocols such as SMS, GPRS, or MQTT for communication.

User Interface Development: Create a user interface to provide feedback to the driver and display relevant information. Design a simple and intuitive interface for easy understanding, potentially using LEDs, LCD displays, or a mobile app.

Emergency Button Handling: Implement the functionality of the emergency button, ensuring that pressing it triggers an immediate alert and provides necessary information.

Central Monitoring System Integration: If applicable, develop the central monitoring system software. Design a system to receive and process data from multiple vehicles simultaneously.

Data Storage and Logging: Include mechanisms for storing accident data and location history. Log relevant information for later analysis or reporting.

Algorithm for False Positive/Negative Handling: Develop algorithms to reduce false positives and negatives in accident detection. Fine-tune thresholds and implement filtering mechanisms to improve accuracy.

Power Management: Implement power management features to optimize energy consumption and extend the system's battery life.

Testing and Validation: Conduct rigorous testing of the entire software system, including simulated accident scenarios and real-world testing. Validate the accuracy of accident detection and the reliability of location tracking.

Security and Privacy Considerations: Implement security measures to protect data during communication and storage. Ensure that the system complies with privacy regulations and guidelines.

Ventral Monitoring System Integration: If applicable, develop the central monitoring system software. Design a system to receive and process data from multiple vehicles simultaneously.
Data Storage and Logging: Include mechanisms for storing accident data and location history. Log relevant information for later analysis or reporting.
Algorithm for False Positive/Negative Handling: Develop algorithms to reduce false positives and negatives in accident detection. Fine-tune thresholds and implement filtering mechanisms to improve accuracy.
Power Management: Implement power management features to optimize energy consumption and extend the system's battery life.
Testing and Validation: Conduct rigorous testing of the entire software system, including simulated accident scenarios and real-world testing. Validate the accuracy of accident detection and the reliability of location tracking.
Security and Privacy Considerations: Implement security measures to protect data during communication and storage. Ensure that the system complies with privacy regulations and guidelines.
Documentation: Document the software architecture, algorithms, and user instructions for future reference and maintenance.
Deployment and Maintenance: Deploy the software onto the microcontroller or embedded platform. Plan for regular maintenance and updates as needed.
Compliance with Regulations: Ensure that the software complies with relevant regulations and standards for vehicle safety and monitoring.
Customize the software based on the specific requirements of the project, vehicle types, and intended users. Regular updates and improvements may be necessary to enhance the system's performance and address emerging challenges.

3.2.1 Software Tools

The software that is used to program the microcontroller is open-source-software and can be downloaded for free on www.arduino.cc. With this ―Arduino software‖ we can write little programs with the microcontroller. These programs are called ―Sketch‖. In the end the sketches are transferred to the microcontroller by USB cable. More on that later on the subject programming‖.

Installation Now one after another the Arduino software and the USB driver for the board have to be installed.

Installation and setup of the Arduino software

1. We have downloaded the Arduino software from www.arduino.cc and installed it on the computer (This was NOT connected to the PC). After that we opened the software file and installed the program named arduino.exe.

Two set ups on the program are important and should be considered.

a) The board that we want to connect has to be selected on the Arduino software. The Arduino Uno‖ is here known as ―Arduino / Genuine Uno.

b) We have to choose the right ―Serial-Port‖, to let the Computer know to which port the board has been connected. That is only possible if the USB driver has been installed correctly. It can be checked this way:

At the moment the Arduino wasn't connected to the PC. If we now choose ―Port‖, under the field ―Tool‖, we will already see one or more ports here (COM1/ COM2/ COM3…). The quantity of the shown ports doesn't depend on the quantity of the USB ports on the computer. When the board gets connected to the computer, we will find one more port.

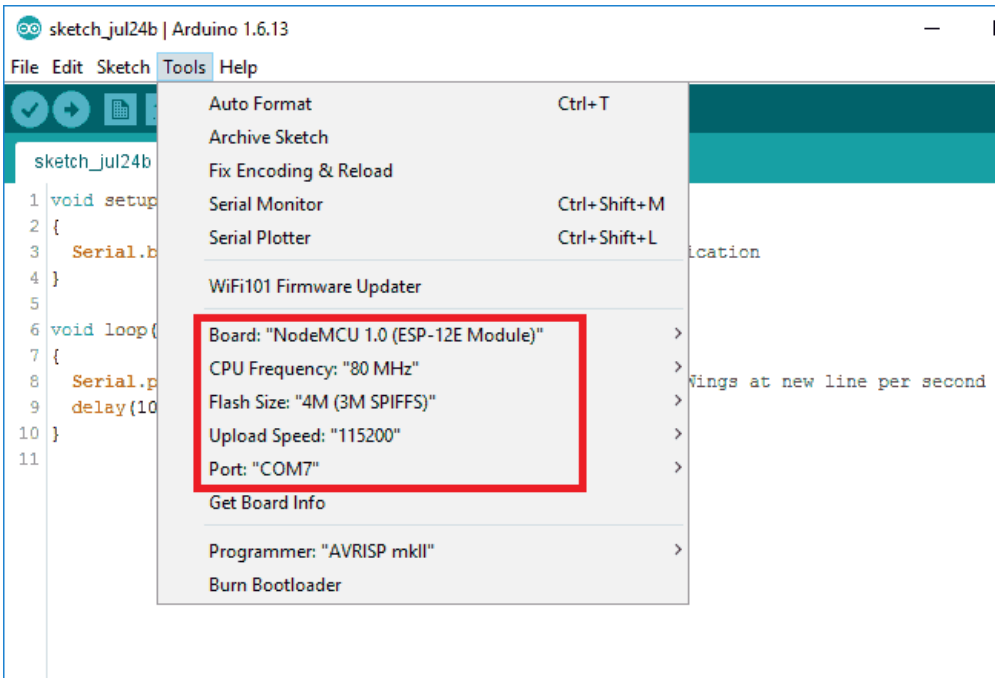


Fig- 3.3: Program Installation Process

3.2.2 Programming

The development cycle is divided into 4 phases:

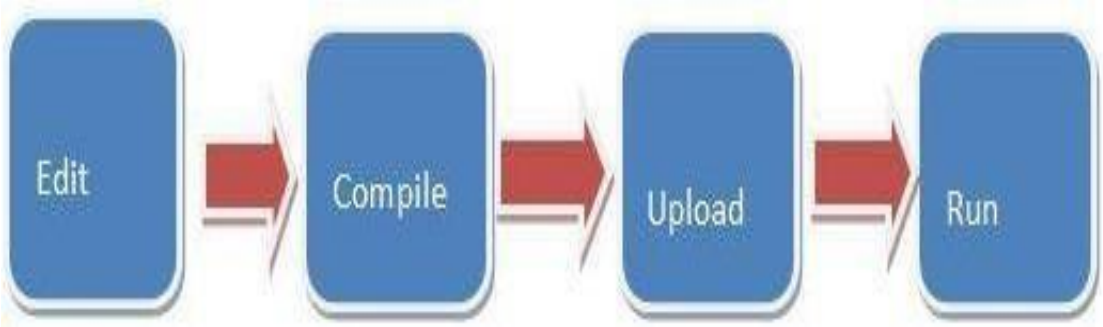


Fig- 3.4: Flowchart of the Compiling Process

Compile: Compile means to translate the sketch into machine language, also known as object.
Code Run: Arduino sketch is executed as soon as terminates the step of uploading on the board

.

3.2.3 Arduino Program Development

- Based on C++ without 80% of the instructions.
- A handful of new commands.
- Programs are called 'sketches'.
- Sketches need two functions:
- void setup ( )
- Void loop ( )
- Setup ( ) runs first and once.
- loop ( ) runs over and over, until power is lost or a new sketch is loaded.

3.2.4 Android Apps

The Blynk software is mainly operated over the Internet. This software has a specific server for sending data. This software has to be registered on the specified server, then login to the software with user name and password. Data is passed to the server through the controller's Internet head, the server sends software data, which we can monitor from anywhere.



Fig- 3.5: Blynk Android Web apps

3.2.5 Proposed Project in Proteus 8.13

Proteus 8 is best simulation software for various designs with microcontroller. It is mainly popular because of availability of almost all microcontrollers in it. So it is a handy tool to test programs and embedded designs for electronics hobbyist. You can simulate your programming of microcontroller in Proteus 8 Simulation Software. After simulating your circuit in Proteus 8 Software you can directly make PCB design with it so it could be a all in one package for students and hobbyists. So I think now you have a little bit idea about what is Proteus software.

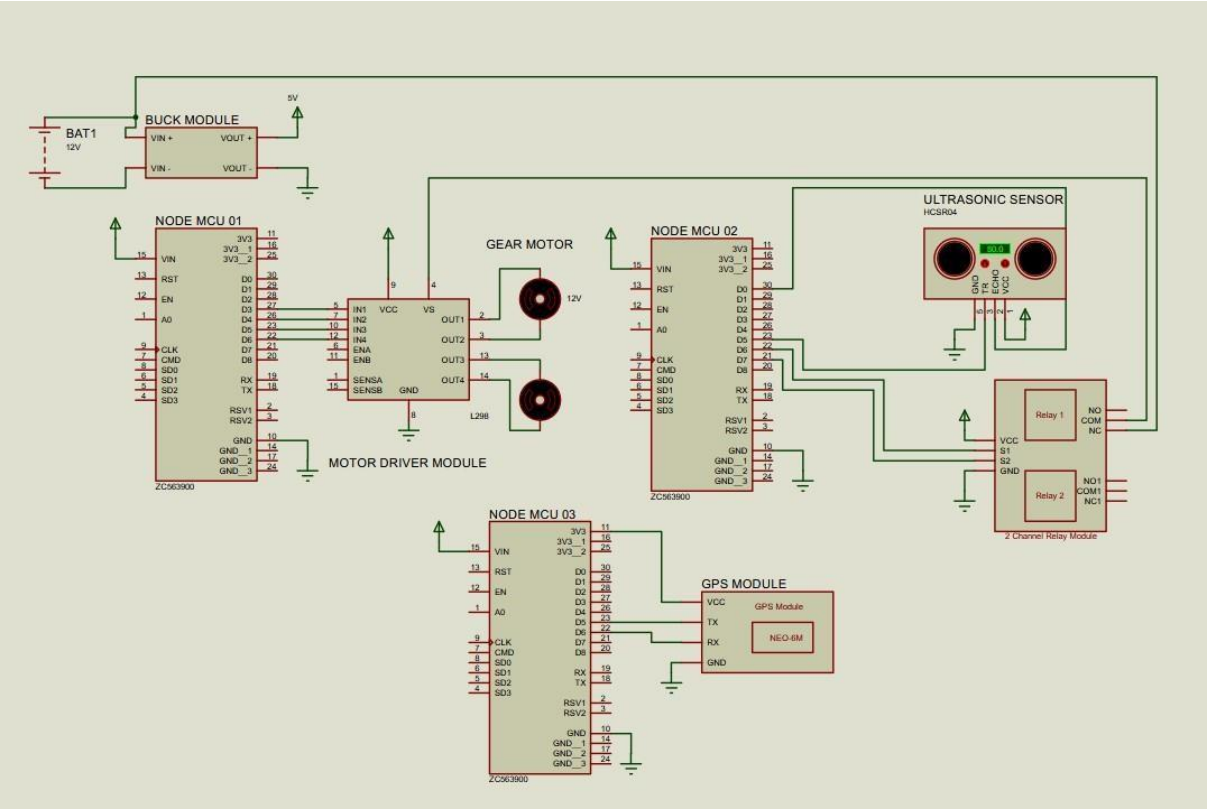


Fig- 3.6: User Interface of Proteus 8.13

3.5 The list of Devices used in the Project is Given Below

- Node MCU.
- Buck Module.
- Battery.
- LCD Display.
- Motor Driver Module.
- Gear Motor
- GPS Module.
- Relay Module.
- Ultrasonic Sensor.

3.6 Node MCU

Node MCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espress if Systems, and hardware which is based on the ESP-12 module. The term "Node MCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the e Lua project, and built on the Espress if Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS.

Node MCU was created shortly after the ESP8266 came out. On December 30, 2013, Espress if Systems began production of the ESP8266.The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core,[citation needed] widely used in IoT applications (see related projects). Node MCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open- hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9.

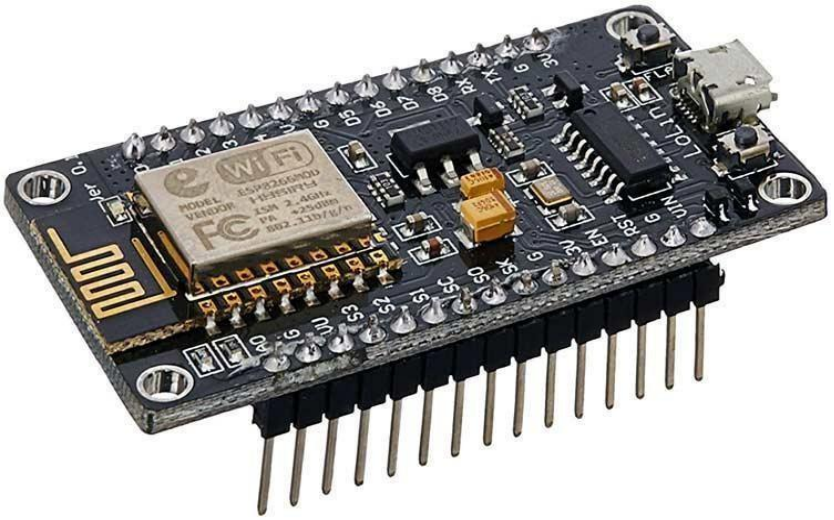


Fig- 3.8: Node MCU

3.6.1 Technical Specifications

- Power input: 4.5V ~ 9V (10VMAX), USB-powered
- Transfer rate: 110-460800bps
- Support UART / GPIO data communication interface
- Support Smart Link Smart Networking
- Working temperature: -40°C ~ + 125°C
- Drive Type: Dual high-power H-bridge
- Don't need to download resetting
- A great set of tools to develop ESP8266
- Flash size: 4MByte
- Lowest cost WI-FI

3.7 Buck Module

The LM2596 series of regulators are monolithic integrated circuits that provide all the active functions for a step-down (buck) switching regulator, capable of driving a 3-A load with



Fig- 3.9: LM2596 Voltage Converter

Excellent line and load regulation. These devices are available in fixed output voltages of 3.3 V, 5 V, 12 V, and an adjustable output version. Requiring a minimum number of external components, these regulators are simple to use and include internal frequency compensation, and a fixed frequency oscillator. The LM2596 series operates at a switching frequency of 150 kHz, thus allowing smaller sized filter components than what would be required with lower frequency switching regulators. Available in a standard 7-pin TO-220 package with several different lead bend options, and a 7-pin TO-263 surface mount package.

3.8 Battery

Lipo Battery 1100mAh 11.1V 3S Lithium polymer battery Pack (LiPo) battery are equipped with heavy duty discharge leads to minimize resistance and sustain high current loads. This Battery stand up to the punishing extremes of aerobatic flight and RC vehicles. Each pack is equipped with gold plated connectors and JST-XH style balance connectors.



Fig-3.10: Lipo Battery

3.8.1 Specification
3.8.2 Specification

- battery Capacity: 1100mah
- battery Voltage: 11.1 Volt
- Battery Size: 62*32*25 mm
- Battery Weight: 84 Gram
- Max continuous discharge: 25C

3.9 Motor Driver Module

The L298 is an integrated monolithic circuit in a 15-lead Multi watt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage. Check here for L298 datasheet.

3.9.1 Specifications

- Driver: L298
- Driver power supply: +5V~+46V
- Driver peak current: 2A
- Logic power output Vss: +5~+7V (internal supply +5V)
- Controlling level: Low -0.3V~1.5V, high: 2.3V~Vss
- Enable signal level: Low -0.3V~1.5V, high: 2.3V~Vss
- Max drive power: 25W (Temperature 75 ℃)
- Working temperature: -25℃~+130℃

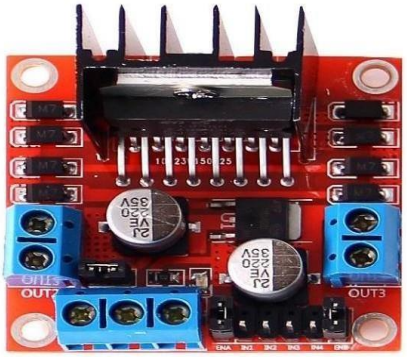


Fig- 3.11: Motor Driver Module

3.10 Gear Motor

A gear motor is a specific type of electrical motor that is designed to produce high torque while maintaining a low horsepower, or low speed, motor output. Gear motors can be found in many different applications, and are probably used in many devices in your home. Gear motors are commonly used in devices such as can openers, garage door openers, washing machine time control knobs and even electric alarm clocks. Common commercial applications of a gear motor include hospital beds, commercial jacks, cranes and many other applications that are too many to list.



Fig- 3.12: Gear Motor

3.11 GPS Module

This article demonstrates how to play with the u-blox NEO-6M global positioning system (GPS) module, a very popular, cost-effective, high-performance GPS module with a ceramic patch antenna, an on-board memory chip, and a backup battery that can be conveniently integrated with a broad range of microcontrollers. Nowadays, two NEO-6M GPS modules are extremely popular — the GY-GPS6MV2 and the GY-GPSV3-NEO.



Fig- 3.13: GPS Module

The u-blox NEO-6M GPS engine on these modules is quite a good one, and it also has high sensitivity for indoor applications. Furthermore, there one MS621FE-compatible rechargeable battery for backup and EEPROM for storing configuration settings. The module works well with a DC input in the 3.3- to 5-V range (thanks to its built-in voltage regulator).
As indicated, the GPS modules are based on the u-blox NEO-6M GPS engine. The type number of the NEO-6M is NEO-6M-0-001, and its ROM/FLASH version is ROM 7.0.3 (PCN reference UBX-TN-11047-1). The NEO-6M module includes one configurable UART interface for serial communication, but the default UART (TTL) baud rate here is 9,600. Because the GPS signal is right-hand circular-polarized (RHCP), the style of the GPS antenna will be different from the common whip antennas used for linear polarized signals.
The most Popular antenna type is the patch antenna. Patch antennas are flat, generally have a ceramic and metal body, and are mounted on a metal base plate. They are often cast in a housing. For more information about u-blox reference designs, see their website. Remember, the position of the antenna mounting is very crucial for optimal performance of the GPS receiver.

When using the patch antenna, it should be oriented parallel to the geographic horizon. The antenna must have full view of the sky, ensuring a direct line of sight with as many visible satellites as possible. The original circuit diagram of the module, borrowed from the web, is shown below:

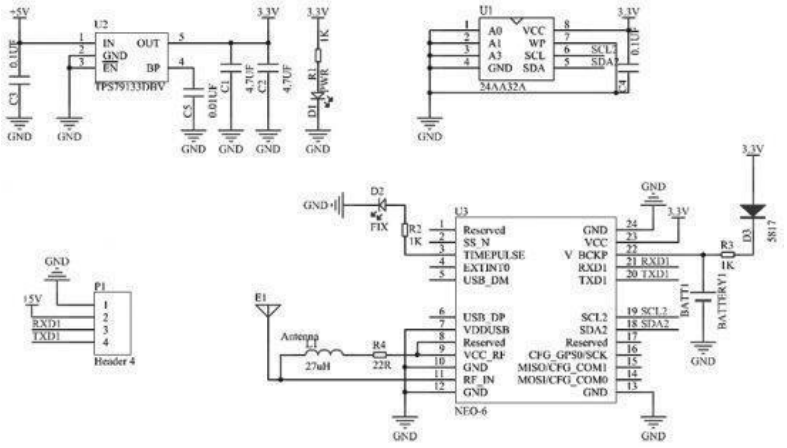


Fig- 3.14: Circuit Diagram of GPS Module

3.12 Relay Module

A relay is an electrically operated device. It has a control system and (also called input circuit or input contactor) and controlled system (also called output circuit or output cont actor). It is frequently used in automatic control circuit. To put it simply, it is an automatic switch to controlling a high-current circuit with a low-current signal.



Fig- 3.15: Relay Module

The advantages of a relay lie in its lower inertia of the moving, stability, long-term reliability and small volume. It is widely adopted in devices of power protection, automation technology, sport, remote control, reconnaissance and communication, as well as in devices of electro mechanics and power electronics. Generally speaking, a relay contains an induction part which can reflect input variable like current, voltage, power, resistance, frequency,

Temperature, pressure, speed and light etc. It also contains an actuator module (output) which can energize or de-energize the connection of controlled circuit. There is an intermediary part between input part and output part that is used to coupling and isolate input current, as well as actuate the output. When the rated value of input (voltage, current and temperature etc.) is above the critical value, the controlled output circuit of relay will be energized or de- energized.
NB: input into a relay can be divided into two categories: electrical quantities (including current, voltage, frequency, power etc.) and non- electrical quantities (including temperature, pressure, speed, etc.)
Features:
The features of 1-Channel Relay module are as follow:

1) Good in safety. In power system and high voltage system, the lower current can control the higher one.
2) 1-channel high voltage system output, meeting the needs of single channel control
3) Wide range of controllable voltage.
4) Being able to control high load current, which can reach 240V, 10A

5) With a normally-open (NO) contact and a normally-closed (NC) contacts
Interface specifications:
The output contacts of a relay (including NO, NC, and the common port) works as a SPDT – Single Pole Double Throw switch. Its operating principle is as follow: VCC --- 5V, GND----for ground
IN1 connects to the control valve which output 3V-5V
Output contacts: connect to applications
Interface Connecting and Setting:

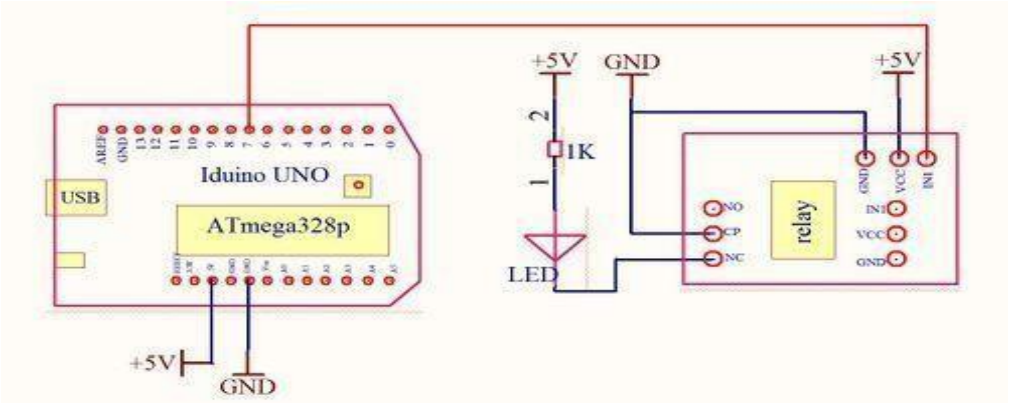


Fig- 3.16: Stematic Of Relay Module

Get Started: Firmware resources: Arduino board (any versions), wires, LED, 5v power supply. Software resource: Arduino IDE The one-channel relay can be programmed to realize the open and close automatically. NB: customers can use any software or firmware to control the module as long as the IN1 of which can input a voltage of 3V-5V.



Fig- 3.17: Pinout of Relay Module

We can do further development with the development tool you like as well as test it in the way of testing firmware. Firmware test: after the connection as in picture1-4, pay attention to the blink of LED, listen to the flicker of relay when it is working. After the upload of code, connect the IN1 pins to the 7th pin of Arduino board separately, you can hear the ticktack of the relay (it keeps opening and closing), in addition, the LED-G is blinking every 2s.

3.14 Ultrasonic Sensor

Ultrasonic transducers and ultrasonic sensors are devices that generate or sense ultrasound energy. They can be divided into three broad categories: transmitters, receivers and transceivers. Transmitters convert electrical signals into ultrasound, receivers convert ultrasound into electrical signals, and transceivers can both transmit and receive ultrasound.
This HC-SR04-Ultrasonic Range Finder is a very popular sensor that is found in many applications where it requires measuring distance and detecting objects.
The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The HC-SR04 ultrasonic sensor uses sonar to determine the distance to an object like bats or dolphins do.

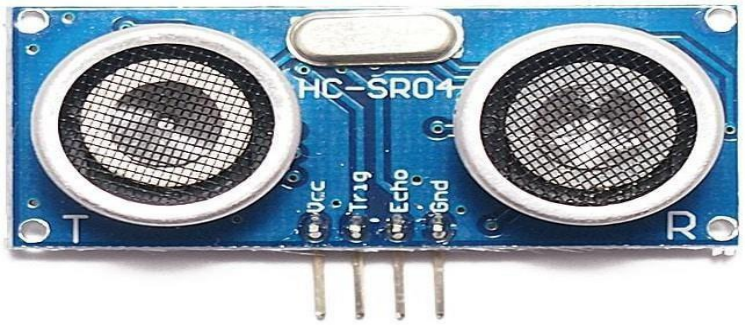


Fig- 3.19: Ultrasonic Sensor

This Ultrasonic Sensor module is a transmitter, a receiver, and a control circuit in one single pack!! It has very handy and compact construction. It offers excellent range accuracy and stable readings in an easy-to-use package. Its operation is not affected by sunlight or black material like Sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect).
The Trigger and the Echo pins are the I/O pins of this module and hence they can be connected to the I/O pins of the microcontroller/Arduino. When the receiver detects the return wave the Echo pin goes high for a particular amount of time which will be equal to the time taken for the wave to return to the sensor.

3.14.1 Technical Specifications

- Operating Voltage: 5 V
- Sonar Sensing Range: 2-400 cm
- Max. Sensing Range: 450 cm
- Frequency: 40 kHz

CHAPTER- 04

RESULT ANALYSIS

4.1 Introduction

The Internet of Things (IoT) has emerged as a transformative technology, enabling real-time connectivity and interaction between devices to create intelligent systems. In the field of transportation, IoT applications have significantly improved vehicle tracking, fleet management, and safety monitoring, addressing long-standing challenges faced by individuals and organizations.

The IoT Smart Vehicle Tracking with Monitoring System is an innovative solution designed to track the real-time location of vehicles and monitor their performance parameters. By integrating IoT devices such as GPS modules, sensors, and cloud computing, the system provides a comprehensive platform for vehicle management. It offers real-time data analytics, actionable insights, and instant alerts, empowering users to make informed decisions and ensure safety, efficiency, and security.

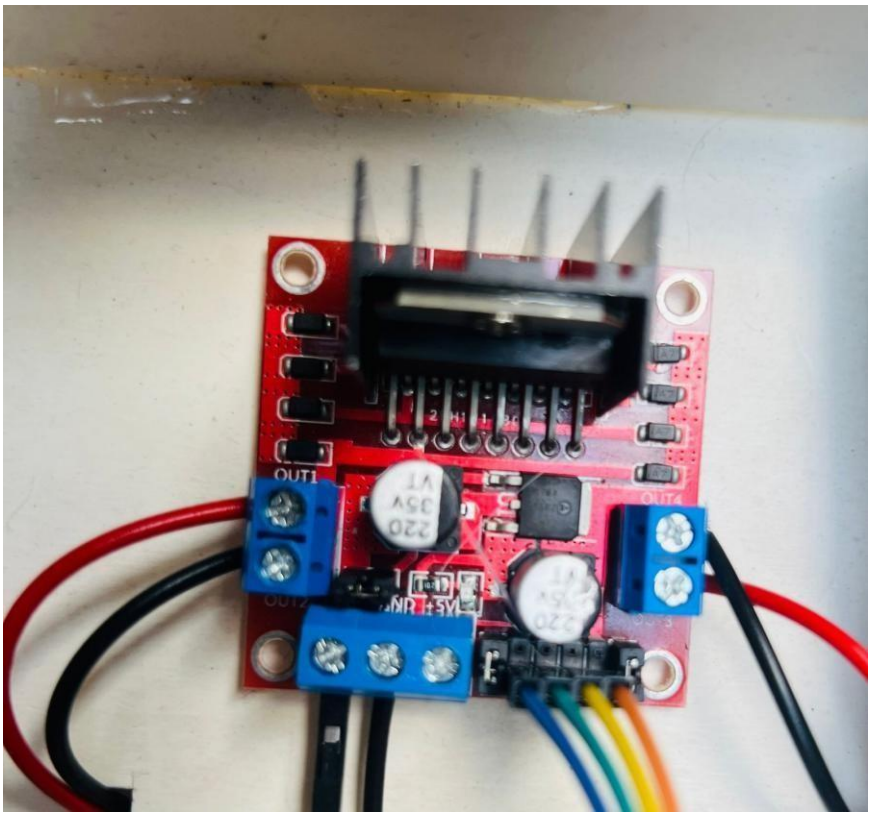


Fig- 4.1: Project Picture

4.2 Result Analysis

- Overview of smart vehicle tracking systems with GPS and Internet of Things (IoT)
- IoT enables real-time communication and tracking.
- Traditional systems lack advanced features like Internet of
- Things (IoT) integration.
- Real-time location and speed monitoring with data analytics.
- Improves logistics efficiency and reduces costs.
- Includes machine learning for autopilot navigation.
- Reduces accident response time, saving lives.
- Monitors fuel levels and other vehicle parameters.
- Allows remote control of vehicles via the internet.
- Applicable in logistics, public transport, and personal use.
- Optimized for low power consumption to extend battery life.

4.2.1 Location Monitoring

The location monitoring aspect of a Vehicle Accident Detection System involves tracking the real-time GPS coordinates of vehicles and reporting this information to a central monitoring system or other relevant entities. Here's an overview of the key components and steps involved in the location monitoring of such a system:

GPS Module Integration: Incorporate a GPS module into the hardware of the vehicle accident detection system. This module will continuously capture the vehicle's current location, speed, and heading.

Data Reading and Processing: Develop software routines to read and process the data received from the GPS module. Extract relevant information such as latitude, longitude, speed, and timestamp.

Real-time Location Updates: Implement a mechanism to provide real-time updates of the vehicle's location. This may involve periodic transmissions or event-triggered updates based on changes in the vehicle's status.

Communication Module: Integrate a communication module (e.g., GSM, GPRS, or LTE) to transmit the location data to a central monitoring system or designated recipients. Utilize secure communication protocols to protect the data during transmission.

Central Monitoring System: Develop or integrate a central monitoring system that receives and processes location data from multiple vehicles simultaneously. The central system may store the data, visualize it on a map, and facilitate real-time monitoring of the entire fleet.

Data Logging: Implement a data logging mechanism to record historical location data for each vehicle. This data can be valuable for post-accident analysis, route optimization, and other insights.

Alerts and Notifications: Set up alerts or notifications based on predefined criteria, such as the vehicle leaving a designated area or excessive speed. Ensure that the central monitoring system is capable of generating and transmitting alerts in real-time.

User Interface: Develop a user interface for the central monitoring system that displays the real-time location of each vehicle on a map.

Include features for querying historical data, tracking routes, and viewing vehicle-specific details.

Geofencing (Optional): Implement geofencing capabilities to define virtual boundaries for vehicles. Trigger alerts when a vehicle enters or exits a geofenced area.

Power Management: Optimize power management to ensure continuous operation without draining the vehicle's battery excessively.

Consider sleep modes or other energy-efficient strategies.

Privacy Considerations: Implement privacy features to ensure that location data is handled in compliance with relevant regulations.

Consider user consent mechanisms and data anonymization practices.

Testing and Validation: Conduct thorough testing to validate the accuracy and reliability of the location monitoring system. Test under various scenarios, including different driving conditions and environmental factors.

Integration with Accident Detection: Integrate the location monitoring system with the accident detection module to provide comprehensive information in the event of an accident.

Documentation: Document the implementation details, including software architecture, communication protocols, and user instructions.

Continuous monitoring and periodic updates to the system are essential to ensure its effectiveness over time. Regular maintenance and software updates may be necessary to address evolving requirements and improve system performance.



Fig- 4.2: Location Monitoring System

4.2.1 Car Oil Level Monitoring

Adding a car oil level monitoring feature to a Vehicle Accident Detection System involves incorporating sensors to measure oil levels, implementing algorithms for monitoring and alerting, and integrating the functionality with the overall system. Below is an outline of the steps involved in this specific aspect of the project?

Selecting Oil Level Sensor: Choose an appropriate oil level sensor. Common types include resistive sensors, capacitive sensors, or ultrasonic sensors. Ensure that the sensor is suitable for automotive applications and can withstand the operating conditions.

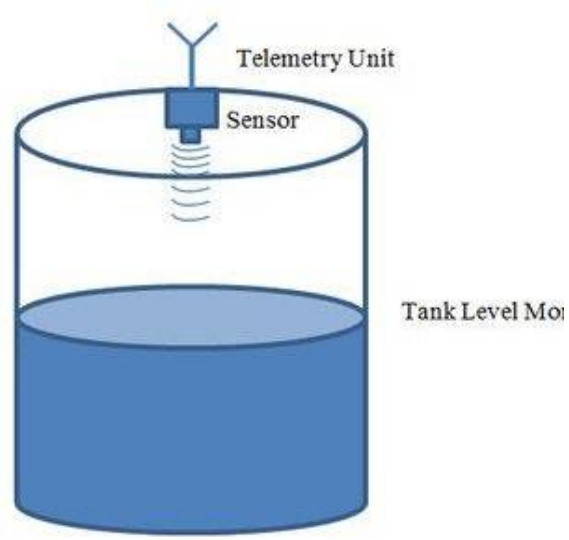


Fig- 4.3: Car Oil Level Monitoring

Integrating Oil Level Sensor: Connect the chosen oil level sensor to the vehicle's engine or oil reservoir. Ensure proper electrical connections and physical mounting.

Sensor Calibration: Calibrate the oil level sensor to provide accurate readings. Establish a baseline for normal oil levels and understand the sensor's behavior under various conditions.

Microcontroller Integration: Connect the oil level sensor to the microcontroller used in the Vehicle Accident Detection System.

Implement code to read data from the oil level sensor and process it within the microcontroller.
Oil Level Monitoring Algorithm: Develop an algorithm to monitor the oil level. This algorithm should compare the real-time oil level readings with predefined thresholds to determine if the oil level is within a safe range.
Thresholds and Alerts: Define thresholds for acceptable oil levels and critical levels that require immediate attention.
Implement alerts or notifications when the oil level drops below the acceptable threshold.
Integration with Central Monitoring System:
If applicable, integrate the oil level monitoring functionality with the central monitoring system used for the Vehicle Accident Detection System. Ensure that oil level data can be transmitted along with other vehicle information
Than we can say we were able to build our Device.
User Interface Updates: Update the user interface of the central monitoring system to display oil level information for each vehicle in the fleet. Implement visual indicators or alerts to notify users of low oil levels.
Communication Module Integration: If the Vehicle Accident Detection System uses a communication module for transmitting data, ensure that oil level data is included in the transmitted information.
User Interface Updates: Update the user interface of the central monitoring system to display oil level information for each vehicle in the fleet. Implement visual indicators or alerts to notify users of low oil levels. Communication Module Integration: If the Vehicle Accident Detection System uses a communication module for transmitting data, ensure that oil level data is included in the transmitted information Power Management: Optimize power management for the oil level monitoring system to minimize energy consumption and avoid unnecessary battery drain.
Testing: Conduct thorough testing of the oil level monitoring system under various conditions, including normal driving scenarios and simulated low oil level situations. Verify the accuracy of the oil level readings and the reliability of the alerting mechanism.
Documentation: Document the integration process, calibration procedures, and algorithms implemented for oil level monitoring.
Provide user instructions for maintenance and troubleshooting.
Compliance and Safety Considerations: Ensure that the oil level monitoring system complies with relevant safety standards and regulations. Consider safety features, such as automatic system shutdown in the event of critical oil levels.
User Education: Educate users, such as drivers or fleet managers, about the importance of monitoring oil levels and responding promptly to alerts.
By integrating oil level monitoring into the Vehicle Accident Detection System, you enhance the overall safety and maintenance capabilities of the vehicle fleet. Regular monitoring and preventive maintenance based on oil level data can contribute to the overall efficiency and longevity of the vehicles.

.

CHAPTER- 05

CONCLUSION AND RECOMMENDATION

5.1 Introduction

The rapid advancements in technology have paved the way for innovative solutions that address challenges in vehicle management, security, and efficiency. Among these, the Internet of Things (IoT) has revolutionized the transportation industry by enabling real-time communication and interaction between vehicles, sensors, and users.

The IoT Smart Vehicle Tracking with Monitoring System is designed to provide an intelligent and comprehensive approach to managing vehicles. This system leverages IoT-enabled devices such as GPS trackers, sensors, and cloud platforms to track vehicle locations, monitor critical parameters, and ensure the safety and efficiency of transportation operations. Through real-time updates, data analytics, and alerts, the system offers users valuable insights to optimize routes, maintain vehicle health, and enhance security.

5.2 Advantage

- We can monitor the oil tank of the vehicle.
- We can find the location of the vehicle.
- IoT monitoring and controlling system.
- Vehicle on the internet, hence reducing arrival and departure times.
- Easy to control.
- Low cost.
- It can there be easily commercialized.

5.3 Disadvantage

- Requires a strong internet and satellite connection to function properly otherwise system does not work.
- Sometime Ultrasonic sensor level indicator sowing high or low.
- GPS signals may fail or become inaccurate inside buildings and mountainous.
- High voltage battery use or sensor and devices used in the systemmay drain battery power quickly.

5.4 Application

- Motorbike Helmet.
- Bicycle Helmet.
- Car.
- Any vehicle.

5.5 Limitation

- Security Concerns: IoT devices often collect sensitive data, making them prime targets for cyberattacks.
- Interoperability Issues: With various devices using different protocols and standards, ensuring seamless communication and interoperability among them can be complex.
- Indoor Tracking Challenges: GPS signals have difficulty penetrating buildings, leading to inaccurate or unavailable location data indoors.
- Privacy Concerns: Location tracking collects sensitive data about an individual's movements and habits. This raises privacy concerns.
- Security Risks: Location tracking systems can be vulnerable to hacking, spoofing, or unauthorized access.

5.6 Future Scope

- We can implement the whole circuit into small module later.
- Less power consuming safety system.
- Camera can be added to capture image of the Vehicle.
- In future a penalty system can be added with this IoT device.

5.7 Conclusion

As the number of accidents on the road in Bangladesh increased day by day, so it is necessary to check speed of the Vehicles on highways. So as to reduce accident cases and to provide a safe journey by controlling high speed of the Vehicle. The proposed IoT device is very helpful for reducing the accident occurs on the road in Bangladesh.
The proposed system has developed for Vehicle speed control and monitor. The proposed IoT device can detect the Vehicle speed and tracking Vehicle current location. The IR sensor has been used for measuring the vehicle speed

REFERENCES

[1] Abdullah H. Alquhali, Mardeni Roslee, Mohamad Y. Alias, Khalid S. Mohamed (2020) IOT Based Real-Time Vehicle Tracking System. (Book)

[2] Ali Mustafa, Mohammed I. A al-Nouman, Osama A. Awad (2019) A Smart real-time tracking system using GSM/GPRS technologies.

[3] Abdullah H. Alquhali, Mardeni Roslee, Mohamad Y. Alias, Khalid S. Mohamed (2020) IOT Based Real-Time Vehicle Tracking System.

[4] Ali Mustafa, Mohammed I. A al-Nouman, Osama A. Awad (2019) A Smart real-time tracking system using GSM/GPRS technologies.

[5] J.Pelegri, J.Alberola, and V.Llario, ―Vehicle detection and car speed monitoring system using GMR magnetic sensors‖, IEEE conference on Industrial Electronics, no.2 (2002

[6] Rajesh Kannan Megalingam, Vineeth Mohan, Ajay Mohanan, Paul Leons, Rizwin Shooja, "Wireless sensor network for vehicle speed monitoring and traffic routing system "IEEE Int.con on Mechanical and Electrical Technology, 2010"

[7] Ravi Kishore Kodali and Sairam, ―Over speed monitoring system‖, IEEE Int.con on Contemporary Computing and Informatics, no.31(2016)

[8] G.Kirankumar, J.Samsuresh, and G.Balaji, ―Vehicle speed monitoring system[VSM] (using Rubee protocol)‖, IACSIT Int.Journal of Engineering and Technology, no.1 (2012

[9] Ahmed Al-Gindy, Ayman Tawfik, Laila Sakhi, Hibaq Mohamad & Manahil Mustafa, ―RFID speed monitoring system ‖, IEEE Int.con on Developments in eSystems Engineering, (2013).

[10] Farooq, M.U., Shakoor, A. and Siddique, A.B. (2017) GPS Based Public Transport Arrival Time Prediction. https://doi.org/10.1109/FIT.2017.00021

[11] Kiran, R., et al. (2017) Implementation of Smart Bus Tracking System Using Wi-Fi.
International Journal of Innovative Research in Science, Engineering and Technology, 6, 12940-12946

[12] Sridevi, K. (2017) Smart Bus Tracking and Management System Using IoT. International Journal for Research in Applied Science and Engineering Technology, 5, 372-374.

Appendix

Program for Accident Detection:

```
#define BLYNK_PRINT Serial

/* Fill-in your Template ID (only if using Blynk.Cloud) */
#define BLYNK_TEMPLATE_ID "TMPL65X-lnCmp"
#define BLYNK_TEMPLATE_NAME "Car Accident Monitoring"
// #define BLYNK_AUTH_TOKEN "X6HphzVPdprOjH9lGRx0jIUM7m6Xv-w1"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Wire.h>
#include <MPU6050.h>


//    for    jayro
MPU6050 mpu;
int16_t ax, ay, az;
int16_t gx, gy, gz;
int x, y;

//for alart
int relay = D6;
int power_relay = D7;

char auth[] = "X6HphzVPdprOjH9lGRx0jIUM7m6Xv-w1";
char ssid[] = "student";
char pass[] = "iotstudent";

#define echoPin D0  // attach pin D2 Arduino to pin Echo of HC-SR04
#define trigPin D5  //attach pin D3 Arduino to pin Trig of HC-SR04

// defines variables
long duration;  // variable for the duration of sound wave travel
int distance;  // variable for the distance measurement
```

```
int light;
int power_control;

void setup() {
  Serial.begin(9600);

  pinMode(relay, OUTPUT);        // Sets the trigPin as an OUTPUT
  pinMode(power_relay, OUTPUT);  // Sets the trigPin as an OUTPUT

  pinMode(trigPin, OUTPUT);  // Sets the trigPin as an OUTPUT
  pinMode(echoPin, INPUT);   // Sets the echoPin as an INPUT
  digitalWrite(relay, HIGH);
  digitalWrite(power_relay, HIGH);

  Wire.begin();

  Blynk.begin(auth, ssid, pass);
}

BLYNK_WRITE(V2)  // V0 is the number of Virtual Pin
{
  if (light == 0) {
    digitalWrite(relay, HIGH);
  }

  if (light == 1) {
    digitalWrite(relay, LOW);
  }
}

BLYNK_WRITE(V3)  // V0 is the number of Virtual Pin
{
  if (power_control == 0) {
    digitalWrite(power_relay, HIGH);
  }

  if (power_control == 1) {
    digitalWrite(power_relay, LOW);
  }
}

void loop() {
  Blynk.run();

  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;  // Speed of sound wave divided by 2 (go and back)
```

```
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  if (distance <= 1) {
    distance = 35;
  }

  if (distance >= 35) {
    distance = 35;
  }

  distance = map(distance, 35, 3, 0, 100);

  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
  x = map(ax, -17000, 17000, 300, 400);  //Send X axis data
  y = map(ay, -17000, 17000, 100, 200);  //Send Y axis data

  Serial.print(x);
  Serial.print("|");
  Serial.println(y);

  if (x > 360 || x < 330 || y > 160 || y < 130) {
    Serial.println("Accident Detected");
    Blynk.virtualWrite(V1, "Accident Detected");
  }

  if (x < 360 && x > 330 && y < 160 && y > 130 && distance > 20) {
    Serial.println("Everything Ok");
    Blynk.virtualWrite(V1, "Ok");
  }

  if (distance < 20) {
    Blynk.virtualWrite(V1, "Oil Level Low");
    Serial.println("Oil Level Low");
  }


  Blynk.virtualWrite(V0, distance);
  delay(1000);
}
```

Program for Car Control:

```
#define ENA 15  // Enable/speed motors Right        GPIO14(D7)
#define ENB 13  // Enable/speed motors Left         GPIO12(D8)
#define IN_1 14  // L298N in1 motors Right          GPIO15(D5)
```

```
#define IN_2 12  // L298N in2 motors Right          GPIO13(D6)
#define IN_3 2  // L298N in3 motors Left           GPIO2(D4)
#define IN_4 0  // L298N in4 motors Left           GPIO0(D3)

#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>

String command;      //String to store app command state.
int speedCar = 800;  // 400 - 1023.
int speed_Coeff = 3;


const char* ssid = "NodeMCU Car";
ESP8266WebServer server(80);

void setup() {
  pinMode(ENA, OUTPUT);
  pinMode(ENB, OUTPUT);
  pinMode(IN_1, OUTPUT);
  pinMode(IN_2, OUTPUT);
  pinMode(IN_3, OUTPUT);
  pinMode(IN_4, OUTPUT);

  Serial.begin(9600);

  // Connecting WiFi

  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid);

  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);

  // Starting WEB-server
  server.on("/", HTTP_handleRoot);
  server.onNotFound(HTTP_handleRoot);
  server.begin();
}

void goAhead() {

  digitalWrite(IN_1, LOW);
  digitalWrite(IN_2, HIGH);
  analogWrite(ENA, speedCar);

  digitalWrite(IN_3, LOW);
  digitalWrite(IN_4, HIGH);
  analogWrite(ENB, speedCar);
```

```
}

void goBack() {

  digitalWrite(IN_1, HIGH);
  digitalWrite(IN_2, LOW);
  analogWrite(ENA, speedCar);

  digitalWrite(IN_3, HIGH);
  digitalWrite(IN_4, LOW);
  analogWrite(ENB, speedCar);
}

void goRight() {

  digitalWrite(IN_1, LOW);
  digitalWrite(IN_2, HIGH);
  analogWrite(ENA, speedCar);

  digitalWrite(IN_3, HIGH);
  digitalWrite(IN_4, LOW);
  analogWrite(ENB, speedCar);
}

void goLeft() {

  digitalWrite(IN_1, HIGH);
  digitalWrite(IN_2, LOW);
  analogWrite(ENA, speedCar);

  digitalWrite(IN_3, LOW);
  digitalWrite(IN_4, HIGH);
  analogWrite(ENB, speedCar);
}


void stopRobot() {

  digitalWrite(IN_1, LOW);
  digitalWrite(IN_2, LOW);
  analogWrite(ENA, speedCar);

  digitalWrite(IN_3, LOW);
  digitalWrite(IN_4, LOW);
  analogWrite(ENB, speedCar);
}

void loop() {

  server.handleClient();
```

```
  command = server.arg();
  Serial.println(command);

  if (command == "F") goAhead();
  else if (command == "B") {
  Serial.println("back");
    goBack();
  } else if (command == "L") goLeft();
  else if (command == "R") goRight();
  else if (command == "I") goAheadRight();
  else if (command == "G") goAheadLeft();
  else if (command == "J") goBackRight();
  else if (command == "H") goBackLeft();
  else if (command == "S") stopRobot();
}
```

Program for Location Monitoring:

```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include "Ubidots.h"


const char* UBIDOTS_TOKEN = "BBUS-X4s4DblmiqIPozqjS9cSuU8CZGdwp6";  // Put here your Ubidots TOKEN
const char* WIFI_SSID = "student";                                  // Put here your Wi-Fi SSID
const char* WIFI_PASS = "iotstudent";

Ubidots ubidots(UBIDOTS_TOKEN, UBI_HTTP);

// Ubidots ubidots(UBIDOTS_TOKEN, UBI_TCP);


static const int RXPin = D5, TXPin = D6;
static const uint32_t GPSBaud = 9600;

float lat = 00.000000;
float lng = 00.000000;
char latar[20] = { ' ' };
char lngar[20] = { ' ' };
int lat_int;
int lng_int;
float lat_float;
float lng_float;
```

```
int lat_fra;
int lng_fra;

// The TinyGPS++ object
TinyGPSPlus gps;

// The serial connection to the GPS device
SoftwareSerial ss(RXPin, TXPin);

void setup() {
  Serial.begin(9600);
  ss.begin(GPSBaud);
  ubidots.wifiConnect(WIFI_SSID, WIFI_PASS);
}

void loop() {
  // This sketch displays information every time a new sentence is correctly encoded.
  if (ss.available() > 0) {
    gps.encode(ss.read());
    Serial.print(F("Location: "));

    if (gps.location.isValid()) {
      // Serial.print(gps.location.lat(), 6);
      lat = gps.location.lat();
      // Serial.print(F(","));
      // Serial.println(gps.location.lng(), 6);
      lng = gps.location.lng();
    }
    Serial.print(lat);
    Serial.print(",");
    Serial.println(lng);

    ubidots.addContext("lat", lat);
    ubidots.addContext("lng",  lng);

    char* context = (char*)malloc(sizeof(char) * 60);
    ubidots.getContext(context);
    int Value = 1;
    ubidots.add("position", Value, context);  // Change for your variable name
    // ubidots.send();
    bool bufferSent = false;
    bufferSent = ubidots.send("gps");  // Will send data to a device label that matches the
device Id

    if (bufferSent) {
      // Do something if values were sent properly
      Serial.println("Values sent by the device");
    }

    smartDelay(60000);
```

```
  }

  if (millis() > 5000 && gps.charsProcessed() < 10) {
    Serial.println(F("No GPS detected: check wiring."));
    while (true)
      ;
  }
}

static void smartDelay(unsigned long ms) {
  unsigned long start = millis();
  do {
    while (ss.available())
      gps.encode(ss.read());
  } while (millis() - start < ms);
}
```