A Mini Project Report On

SafeDrive: Preventing Drunk Driving with IoT Technology

Submitted in Partial fulfillment of the requirements for the award of the Degree of

Bachelor of Technology

In

Department of Computer Science and Engineering

By

Md Abdul Abed	22241A0547
Ambati Sai Teja	23245A0501
Gonewar Pavan Kumar	23245A0506
Cheepelly Siddhartha	22241A0518

Under the Esteemed guidance of

Dr K Anuradha

Professor

Department of Computer Science and Engineering

GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous)

Bachupally,Kukatpally,Hyderabad,Telangana,India,500090

2024-2025

GOKARAJU RANGARAJU

INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous)

CERTIFICATE

This is to certify that the Mini Project entitled "**SafeDrive: Preventing Drunk Drvining with IoT Technology**" is submitted by **Md Abdul Abed (22241A0547), Ambati Sai Teja (23245A0501), Gonewar Pavan Kumar (23245A0506), Cheepelly Siddhartha (22241A0518)** Partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in Computer Science and Engineering during the academic year **2024-2025.**

INTERNAL GUIDE

Dr. K Anuradha

Professor

HEAD OF THE DEPARTMENT

Dr. B. SANKARA BABU

Professor & HoD

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

Many people helped us directly and indirectly to complete our project successfully. We would like to take this opportunity to thank one and all. First, we wish to express our deep gratitude to our internal guide **Dr K. Anuradha**, **Professor,** Department of CSE for her support in the completion of our project report. We wish to express our honest and sincere thanks to **V. Jyoti and K. Adi Lakshmi** for coordinating in conducting the project reviews. We express our gratitude to **Dr. B. Sankara Babu, HOD**, department of CSE for providing resources, and to the principal **Dr. J. Praveen** for providing the facilities to complete our Mini Project**.** We would like to thank all our faculty and friends for their help and constructive criticism during the project completion phase. Finally, we are very much indebted to our parents for their moral support and encouragement to achieve goals.

Md Abdul Abed - 22241A0547
Ambati Sai Teja - 23245A0501
Gonewar Pavan Kumar - 23245A0506
Cheepelly Siddhartha - 22241A0518

DECLARATION

We hereby declare that the Mini Project entitled "**SafeDrive: Preventing Drunk Drvining with IoT Technology** " is the work done during the period from 16th Jan 2025 to 13th May 2025 and is submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering from **Gokaraju Rangaraju Institute of Engineering and Technology.** The results embodied in this project have not been submitted to any other university or Institution for the award of any degree or diploma.

Md Abdul Abed - 22241A0547
Ambati Sai Teja - 23245A0501
Gonewar Pavan Kumar - 23245A0506
Cheepelly Siddhartha - 22241A0518

	Table of Contents	
Chapter	**TITLE**	**Page No**

	LIST OF FIGURES	
Fig No	**Fig Title**	**Page No**
1.1	System Design	3
3.2.1	MQ3 Sensor	13
3.2.2	Neo 6M GPS Module	13
3.2.3	ESP32 Microcontroller	14
3.2.4	Jumper Wires	14
3.2.5	BreadBoard	15
3.2.6	Gear Motor	15
3.2.7	Battery	16
3.2.8	L298N 2A Based Motor Driver Module	16
4.1.2	System Architecture	19
4.3.1	Use Case Diagram	22
4.3.2	Class Diagram	23
4.3.3	Sequence Diagram	24
5.2.1	Hardware Setup	37
5.2.2	Dash Board	38

	LIST OF TABLES	
Table No	**Table Name**	**Page No**
5.3.1	Alcohol Detection and Vehicle Stop or Not	39
5.3.2	Remote Monitoring by Law Enforcement	39

ABSTRACT

Drunk driving is a serious issue, resulting in many accidents regardless of strict regulations and public education campaigns. Classical approaches, such as police roadside breathalyzer screening, identify only drunk drivers when they are on the road. To avoid accidents of this sort, our project presents SafeDrive, an intelligent alcohol detection system using IoT technology.

This device integrates the MQ-3 alcohol sensor and an ESP32 microcontroller to continuously monitor the driver's breath for alcohol level. The sensor can be mounted on the seatbelt or wherever else near the driver to efficiently detect. When the alcohol level is above the allowed level before the car is allowed to start, the system will not let the car start. But if the driver drinks and drives and crosses the limit, the system decelerates the vehicle slowly rather than abruptly stopping it, making it safe. Also, the ESP32 provides real-time data transmission, and with a special application, law enforcement officers can monitor the live location of the vehicle, alcohol content, and driver information. This allows the authorities to take action accordingly prior to the occurrence of the accident. Through the incorporation of IoT technology, real-time tracking, and controlled vehicle response, SafeDrive presents a productive and proactive means of curbing accidents caused by drunk driving, in turn encouraging road safety.

CHAPTER 1

INTRODUCTION

Drunk driving remains one of the largest problems in road safety and results in thousands of avoidable deaths and injuries each year. Despite the presence of stringent traffic laws and awareness campaigns, many individuals continue to drive after alcohol intoxication. Traditional methods of identifying drunk drivers using roadside breathalyzer tests are typically response-based and limited in scope. These conventional methods only detect drunk drivers once they are already in motion, which is a significant risk to the safety of society.

In an attempt to overcome the weaknesses of traditional systems, this project introduces SafeDrive, a smart alcohol detection and vehicle control system based on Internet of Things (IoT) technology. The major goal of this system is to avoid drunk driving accidents ahead of time by monitoring the driver in real-time and acting preventatively before an accident can be achieved. The sensor is built with the MQ3 alcohol sensor, which is able to detect alcohol in the breath of the driver. The sensor is mounted along with an ESP32 microcontroller, which reads the data and makes real-time decisions. The sensor can be installed near the driver, i.e., on the seatbelt, such that it can monitor breath all the time. If the measured alcohol level is beyond the allowed threshold, the ESP32 controller intervenes it either does not allow the engine to start or, if the car is already moving, smoothly brings it to a stop without sudden braking, thereby ensuring safety. The system also includes a GPS module for monitoring the location of the vehicle. This data, the alcohol level, and driver information are then sent to cloud storage using the ESP32. The information is then accessible to law enforcement officials through a remote monitoring portal. Police officers are able to track real-time updates and respond accordingly before an accident is initiated.

By integrating real-time alcohol detection, GPS tracking, and remote monitoring, SafeDrive offers a holistic and proactive solution to drunk driving. The integration of IoT offers real-time monitoring and smart response, thereby enhancing public safety and eradicating drunk driving hazards.

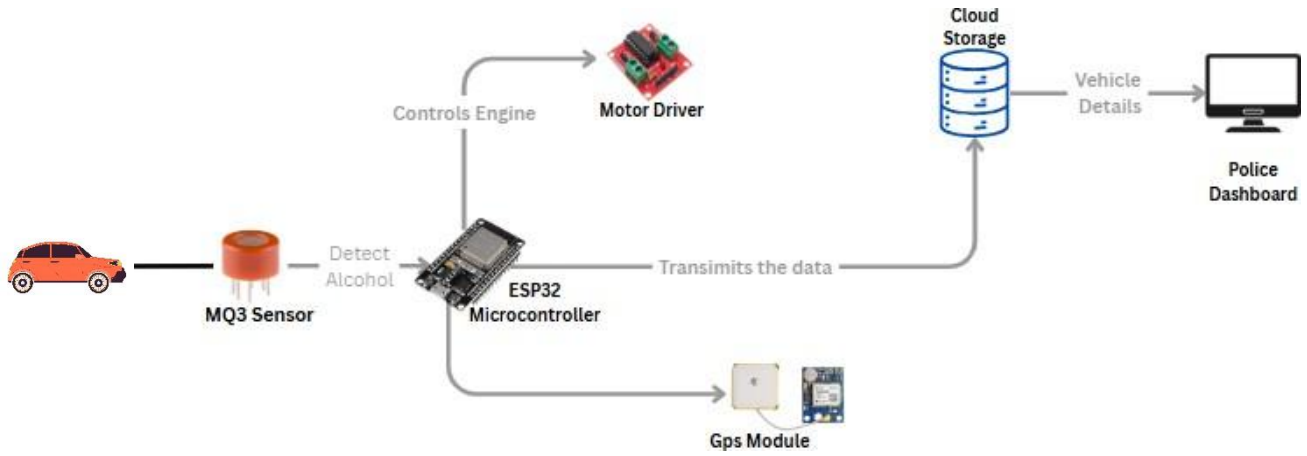


Fig 1.1: System Design

This diagram shows the simple and smooth flow of the SafeDrive system, which detects drunk driving. First, the MQ3 Sensor detects alcohol from the driver's breath. When alcohol is detected, the data is sent to the computer, which in this case is an ESP32 microcontroller. The ESP32 is very important! The ESP32 takes the data and analysis for the alcohol along with controlling the motor driver to control the engine. If the illegal level of alcohol in the blood is too unsafe, the motor driver prevents the engine from starting.

At the same time, the GPS module is connected to the ESP32, which relays to the ESP32 the current location of the vehicle. All condensing the rest of the information, which includes both location and alcohol level, will all get sent to cloud storage for security. At that point, the data from cloud storage is displayed on a police dashboard for every officer to see. Officers can monitor the situation and intervene in real-time if needed. Overall, the system provides the user an intelligent design that enhances road safety by detecting drunk driving defects and can respond to it live.

CHAPTER 2

LITERATURE SURVEY

[1]"IoT Based Alcohol Detection and Vehicle Control System" (2024) The study presented in this paper examines the implementation of IoT-based human biometric systems for the detection of alcohol to enhance the security of the driver. The system aims at detecting the alcohol content in the system of a driver via IoT technologies by taking parameters such as accuracy, response time, transmission efficiency of data, and the overall effectiveness toward preventing accidents. By constant observation of such variables, the system works to achieve an active safety system by identifying drunken drivers in real time and issuing a warning either to the driver or to the concerned authorities when needed. Besides, it has the ability to execute automatic safety measures like the inability of the engine to be started or braking of a vehicle if alcohol concentrations are recorded over a permissible level. The data gathered can also be uploaded to a central cloud platform to assist in enforcing traffic regulations and tracking repeat offenders. Nevertheless, the operation of the system may be hindered under certain circumstances. In particular, the system will not be able to determine the alcoholic content precisely when the driver's mouth is masked by a mask, scarf, or helmet, as the system needs a direct breath sample for analysis. Also, if the sensor duct is clogged with dust, moisture, or other impurities, then the accuracy of the reading could be spoiled, leading to false readings, lag in response, or total sensor malfunction. Environmental conditions like humidity, ambient temperature, or distracting odors like smoke or perfume could also affect sensor output. In addition, latency in the system due to oscillating network linkages, component failures, or power outages can slow the processing of data in real time and the precision of responses. Long-term exposure also presents long-term durability issues and periodic re-calibration problems for sensors like the MQ-3 to prevent long-term variance and unreliability. Although there is much potential for this system to reduce accidents and make driving safer, these potential flaws must be perfected so that it can function at its best under any driving conditions. Enhancements in the future can involve incorporating machine learning models for adaptive analysis, biometric authentication for driver identification, and adding redundant sensing or self-cleaning technologies to maintain consistent performance across various environments.

[2] "Realtime Alcohol Detection and Engine Locking System with ESP8266" (2024) This article presents the design of an engine locking and real-time alcohol sensor based on an ESP8266 microcontroller together with an MQ-3 alcohol sensor. It is a system designed to enhance road safety through continuous monitoring of the breath of the driver for alcohol content and automatic response if drunkenness is identified. Upon the detection of alcohol levels over a predefined set limit, the

system can remotely shut down the ignition system of the vehicle, hence stopping the driver from driving under the influence. The ESP8266 module also allows wireless communication, whereby the system can send alerts or warnings to authorities or concerned relatives in real time. This low-cost and small-scale system renders it a good contender for incorporation in commercial as well as individual cars. However, the system does possess some drawbacks that can affect its performance and reliability in real-world application. One of these drawbacks is the lag in sending notifications, which can decrease the effectiveness of prompt intervention. This delay can be attributed to network instability, processing delay, or power fluctuations within the ESP8266 module. Furthermore, there is also a possibility of user bypassing—where the driver can try to deceive the system by taking external airflow, breath of another person, or physically manipulating the sensor or connections in order not to get detected. These vulnerabilities indicate the necessity for increased security and stability in the design of the system. Future enhancements could include secure biometric authentication that the driver being tracked is indeed the actual driver, and the development of tamper-proof hardware enclosures. Real-time cloud logging, GPS-monitored vehicle tracking, and AI-driven monitoring of driving behavior could also add more value to the system as a preventive measure against driving while intoxicated.

[3]“Alcohol Detection and Theft Alert System for Vehicles” (2023) In this paper, an alcohol detection and vehicle theft alert system for cars is proposed based on a mixture of alcohol sensors, GPS, and GSM modules to improve both driver safety and vehicle security. The system is aimed at detecting alcohol in the breath of the driver and, at the same time, monitoring the position of the vehicle with GPS technology. If alcohol detection is detected beyond a set level, the system can disable car ignition and issue alerts via GSM to pre-registered contacts. The system also has a theft alert system that tracks unauthorized access or movement of the vehicle. Upon activation, it is meant to alert the owner of the vehicle and authorities via SMS with the actual location of the vehicle. This dual-use configuration renders it effective both as a drunk-driving prevention measure and as a method of reducing vehicle theft risk. However, several operational deficits impede its performance in real-world circumstances. The lack of confidence in the system is primarily contributed by the occurrence of false positives in detecting. theft—legal. or. normal. use. being. reported. as. suspicious. activity. This may lead to unjustified warnings and. confusion. with. diminishing. confidence. in. the. system. Yet another major flaw is the system's inability to continuously notify the police or respective authorities, particularly where there are network outages in an area or because of malfunctioning GSM modules. Gaps in communications like these make the theft guard mechanism weak and diminish the value of the overall system. To address these challenges, upcoming improvements could include simplification of the theft detection algorithm based on context-aware input, such as multi-factor driver authentication

and redundant communications channels (e.g., mobile application or internet warning) in addition to GSM. Incorporating a smarter decision layer that incorporates machine learning would also help eliminate false alarms while filtering and sending critical warnings reliably.

[4]“Safe Transportation System using IoT based Alcohol Detection” (2023) The current research proposes an IoT-based system with the aim of ensuring transport becomes safer through detecting alcohol use among drivers. The system includes a BAC sensor, microprocessor, and communication module, which interact to determine if the driver has used alcohol. When the measured level of alcohol exceeds the legal limit, the system is capable of preventing the car from being started and sending notifications to the authorized persons or authorities. It's a good measure for eliminating drunk driving by monitoring the driver's state in real time continuously and acting immediately.But one of its biggest handicaps is that it only speaks to finding alcohol—it doesn't have an interface to other critical vehicle safety systems such as lane watch, automatic braking, or driver fatigue detection. With no interfaces, the system can't provide every-angle protection or react to sophisticated driving conditions. It does not sends alerts to authorized individuals or authorities. It's a useful solution intended to stop drunk driving by tracking the condition of the driver in real-time and taking action immediately. But one of the significant drawbacks is that this system is alcohol-only—it does not integrate with other key in-vehicle safety systems such as lane detection, automatic braking, or driver fatigue monitoring. Without integration, the system cannot provide overall protection or react to more advanced driving scenarios. Also, the paper does not mention any contingency plan if the sensor malfunctions or the network connection is lost, which can be a problem. To enhance this system, future models should integrate alcohol detection with other intelligent features present in contemporary cars and potentially utilize AI to make wiser decisions based on the driver's overall behavior and the car's environment. Moreover, the paper is not mentioning anything as a back plan if there's a sensor failure or an end of connection within the network, which will be an area of concern. Improving upon this system in subsequent versions needs to integrate the alcohol detection mechanism with other features smart enough seen on contemporary automobiles and potentially include AI to better inform decisions depending on the
general behavior and condition of the car.

[5]“Drunk Driving Prevention and Automatic Bus Pass Tracking by using Image Processing” (2023). This article discusses a system that prevents drunk driving and automates bus pass tracking using a combination of alcohol detection and image processing. The main emphasis is on preventing the vehicle—especially public transport such as buses—from starting if the driver is under the influence of alcohol. The device tracks the BAC of the driver and, when it exceeds a predetermined legal limit,

stops the ignition in Deadman fashion so that the vehicle cannot move. Besides this, passenger tracking by image processing for automatic bus pass verification is also mentioned briefly in the paper, providing an extra level of functionality to public transport systems. This two-in-one strategy increases safety in addition to efficiency. One of the glaring omissions of the paper is the lack of attention to the reliability and precision of the alcohol sensor utilized by the system. Some of the items not addressed in so many words are sensor aging and calibration, environmental interference, and maintenance requirements. These have a tremendous influence on the quality of the performance of the sensor and can generate faulty readings if not addressed correctly. To make the system more robust, additional research can investigate how the accuracy of sensors can be sustained in the long term, maybe through periodic calibration methods or the addition of self-testing capabilities. It would also be beneficial to subject the system to various environmental conditions and have robust validation to produce consistent results in the actual application environments.

[6] AI-Powered Driver Behavior Prediction, Drunk Driving Prevention, Accident Detection, and Insurance Integration" (2023) This study introduces an AI-driven system with all-round road safety improvements in mind, predicting the behavior of the driver, avoiding drunk driving, sensing accidents, and collaborating with motor insurance systems. The system relies on IoT technology for remotely monitoring the driver's Blood Alcohol Concentration (BAC) in real-time. Besides detecting alcohol, it uses AI to monitor patterns in driver behavior via a synthetic dataset of driver behavior, which identifies signs of drowsiness, aggression, or distraction. Where abnormal behavior or elevated BAC levels are noticed, the system can intervene proactively by notifying the driver, alerting emergency contacts, or even initiating insurance-related workflows during an accident. This multi-purpose strategy seeks not only to lower road crashes but also to simplify the post-crash procedure through automated insurance notification. Yet, although the system technical functionality is well explained, one of the concerns is that there is little discussion of data security. Since the system involves real-time data exchange between the vehicle and remote monitoring systems, any communication disruption may be at the expense of driver privacy or system integrity .Issues like encryption, secure cloud storage, and protection against cyber-attacks are not addressed in the paper, and it has the potential risk for both drivers and service providers. To facilitate effective deployment, it is essential that subsequent innovations target fortifying the security platform—ensuring all data transfer is encrypted and secure from unauthorized access. Adding cybersecurity features like secure authentication mechanisms, routine audits, and end-to-end encryption can do much to enhance user trust and system resilience.

[7] "In-Car Breathalyzer Systems for Enhanced Road Safety through SVM Classification and IoT Connectivity" (2024) This paper presents an in-car breathalyzer system aimed at improving road

safety through the utilization of IoT connectivity and SVM (Support Vector Machine) classification to identify alcohol concentration in drivers. The system is constantly monitoring the driver's BAC (Blood Alcohol Concentration) and, upon detection of a high level, provides alerts to the driver as a warning. With the inclusion of SVM classification, the system is able to make more informed decisions from data collected, which makes it more reliable in determining safe versus unsafe BAC levels. The IoT component facilitates real-time data transmission and remote monitoring, hence being suitable for mounting in private and commercial vehicles as a preventive safety feature. The report is short of detail on user interface and is not explicit as to how the driver is receiving these alerts. Whether the response is provided visually, audially, or as haptic warnings (such as seat vibration or steering wheel alarms) is unknown. This omitted information leaves open questions regarding system usability and its ability to command the driver's attention in an emergency. It is important that an effectively designed user interface guarantee a prompt and adequate response by drivers to warnings. Future developments must aim at creating an intuitive and multi-modal notification system in which visual, auditory, and potentially tactile warnings work together for increased driver responsiveness. The user interface must also be tested against clarity, accessibility, and minimal distraction, justifying compliance with safety requirements and drivers' expectations.

[8]"IoT Based Alcohol Detection and Vehicle Control System" (2024) In this article, an IoT-based alcohol detection and vehicle control system is proposed for improving road safety by avoiding drunk driving. The system operates by checking the alcohol level in the driver's body and, if it is over a given limit, can perform actions like denying the ignition of the car or refusing to start the vehicle. The system lays emphasis on areas such as response time of the sensors, blood alcohol concentration limits, and detection precision to ensure smooth functioning. The system has the ability to gather information in real-time with the help of IoT technology, which would prove useful in real-time monitoring and timely intervention. One of the major drawbacks of the system is that it lacks real-time alert. While the alcohol detection functionality is functional, it fails to provide instant alerts to the driver or external authorities in case alcohol is found. Immediate communication would be essential to act instantly, e.g., alerting the driver to act at the moment or alerting emergency contacts or the authorities. Besides, the system is limited to alcohol detection and does not feature other required vehicle safety features such as monitoring driver behavior or compatibility with existing vehicle safety systems like automatic braking or lane departure warning. To increase the comprehensiveness of the system, future versions can introduce a real-time alert system and expand its capability to detect other driving hazards apart from alcohol consumption.

[9] "Design and Implementation of In-Vehicle Alcohol Detection and Speed Control System" (2022) This paper presents the design and development of a vehicle alcohol detection and speed control

system using IoT technology to increase road safety. The system utilizes an Arduino Nano, an MQ-3 sensor for alcohol, and Blynk Cloud for real-time alcohol detection, speed control, and SMS alerting. When the alcohol level exceeds a predetermined limit, the system triggers actions such as reducing the speed of the vehicle or providing SMS alerts to enrolled contacts. Cloud-based connectivity functionality offers the mechanism for real-time monitoring and instant communication to enhance safety features further. Nevertheless, the system does have many flaws that could compromise its usability in real life. One of the system's main flaws is a lack of integration of GPRS (General Packet Radio Service) into the system, which would limit the usability and functionality of the system's communicative tasks, especially where there is no network coverage. Additionally, the system can suffer from false alerts, whereby the alcohol would be detected by the sensor where alcohol was not present, issuing useless warnings. Additionally, the system lacks environment sensitivity to such variables as air quality, humidity, or temperature that may impair readings by the sensors and hence interfere with precision. Another important drawback is that Google Geolocation API support is not included in the system, which may have been used to support even more advanced features like location-based alerts or geofencing to secure driving zones. To improve the system, future upgrading can involve integrating GPRS functionalities, enhancing sensor calibration, and adding location-based services to augment real-time response and monitoring.

[10]"A Lightweight In-Vehicle Alcohol Detection Using Smart Sensing and Supervised Learning" (2022) The study suggests a low-cost in-vehicle alcohol detection system using six MQ-3 alcohol sensors with an optimizable shallow neural network (O-SNN) to dentify the alcohol level of drivers. The system is optimized to provide efficient and accurate detection while keeping hardware requirements low. Use of more than a single sensor enhances the reliability of the system in determining the level of alcohol concentration of the driver and using machine learning to validate the result. Insertion of a shallow neural network enhances the system's ability to separate correct from incorrect outcomes, further enhancing accuracy. Nevertheless, one of the most profound lacks this system is that it is not inclusive of the sensor being impacted by surrounding materials. Environmental factors like temperature, humidity, or the presence of other chemicals (e.g., air fresheners or smoke) can affect the performance of the MQ-3 sensors to provide false or inconsistent readings. The performance of the system in actual use may be impaired not considering these factors. Subsequent studies can also aim to make the sensors more robust through calibration methods that account for environmental factors and adding other sensors or algorithms to filter out noise from non-alcoholic items in the atmosphere.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 Software Requirements

- Operating System: Windows

  Microsoft's Windows operating system consists of a family of popular operating systems that are known for their Graphical User Interface (GUI), multitasking capabilities, and software support. The Windows OS started with Windows 1.0 in 1985 (with a very rudimentary GUI) and has really developed since then. Some of the rich features of Windows include a very user-friendly GUI (with the Windows Start menu and taskbar); robust networking features; multitasking abilities through the Task Manager and multi-desktops (as seen in Windows 10); gaming capabilities through DirectX and Xbox; and finally, very versatile editions of Windows that are tailored for specific needs. The Windows Home editions are geared to the general consumer while the Professional editions are aimed at a business consumer and have many more features available to users such as domain join, and group policy management.

- Programming Languages - Arduino C++:

  Programming languages are used to teach a computer to do some things. Each programming language has rules to abide by and specifies a vocabulary, similar to a natural language. Yet, processors cannot understand them directly, and a program that can be read by a human must be translated into binary values that can be understood by a computer. C++ is a very widely used and versatile language that typically needs a compiler which does the translation. Arduino C++ interacts so intimately with the hardware, there are quick response times and little delay—critical for safety-oriented tasks such as detecting alcohol concentration and responding instantly. And in addition, there's a gigantic Arduino community out there, complete with documentation, pre-written libraries, and plentiful advice, which simplifies troubleshooting and accelerates development.

- Frontend Development for Web Application:
  HTML, CSS, and **JavaScript** constitute the core of web development and are a prerequisite for developing the user interface of any web application.

HTML (HyperText Markup Language) organizes the content of web pages.

CSS (Cascading Style Sheets) styles that content—specifying layout, colors, fonts, and overall look.

JavaScript adds interactivity and dynamic behavior, allowing the application to respond to user actions in real time.

- IDE: Visual Studio Code
  Visual Studio Code (VS Code) is a free, open-source code editor from Microsoft. Key features include debugging, syntax highlighting, and version control support. While it does have a level of customization available through extensions, VS Code has a wealth of programming languages and remains popular because of speed and performance.

- Arduino IDE
  The IDE is IDE-friendly for Arduino C/C++ programming, and it features an extensive set of inbuilt libraries that ensure easy access to sensors, modules (such as GPS or Wi-Fi), and other hardware pieces. It also features: Serial monitor for debugging and real-time visual representation of data Board and port selection for selecting and connecting hardware easily Sketch management for project organization and execution. Its uncluttered interface, open-source programming, and robust community backing make it a favorite among the developers of hardware systems such as SafeDrive, where real-time sensor monitoring and control are integral.

- MQTT Explorer
  MQTT (Message Queuing Telemetry Transport) Explorer Is an application used to interact with MQTT Brokers. MQTT is a protocol used to get the Realtime data of small sensors.
  Key Features are: Real-Time Monitoring of Data: Real-time monitoring of incoming and outgoing messages is supported for instant observation of data that is being transmitted. Message Publishing and Subscribing: MQTT Explorer makes it possible to publish messages onto desired topics and subscribe to topics so that the users can be notified, providing seamless interaction with the MQTT broker. MQTT Broker Connectivity: The software allows connection to different MQTT brokers so that users can work on multiple brokers using a single interface.

3.2 Hardware Requirements

- **MQ3 Sensor:**

Fig 3.2.1: MQ3 Sensor

MQ3 Alcohol Sensor is a highly affordable and sought-after sensor used for the detection of alcohol vapors in the atmosphere. It is widely utilized in breathalyzer instruments, alcohol detecting systems, and other applications involving the measurement of alcohol concentration.
The MQ-3 sensor has a tin dioxide (SnO_2) sensitive layer that responds to alcohol molecules. With the exposure to the sensor surface by vapors of alcohol, the sensor's electrical resistance is modified. The modification of resistance is transformed into an analog signal, which can be detected and handled with a microcontroller (like the ESP32).

- **Neo 6M GPS Module:**

Fig 3.2.2: Neo 6M GPS Module

The NEO-6M GPS Module is a small and stable GPS (Global Positioning System) receiver employed to identify the actual geographical location of an object, vehicle, or individual in real-time. It is extensively applied in navigation systems, tracking, and IoT-based location-aware systems such as SafeDrive. NEO-6M uses signals from several GPS satellites revolving around the Earth to determine its latitude, longitude, altitude, and velocity by using triangulation. All this information is transmitted to the

microcontroller in the form of NMEA sentences (text-based GPS data).

- **ESP32 Microcontroller:**

Fig 3.2.3: ESP32 Microcontroller

Esp32 is a low cost and high-performance microcontroller, consists of on board wi-fi and bluetooth capabilities. Mostly used in IoT projects. It is a dual core processor with many input and output pins. It reads the data from various sensors and uploads the data into cloud. Due to its flexibility and speed, it is able to execute several tasks with ease, such as gathering sensor data, processing it, and transmitting it to other devices or systems in real time. The ESP32 interprets this information in real time and can be used to control vehicle systems, for example, preventing the ignition or slowing down the vehicle when alcohol is present. Its wireless feature allows live data transmission to a web application, which makes it an ideal candidate for smart, connected safety systems.

- **Jumper Wires:**

Fig 3.2.4: Jumper Wires

Jumper wires are essential components in electronics and prototyping, commonly used to establish connections between various points on a breadboard, between components on a circuit board, or between a circuit board and external components.

- **Breadboard:**

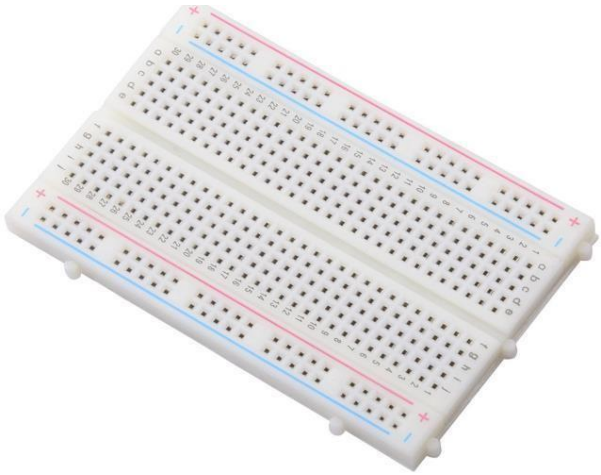


Fig 3.2.5: BreadBoard

A breadboard is a fundamental tool used in electronics prototyping and circuit design. It allows electronic components to be easily connected together without soldering, making it ideal for creating and testing circuits quickly.

- **Gear Motor:**

Fig 3.2.6: Gear Motor

A gear motor combines an electric motor and a gearbox to increase its functionality, especially in those applications that need high torque and regulated speed. The motor itself can be a DC motor or a stepper motor, depending on the torque and speed demands of the application. DC gear motors are widely used because they are reliable, easy to control, and appropriate for multiple industrial and consumer uses.

- **Battery:**

Fig 3.2.7: Battery

A battery is an electrochemical device that stores energy in chemical form and converts it into electrical energy when needed. It consists of one or more electrochemical cells, each containing positive and negative electrodes immersed in an electrolyte solution. During discharge, chemical reactions occur between the electrodes and electrolyte, releasing electrons that flow through an external circuit, generating electrical current.

- **L298N 2A Based Motor Driver Module:**

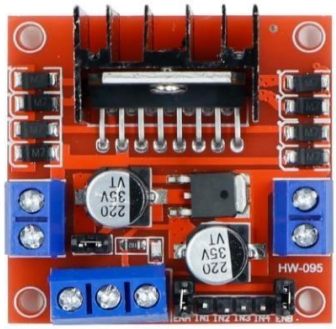


Fig 3.2.8: L298N 2A Based Motor Driver Module

The L298N 2A Motor Driver Module is a simple electrical component that can be used to run DC motors, stepper motors, and motors of other types. Where this gets used the most, in my experience, is in robot or automation projects that require speed and direction control. It has an H-Bridge circuit that can allow the motor to spin positively (clockwise) or negatively (counterclockwise). The L298N also allows for speed adjustments using Pulse Width Modulation (PWM). The L298N can provide up to 2A of current to the motors, which means it could be useful for moderate power applications. It's usually found with microcontrollers like Arduino or ESP32. The L298N is also protected against overcurrent and thermal shutdown; therefore, the module and motors should be safe to use.

3.3 Methodology

The SafeDrive system is created to harmlessly stop drunk drinking and driving by detecting alcohol in the driver’s breath, controlling whether the vehicle ignition or vehicle can move, notifying the police, and retaining information on a cloud server. The process of developing this system involves the following processes:

1. Sensor constructor and alcohol detection process

The first step for this project is to constructor a MQ-3 alcohol sensor that will go inside the seatbelt assembly or the dashboard area. The place does need to be close enough in order to detect the driver breath. The sensor will provide an analogue voltage that is proportional to alcohol present in the driver's breath. This analogue voltage can be read by the ESP32 microcontroller via its analogue input pin. The threshold value is programmed into the ESP32 code so it can determine if the alcohol concentration level is safe or unsafe.

2. Real-time decision-making process

The ESP32 constantly checks the current level of alcohol concentration. If the alcohol concentration is below the threshold, the vehicle can begin to start and operate normally. If the alcohol concentration exceeds the threshold level: The motor driver module (L298N) can be used to inhibit the engine from starting or safely bring a moving vehicle to stop or slows down. The buzzer which is connected to the ESP32 will activate and inform the driver the activity constitutes a violation.

3. GPS Tracking and Data Storage

At present, there is a GPS module connected with the ESP32 for tracking real-time location of the vehicle. The ESP32 reads the GPS coordinates and saves them either at fixed intervals or when it detects an alcohol thresholdviolation.

4. Cloud Connectivity, Remote Monitoring

The ESP32 has been programmed to communicate data (alcohol level, gps location, alert status) to a cloud database (e.g. firebase). The data is then visualized on a web dashboard that is available to an authority and/or monitoring staff. Law enforcement can access the dashboard to view violations and vehicle location and notifications, and all the data can be viewed in real-time.

5. Testing and Validation

Objects were tested in multiple test cases with different levels of alcohol impairment, to assess the accuracy and reliability of the system. Tests were also done to ensure vehicle control was stable, and that alerts, GPS updating and cloud data were all working as anticipated.

CHAPTER 4

PROPOSED APPROACH, MODULE DESCRIPTION AND UML DIAGRAMS

4.1 Proposed Approach

The SafeDrive project provides a smart car control system, preventing drunk driving with cloud-based monitoring and live alcohol detection. It employs an MQ3 alcohol sensor to not let the engine start if the driver is drunk. The vehicle is not unlocked and the system sends a warning along with the vehicle's GPS location to remote police stations via a dashboard if alcohol is found. The objective is to provide road safety by means of an instant, automatic, and real-time response system.

In essence, the system architecture consists of hardware blocks such as MQ3 sensor, ESP32 microcontroller, GPS module, and motor driver augmented by cloud integration through Firebase and web-monitoring interface developed through ReactJS. Such modularity enables real-time decision-making as well as open observation of the vehicle's key data.

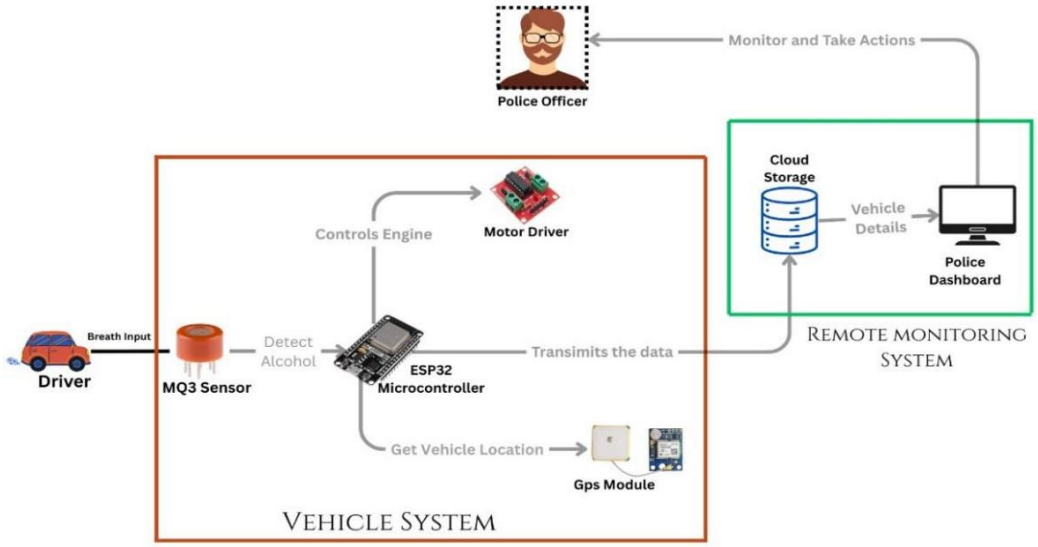


Fig 4.1.2: System Architecture

This chart clearly shows how the **SafeDrive Vehicle Monitoring System** works to stop drunk driving through the use of smart technology and continuous real-time updates. The process begins when the driver exhales into the system. The system knows to begin night only when it can accurately check for alcohol use with the use of the **MQ3 alcohol sensor**. The MQ3 sensor checks the breath to see if there is alcohol in

the breath. If it detects alcohol, it immediately sends the alcohol level data to the **ESP32 microcontroller**. The ESP32 is the main microcontroller to the system, and if the alcohol level is too high, the ESP32 will take action and stop the engine from running anymore from the **motor driver** keeping the vehicle still and stop the vehicle from moving under unsafe conditions.

At the same time, when the vehicle is still and the driver alcohol is detected, the ESP32 is obtaining the car location from the **GPS module**. The ESP32 would send the latest data point - the alcohol level, car location data and important car information to **cloud storage**. The officer can see this data from the newly created **police dashboard** to monitor in a real-time dashboard. The officer or police department could initiate action based on real-time data points and proceed with measures to mitigate the situation. This is a new type of **remote monitoring system** and together provides law enforcement with a way to catch drunk driving early and take action with better information, helping to improve safety and the roadways for all.

For the purpose of modularity and convenience, the solution has been divided into four typical layers:

1. **Sensing Layer:**

Saves driver's breath using MQ3 sensor.

Alcohol senses and returns to microcontroller.

2. **Control Layer:**

ESP32 executes the sensor and determines whether to turn on or off the motor.

Compares present location using GPS module while sensing alcohol.

3. **Communication Layer:**

Sends alcohol and location information to cloud using Wi-Fi and Firebase services.

4. **Monitoring Layer:**

Real-time status of cars and alarm screen on police dashboard built using ReactJS.

Enables monitoring, tracking, and corresponding police reaction.

SafeDrive is an alcohol sensing anticipation system that will not allow drunk drivers to start their cars, minimizing the chances of accidents. It can detect alcohol levels and send real-time information such as vehicle status and GPS coordinates using MQTT for instant, secure communication even on low-bandwidth networks. The real-time feedback avoids the issue at the very source to provide a safer drive.

The system is modular and scalable as well, thus can be implemented in single vehicles and large fleets such as public or delivery transportation. SafeDrive remains inexpensive and affordable even when built with low-cost components such as ESP32 microcontroller and MQ3 alcohol sensor, hence becoming even more feasible in resource-constrained environments. Low power consumption and wireless technology also

enhance long-term utilization, bridging the gap between human security and smart automaton to enhance road safety.

4.2 Modules Description:

Module 1: Detection and Monitoring of Alcohol

This is the philosophy of the SafeDrive system. With an MQ-3 alcohol sensor placed near the driver—a position such as on the seatbelt—it tests for alcohol from the driver's breath at all times. In real-time, this data is read by the ESP32 microcontroller. At a level that exceeds the legally allowed, the system acts to ensure safety.

Module 2: Vehicle Lock System

If alcohol is present before the engine is started, this module comes into action in an attempt to prevent the car from starting. It activates a relay connected to the ignition of the car, which secures the engine until it is safe to drive. This will prevent the car from being driven until it is safe to do so.

Module 3: Safe Engine Control During Driving

At other times, the driver might drink after the vehicle has been started. In such a situation, rather than suddenly stopping the vehicle, which is risky, this module slows down the vehicle gradually. It's a clever method of slowing down the car to a safe stop without endangering anyone.

Module 4: Real-Time GPS Tracking

To track where the car is, this module uses a GPS module that is interfaced with the ESP32. It gets the current location (latitude and longitude) continuously, so the system knows where the car is at any time. This proves to be useful especially when there is a requirement to send notifications.

Module 5: Alert and Communication

When the system picks up signs of trouble like the driver having too much to drink—it immediately shares important details, such as how high the alcohol level is, where the vehicle is at that moment, and who the owner is. Instead of sending a regular text message, it uses MQTT, a smart and speedy way to send data online. On the other end, the police dashboard is always tuned in and ready, so they get the alert instantly. This way, help can reach faster, and the whole process is smoother and more dependable—without relying on outdated SMS technology.

Module 6: Cloud Updates to Police Monitoring

With MQTT (a low-latency messaging protocol), all the data is sent to a centralized system in real time. The police or safety authorities can subscribe to the data stream and view each vehicle's status in real time, enabling them to respond promptly before anything goes wrong.

Module 7: Web Dashboard for Remote Monitoring

All the data collected from the vehicle—like alcohol readings, GPS location, and system status—is shown on a ReactJS-based web dashboard. This user-friendly interface makes it easy for administrators or law enforcement to track everything from one place, in real-time.

4.3 UML DIAGRAMS

Use case Diagram:

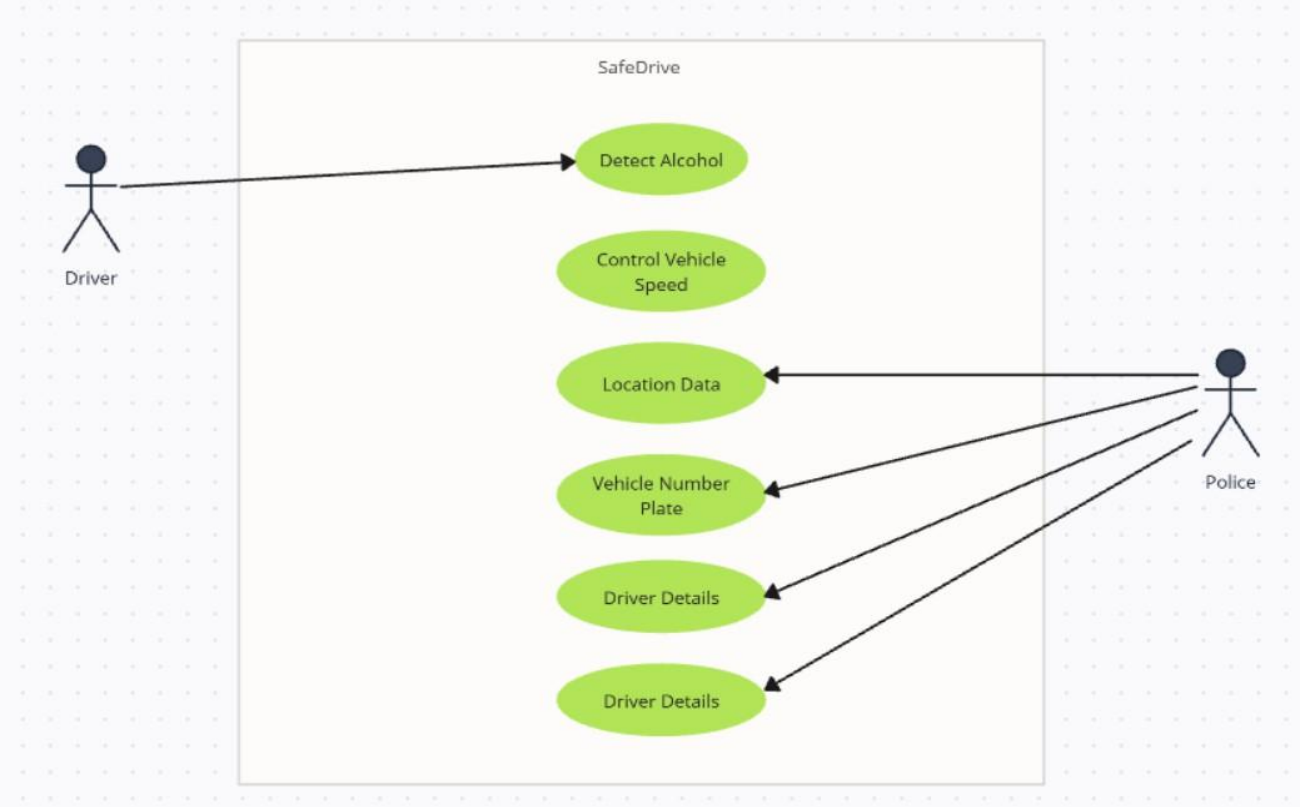


Fig 4.3.1: Use CaseDiagram

This use case diagram illustrates the most important interactions within the SafeDrive system between two primary actors: the Driver and the Police. The Driver is interacting with the system mostly in order to trigger the alcohol detection process. If alcohol is present, the system will take automatic control actions such as adjusting the speed of the vehicle for safety purposes. Parallel to this, the Police are also provided with access to key data points by the system. These are the location of the vehicle, number plate, and driver details, which

are automatically dispatched to them if there is any unusual alcohol level or dangerous driving behavior. These provisions assist authorities in tracking and taking action on suspected cases of drunken driving. In general, this diagram illustrates how SafeDrive promotes road safety by linking the status of the driver to the awareness of the police and control of the car.

Class diagram:

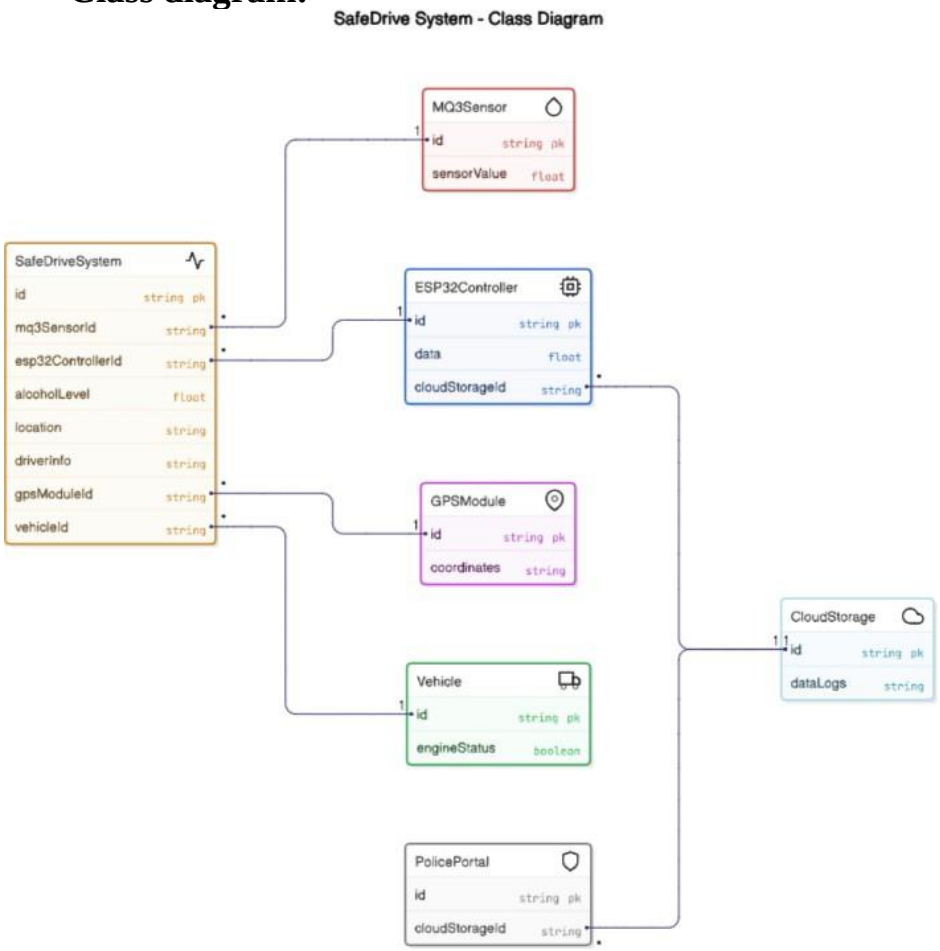


Fig 4.3.2: Class Diagram

The SafeDrive system class diagram illustrates how each of the main elements in the project interacts. At the core of it is the SafeDrive System, which interfaces with various components such as the MQ3 alcohol sensor, the ESP32 controller, GPS module, and the car itself. The MQ3 sensor checks if the driver has consumed alcohol, and the ESP32 reads the data and processes it and then proceeds accordingly—like cutting off the engine when required. It also transfers the data to a cloud storage system. The GPS module assists in vehicle location tracking, and the vehicle class operates the engine based on the ESP32 decision process. There is also a cloud storage system in which all the data is stored, and a police portal via which authorities can retrieve this data and take action if required. In general, the diagram clearly shows how each component is connected and works together to prevent drunk driving.

Sequence diagram:

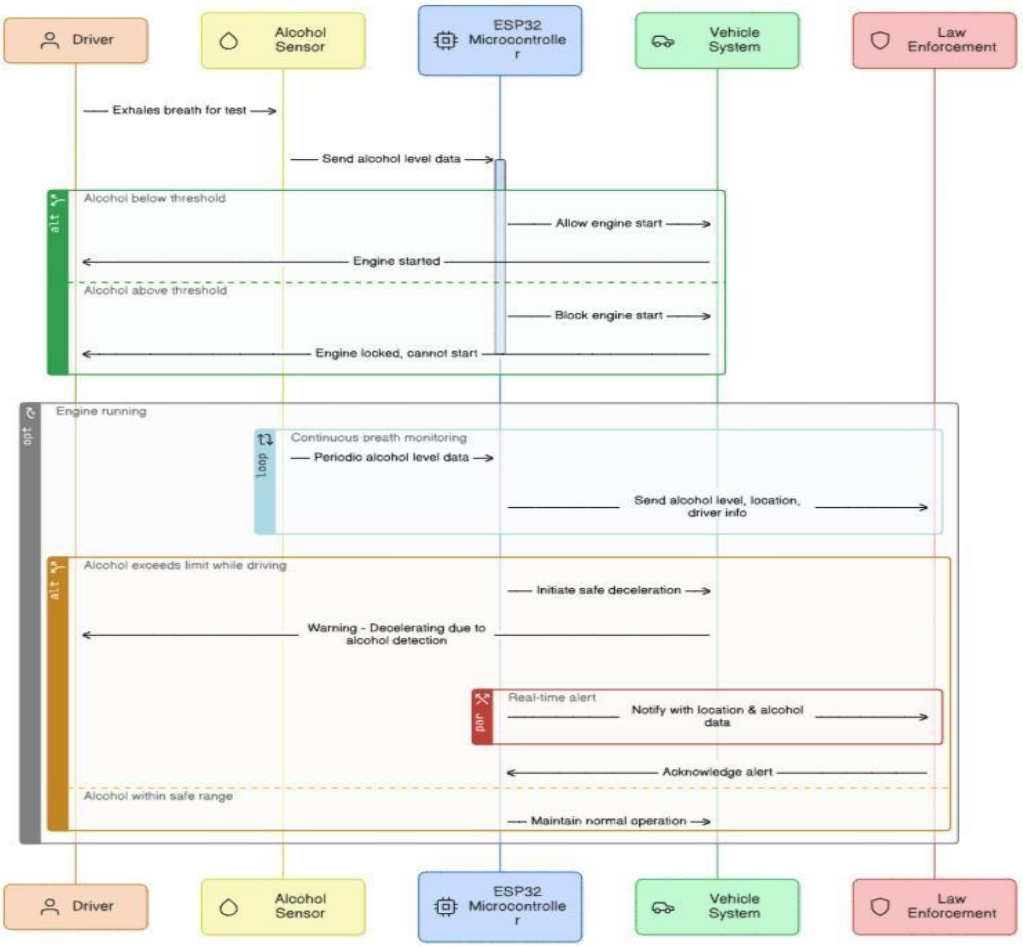


Fig 4.3.3: Sequence Diagram

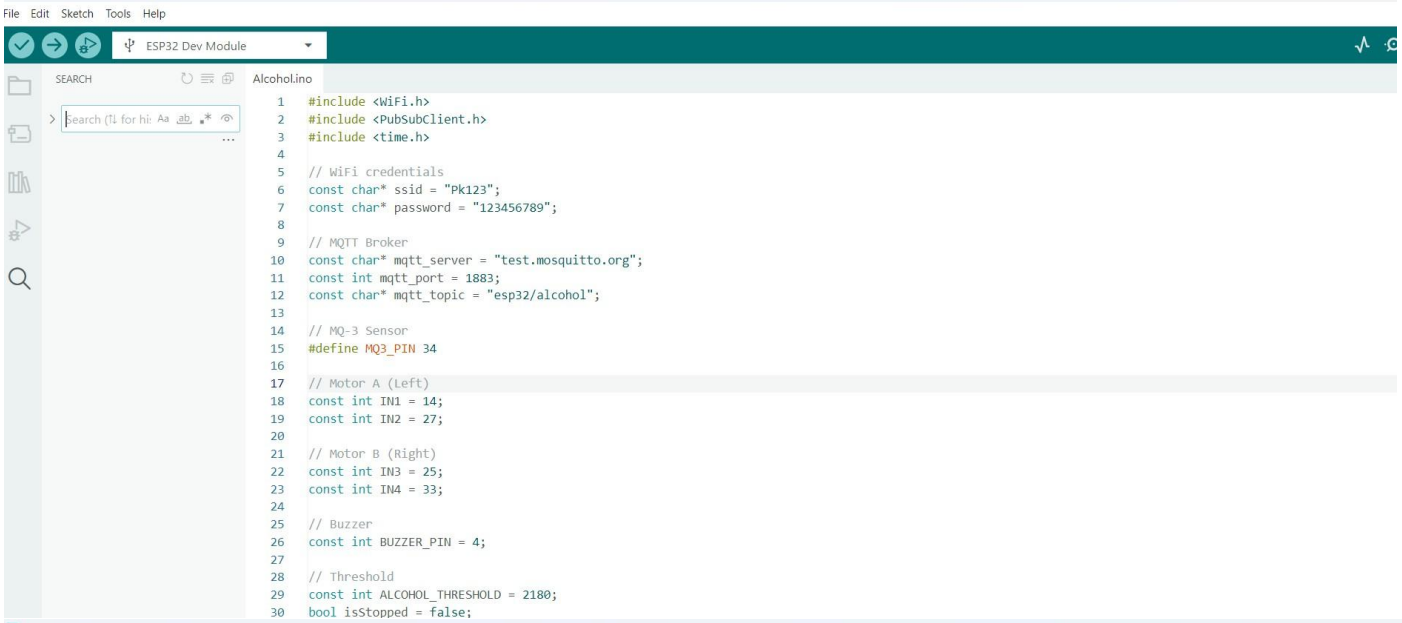
The sequence diagram indicates how the SafeDrive system functions to prevent drunk driving through monitoring in real time and controlling the car automatically. The process begins when the driver blows into the alcohol sensor before starting the vehicle. The sensor detects the amount of alcohol and sends the information to the ESP32 microcontroller. If the alcohol content is less than the allowed level, the system permits the engine to be started; otherwise, it prevents engine ignition so that the vehicle cannot be driven. Once the car has been started, the system constantly monitors the breath of the driver and resets the alcohol level periodically. If the alcohol level is over the driving limit, the system initiates a controlled braking process, warns the driver, and at the same time sends an instant message with the alcohol level, car location, and the driver's details to the law enforcement agency.If the alcohol content falls within a permissible limit, the car runs normal without any obstruction. This chart easily illustrates how various components interact to maintain road safety by monitoring and reacting to the driver's condition in real time

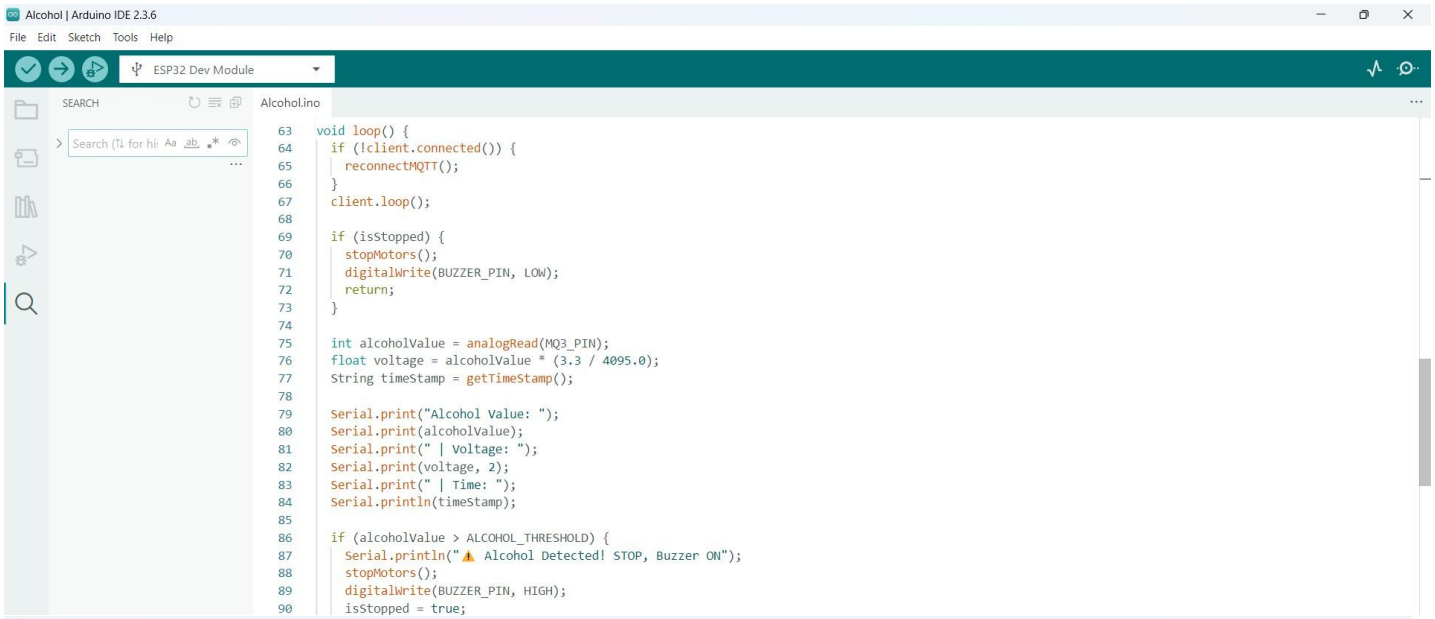
CHAPTER 5

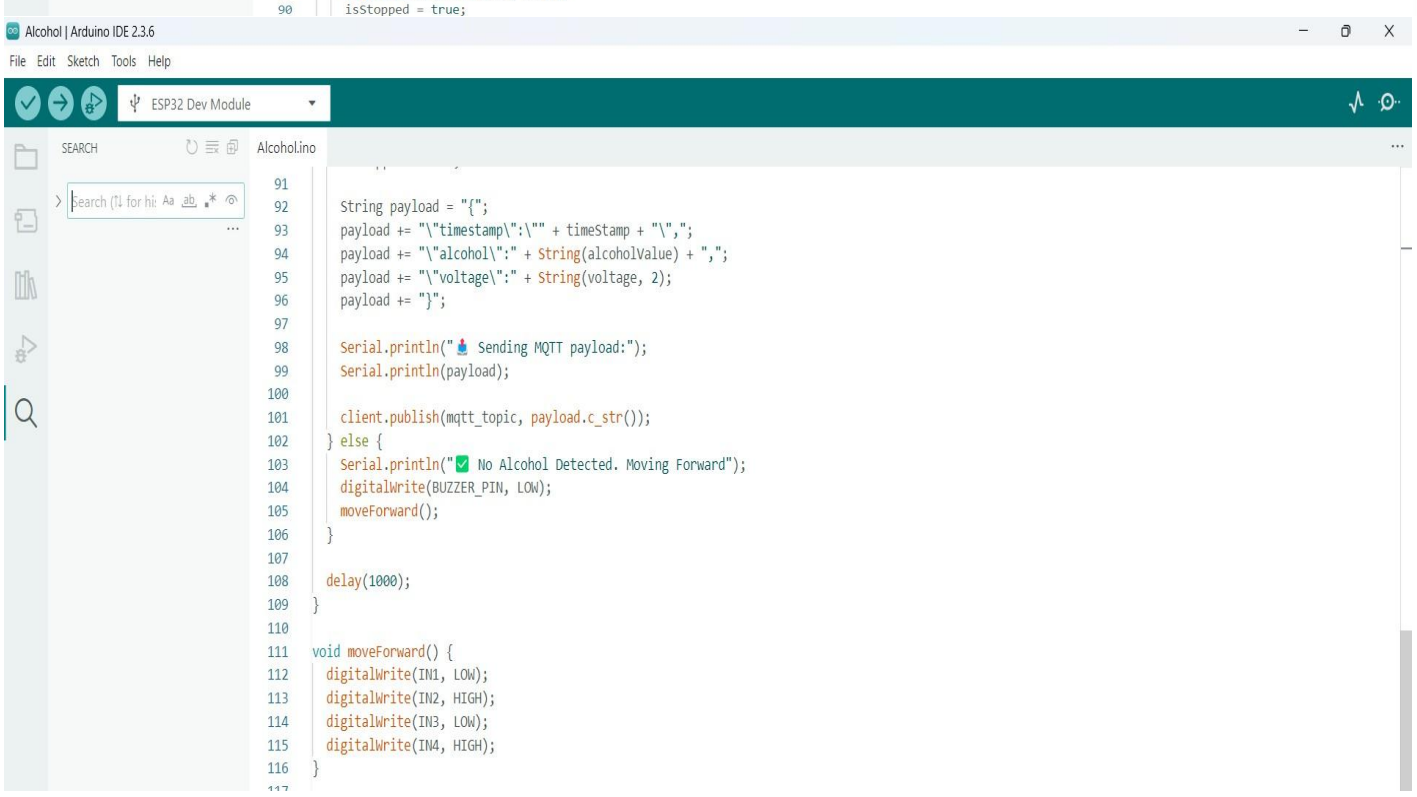
IMPLEMENTATION, EXPERIMENTAL RESULTS & TEST CASES

5.1 Code

5.1.1 Arduino Code

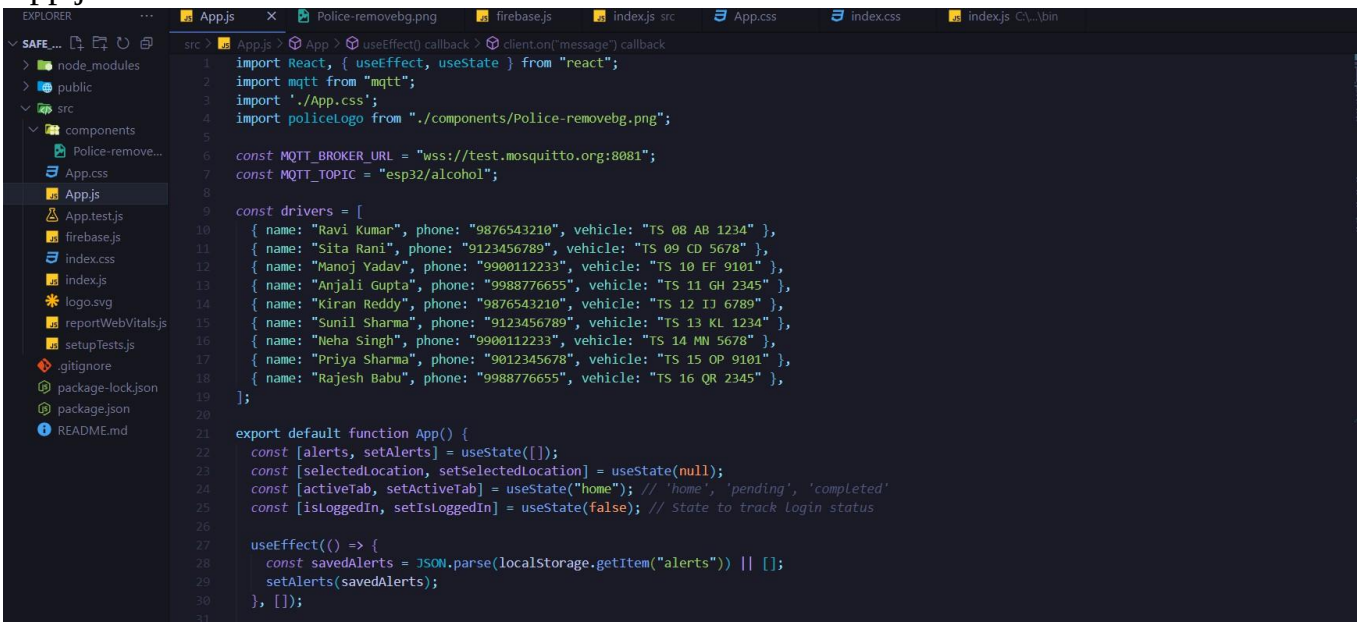


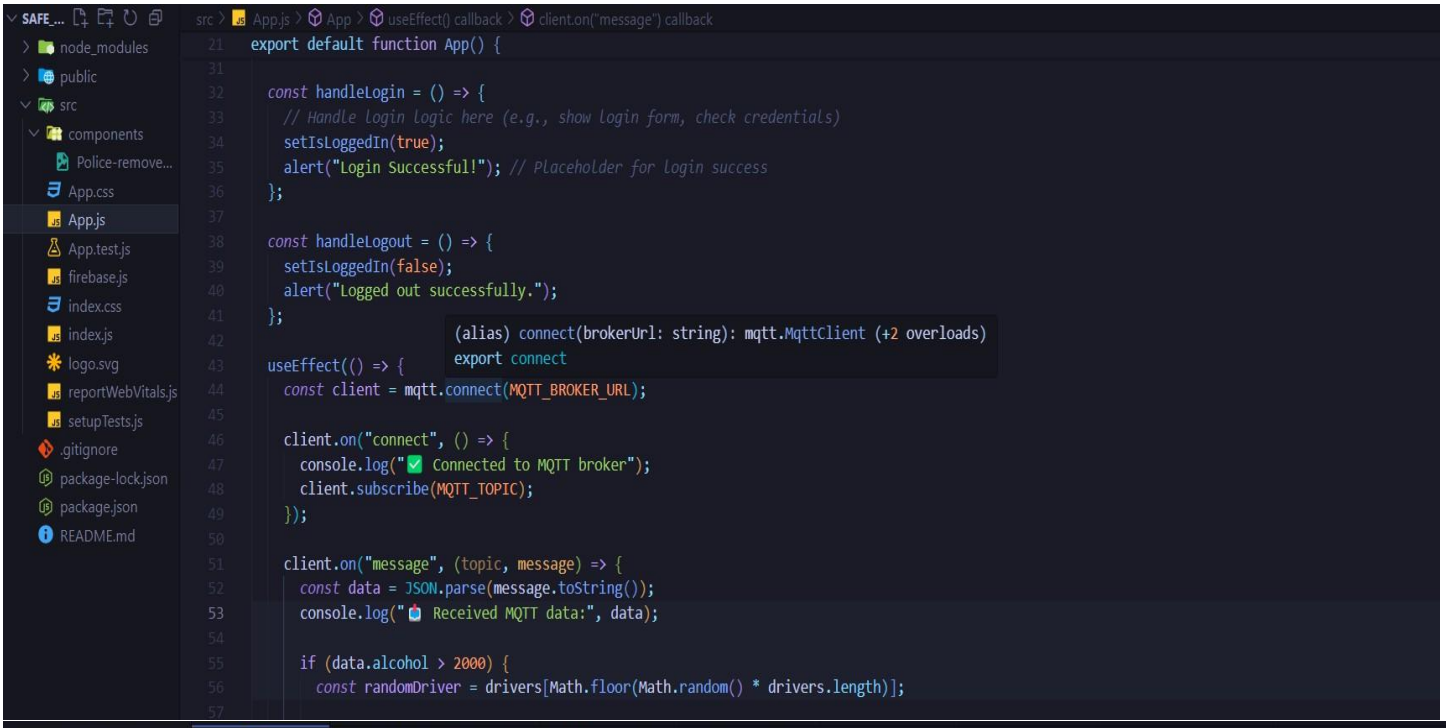

Alcohol | Arduino IDE 2.3.6
File Edit Sketch Tools Help
ESP32 Dev Module
SEARCH
Alcohol.ino
63 void loop() {
64 if (!client.connected()) {
65 reconnectMQTT();
66 }
67 client.loop();
68
69 if (isStopped) {
70 stopMotors();
71 digitalWrite(BUZZER_PIN, LOW);
72 return;
73 }
74
75 int alcoholValue = analogRead(MQ3_PIN);
76 float voltage = alcoholValue * (3.3 / 4095.0);
77 String timeStamp = getTimeStamp();
78
79 Serial.print("Alcohol Value: ");
80 Serial.print(alcoholValue);
81 Serial.print(" | Voltage: ");
82 Serial.print(voltage, 2);
83 Serial.print(" | Time: ");
84 Serial.println(timeStamp);
85
86 if (alcoholValue > ALCOHOL_THRESHOLD) {
87 Serial.println("⚠ Alcohol Detected! STOP, Buzzer ON");
88 stopMotors();
89 digitalWrite(BUZZER_PIN, HIGH);
90 isStopped = true;

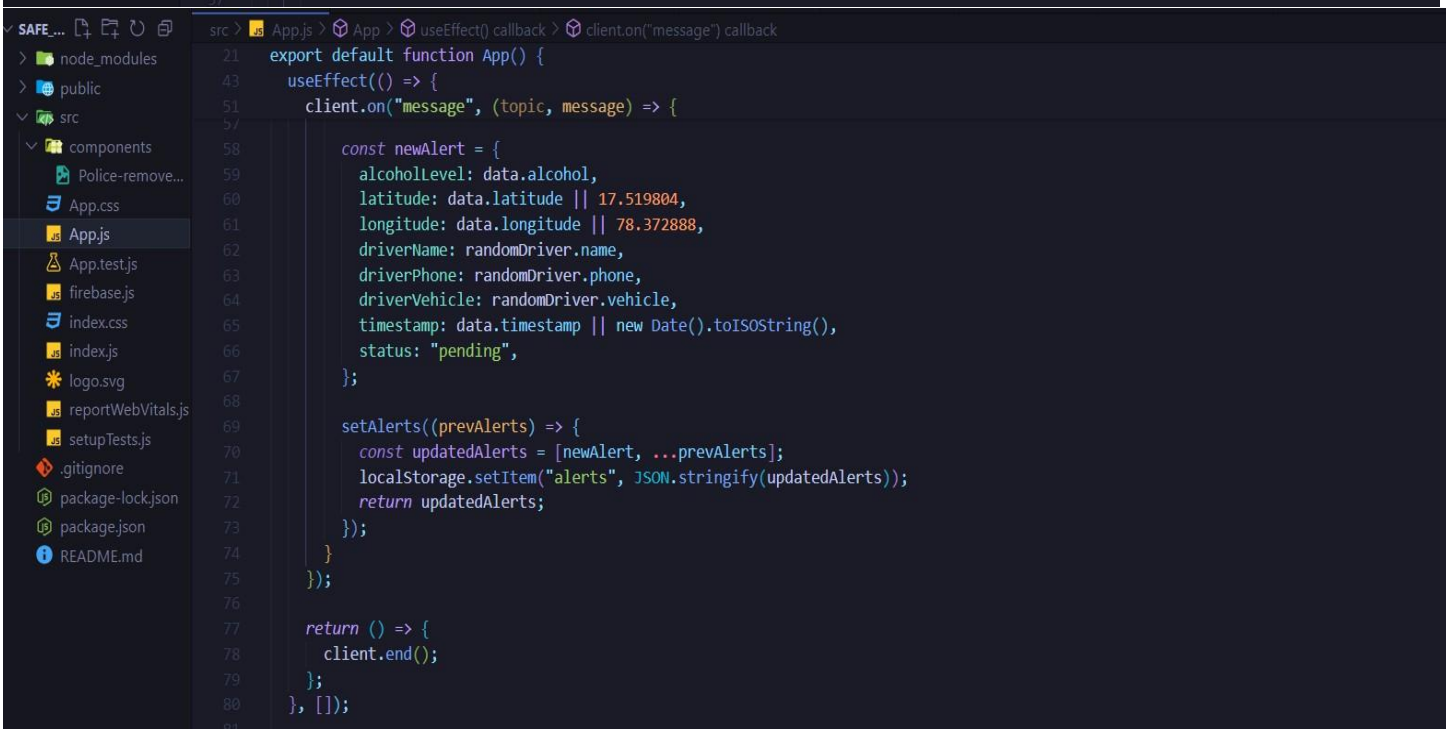

Alcohol | Arduino IDE 2.3.6
File Edit Sketch Tools Help
ESP32 Dev Module
SEARCH
Alcohol.ino
91
92 String payload = "{";
93 payload += "\"timestamp\":\"" + timeStamp + "\",";
94 payload += "\"alcohol\":" + String(alcoholValue) + ",";
95 payload += "\"voltage\":" + String(voltage, 2);
96 payload += "}";
97
98 Serial.println("🚀 Sending MQTT payload:");
99 Serial.println(payload);
100
101 client.publish(mqtt_topic, payload.c_str());
102 } else {
103 Serial.println("✅ No Alcohol Detected. Moving Forward");
104 digitalWrite(BUZZER_PIN, LOW);
105 moveForward();
106 }
107
108 delay(1000);
109 }
110
111 void moveForward() {
112 digitalWrite(IN1, LOW);
113 digitalWrite(IN2, HIGH);
114 digitalWrite(IN3, LOW);
115 digitalWrite(IN4, HIGH);
116 }

5.1.2 Web App Code:

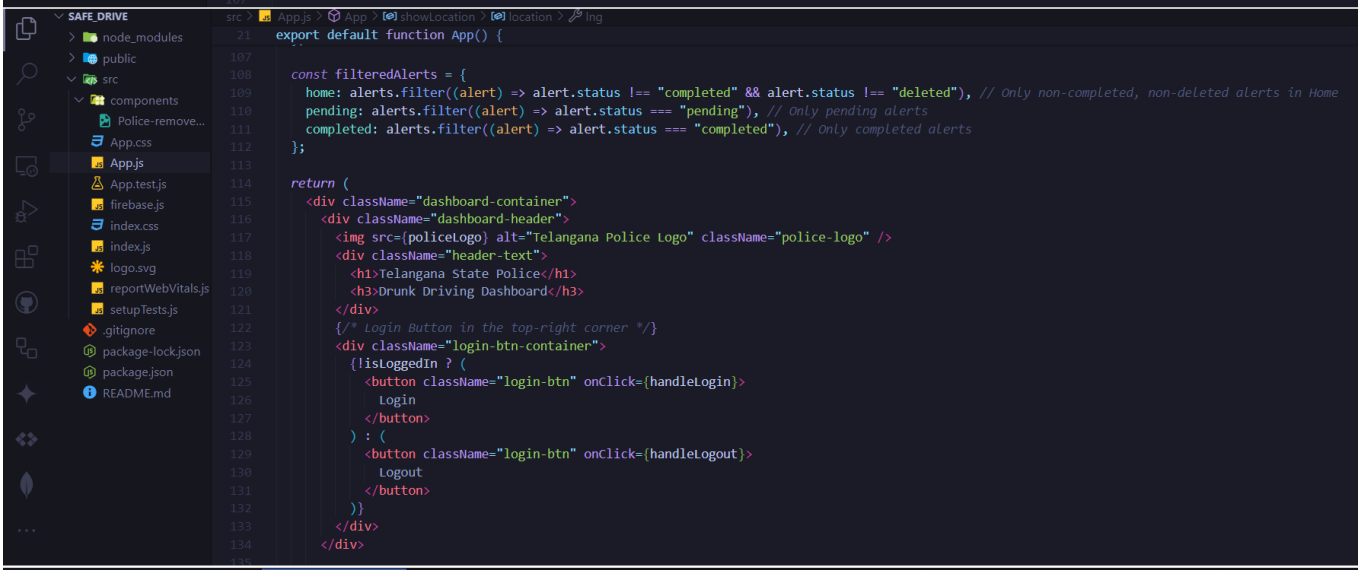
App.js




SAFE_...
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > useEffect() callback > client.on("message") callback
export default function App() {
const handleLogin = () => {
// Handle login logic here (e.g., show login form, check credentials)
setIsLoggedIn(true);
alert("Login Successful!"); // Placeholder for login success
};
const handleLogout = () => {
setIsLoggedIn(false);
alert("Logged out successfully.");
};
(alias) connect(brokerUrl: string): mqtt.MqttClient (+2 overloads)
export connect
useEffect(() => {
const client = mqtt.connect(MQTT_BROKER_URL);
client.on("connect", () => {
console.log("✅ Connected to MQTT broker");
client.subscribe(MQTT_TOPIC);
});
client.on("message", (topic, message) => {
const data = JSON.parse(message.toString());
console.log("📩 Received MQTT data:", data);
if (data.alcohol > 2000) {
const randomDriver = drivers[Math.floor(Math.random() * drivers.length)];


SAFE_...
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > useEffect() callback > client.on("message") callback
export default function App() {
useEffect(() => {
client.on("message", (topic, message) => {
const newAlert = {
alcoholLevel: data.alcohol,
latitude: data.latitude || 17.519804,
longitude: data.longitude || 78.372888,
driverName: randomDriver.name,
driverPhone: randomDriver.phone,
driverVehicle: randomDriver.vehicle,
timestamp: data.timestamp || new Date().toISOString(),
status: "pending",
};
setAlerts((prevAlerts) => {
const updatedAlerts = [newAlert, ...prevAlerts];
localStorage.setItem("alerts", JSON.stringify(updatedAlerts));
return updatedAlerts;
});
}
});
return () => {
client.end();
};
}, []);

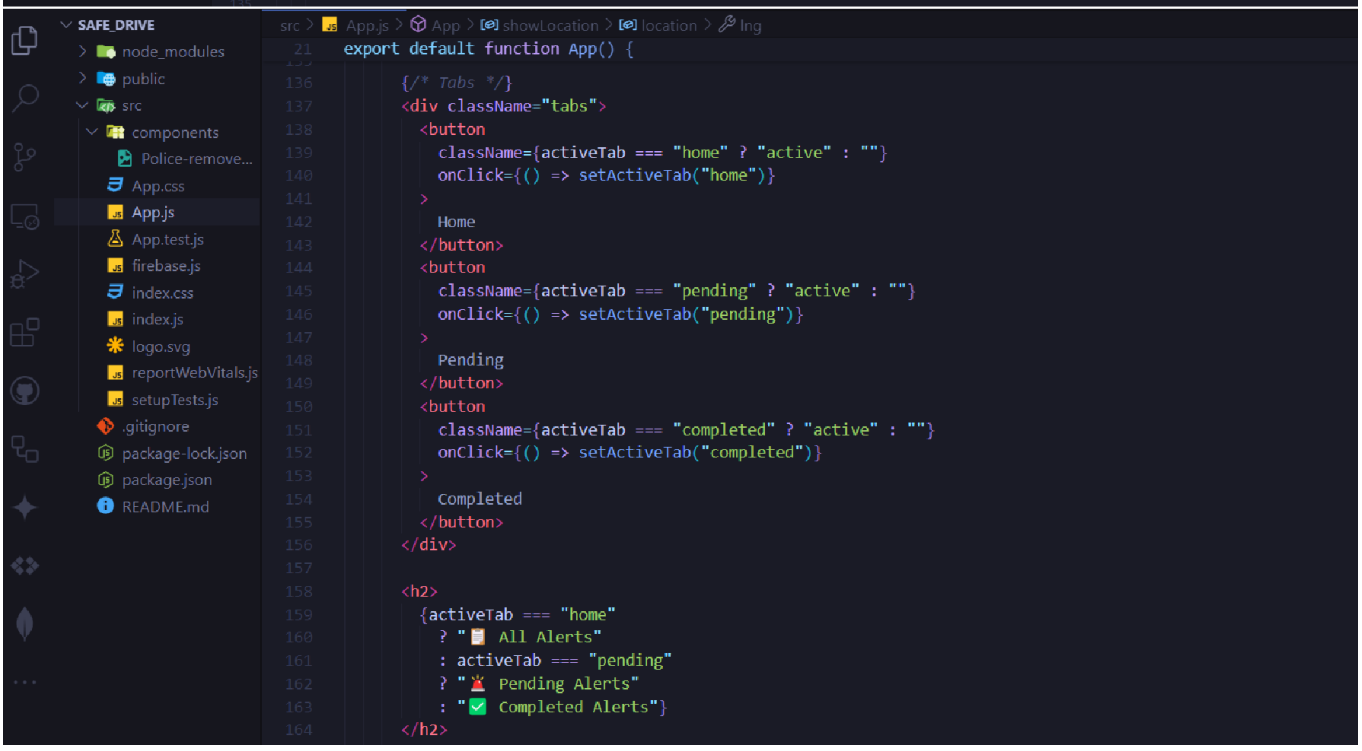

SAFE_DRIVE
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > showLocation > location > lng
export default function App() {
const showLocation = (lat, lng, index) => {
const location = {
lat: data.lattitude || 17.519804,
lng: data.longitude || 78.372888,
};
setSelectedLocation({ ...location, index });
};
const markAsCompleted = (index) => {
const updatedAlerts = alerts.map((alert, i) =>
i === index ? { ...alert, status: "completed" } : alert
);
setAlerts(updatedAlerts);
localStorage.setItem("alerts", JSON.stringify(updatedAlerts));
};
const deleteAlert = (index) => {
const updatedAlerts = alerts.filter((_, i) => i !== index); // Remove the alert completely
setAlerts(updatedAlerts);
localStorage.setItem("alerts", JSON.stringify(updatedAlerts));
if (selectedLocation && index === selectedLocation.index) {
setSelectedLocation(null);
}
};


SAFE_DRIVE
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > showLocation > location > lng
export default function App() {
const filteredAlerts = {
home: alerts.filter((alert) => alert.status !== "completed" && alert.status !== "deleted"), // Only non-completed, non-deleted alerts in Home
pending: alerts.filter((alert) => alert.status === "pending"), // Only pending alerts
completed: alerts.filter((alert) => alert.status === "completed"), // Only completed alerts
};
return (

<h1>Telangana State Police</h1>
<h3>Drunk Driving Dashboard</h3>

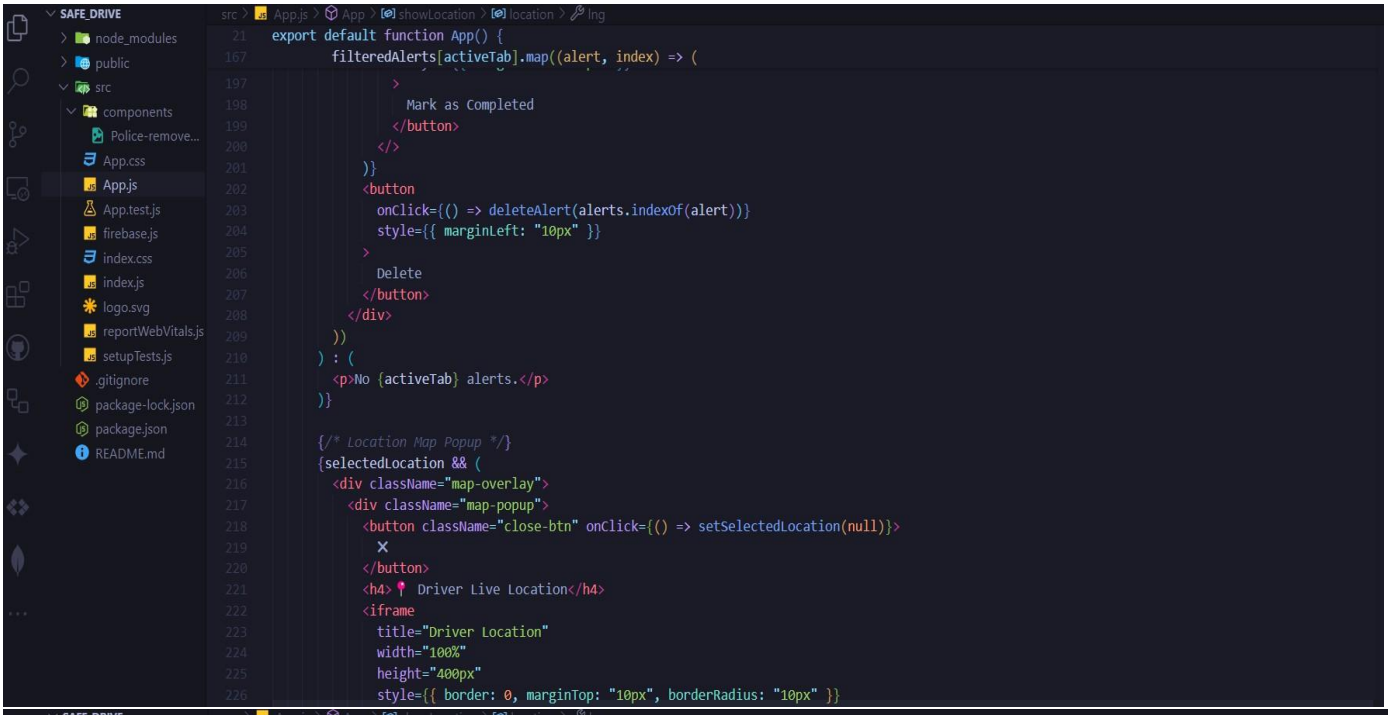
{/* Login Button in the top-right corner */}

{!isLoggedIn ? (
<button className="login-btn" onClick={handleLogin}>
Login
</button>
) : (
<button className="login-btn" onClick={handleLogout}>
Logout
</button>
)}


SAFE_DRIVE
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > showLocation > location > lng
export default function App() {
{/* Tabs */}

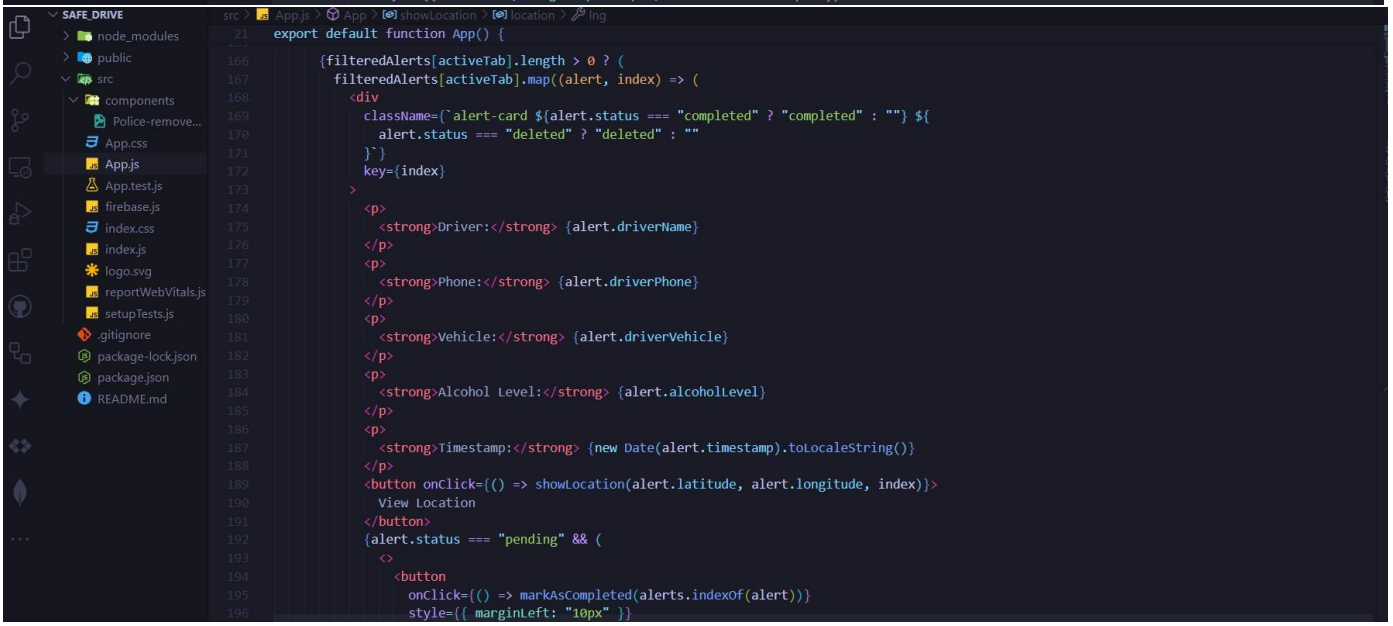
<button
className={activeTab === "home" ? "active" : ""}
onClick={() => setActiveTab("home")}
>
Home
</button>
<button
className={activeTab === "pending" ? "active" : ""}
onClick={() => setActiveTab("pending")}
>
Pending
</button>
<button
className={activeTab === "completed" ? "active" : ""}
onClick={() => setActiveTab("completed")}
>
Completed
</button>

<h2>
{activeTab === "home"
? "📋 All Alerts"
: activeTab === "pending"
? "🚨 Pending Alerts"
: "✅ Completed Alerts"}
</h2>


SAFE_DRIVE
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > showLocation > location > lng
export default function App() {
filteredAlerts[activeTab].map((alert, index) => (
Mark as Completed
</button>
</>
)}
<button
onClick={() => deleteAlert(alerts.indexOf(alert))}
style={{ marginLeft: "10px" }}
>
Delete
</button>

))
) : (
No {activeTab} alerts.
)}
{/* Location Map Popup */}
{selectedLocation && (

<button className="close-btn" onClick={() => setSelectedLocation(null)}>
×
</button>
<h4>📍 Driver Live Location</h4>
<iframe
title="Driver Location"
width="100%"
height="400px"
style={{ border: 0, marginTop: "10px", borderRadius: "10px" }}


SAFE_DRIVE
node_modules
public
src
components
Police-remove...
App.css
App.js
App.test.js
firebase.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json
package.json
README.md
src > App.js > App > showLocation > location > lng
export default function App() {
{filteredAlerts[activeTab].length > 0 ? (
filteredAlerts[activeTab].map((alert, index) => (
<div
className={`alert-card ${alert.status === "completed" ? "completed" : ""} ${
alert.status === "deleted" ? "deleted" : ""
}`}
key={index}
>

Driver: {alert.driverName}

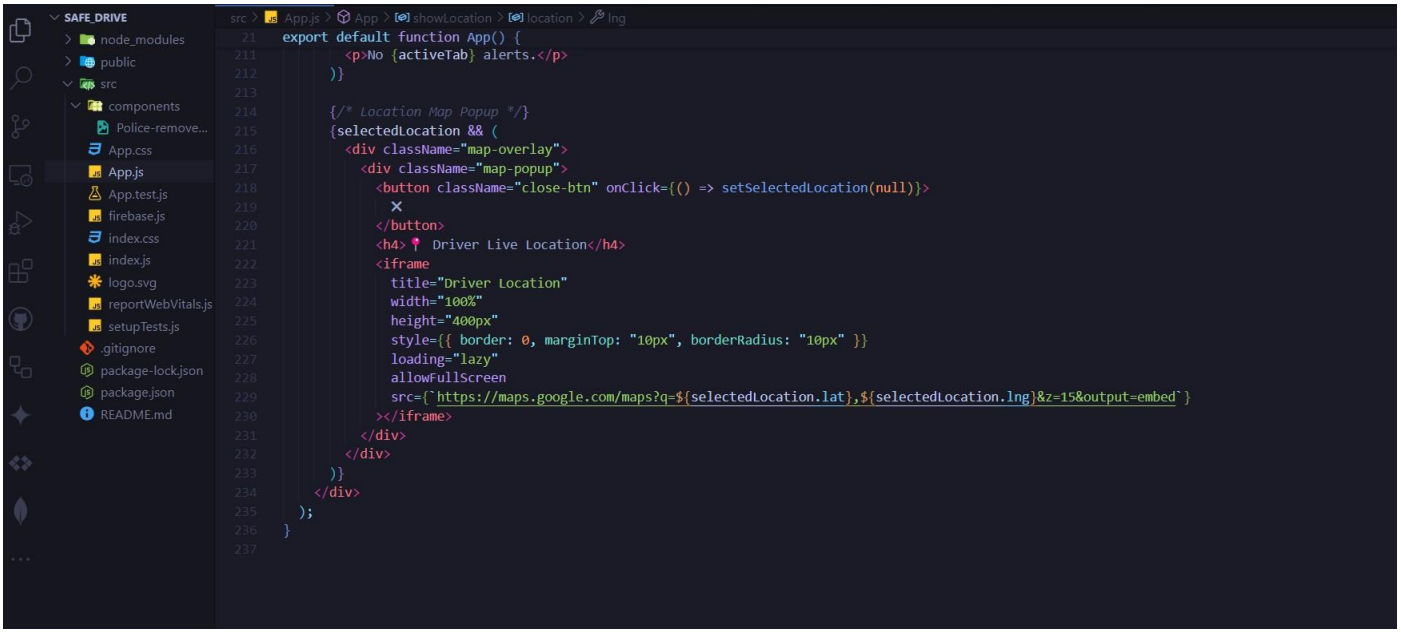
Phone: {alert.driverPhone}

Vehicle: {alert.driverVehicle}

Alcohol Level: {alert.alcoholLevel}

Timestamp: {new Date(alert.timestamp).toLocaleString()}

<button onClick={() => showLocation(alert.latitude, alert.longitude, index)}>
View Location
</button>
{alert.status === "pending" && (
<>
<button
onClick={() => markAsCompleted(alerts.indexOf(alert))}
style={{ marginLeft: "10px" }}



App.css

```css
/* Reset */
body {
  margin: 0;
  padding: 0;
  font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
  background: linear-gradient(to right, #c2d9ec, #e0f0fc); /* light police blue gradient */
}

/* Dashboard Container */
.dashboard-container {
  height: 100vh;
  width: 100vw;
  padding: 20px;
  background-color: #ffffff;
  box-sizing: border-box;
  text-align: center;
  overflow-y: auto;
}

/* Header Styles */
.dashboard-header {
  display: flex;
  align-items: center;
  justify-content: space-between;
  padding: 10px 20px;
  background: linear-gradient(to right, #002855, #b30000);
  color: white;
}
```

```css
.police-logo {
  width: 70px;
  height: auto;
}

.header-text {
  flex: 1;
}

.header-text h1 {
  margin: 0;
  font-size: 28px;
}

.header-text h3 {
  margin: 5px 0 0 0;
  font-weight: normal;
  font-size: 18px;
}

.login-btn-container {
  display: flex;
  justify-content: flex-end;
}
```

```css
.login-btn {
  background-color: black;
  border: none;
  padding: 10px 20px;
  font-size: 16px;
  color: blue;
  cursor: pointer;
  border-radius: 5px;
}

.login-btn:hover {
  background-color: blue;
  color: white;
}

/* Alerts Grid */
.alerts-container {
  display: grid;
  grid-template-columns: repeat(3, 1fr); /* 3 alerts per row on desktop */
  gap: 20px;
  width: 100%;
  max-width: 1200px;
  margin: 20px auto;
  padding: 10px;
  box-sizing: border-box;
}
```

```css
/* Alert Card */
.alert-card {
  background: linear-gradient(to right, #dbeeff, #f2f9ff);
  border: 1px solid #8aaedf;
  border-left: 6px solid #b30000;
  padding: 15px 20px;
  border-radius: 10px;
  text-align: left;
  transition: box-shadow 0.3s ease;
}

.alert-card:hover {
  box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);
}

.alert-driver {
  font-size: 20px;
  font-weight: bold;
  color: #002855;
  text-transform: uppercase;
  margin-bottom: 10px;
}

.alert-card p {
  margin: 6px 0;
  font-size: 15px;
  color: #002855;
}

.alert-card strong {
  color: #000;
}

.alert-card.completed {
  background-color: #e8f5e9;
  border-left: 6px solid green;
}

/* Buttons */
button {
  margin-top: 10px;
  padding: 8px 14px;
  border: none;
  border-radius: 8px;
  font-weight: bold;
  cursor: pointer;
  transition: background-color 0.3s;
  display: inline-block;
}

.alert-card button {
  margin-right: 10px;
}

/* View Location Button - Police Blue */
button:first-of-type {
  background-color: #002855;
  color: white;
}

button:first-of-type:hover {
  background-color: #1b3a66;
}
```

```css
/* Delete Button - Deep Red */
button:last-of-type {
  background-color: #c30010;
  color: white;
}

button:last-of-type:hover {
  background-color: #9e000d;
}

/* Tabs */
.tabs {
  display: flex;
  justify-content: center;
  margin: 20px 0;
}

.tabs button {
  padding: 10px 20px;
  border: none;
  background: #ddd;
  cursor: pointer;
  margin: 0 5px;
  border-radius: 5px;
  font-weight: bold;
}

.tabs .active {
  background-color: #007bff;
  color: white;
}
```

```css
.tabs button:hover {
  background-color: #0056b3;
  color: white;
}

/* Map Overlay */
.map-overlay {
  position: fixed;
  top: 0;
  left: 0;
  width: 100vw;
  height: 100vh;
  background-color: rgba(0, 0, 0, 0.7);
  display: flex;
  justify-content: center;
  align-items: center;
  z-index: 1000;
}

.map-popup {
  background: white;
  padding: 20px;
  width: 90%;
  max-width: 800px;
  border-radius: 12px;
  box-shadow: 0 8px 30px rgba(0, 0, 0, 0.3);
  position: relative;
  animation: fadeIn 0.3s ease;
}
```

```css
.map-container {
  margin-top: 25px;
  text-align: center;
}

.map-container h4 {
  margin-bottom: 10px;
  color: #333;
}

iframe {
  border-radius: 12px;
  border: 0;
  box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
}

/* Close Button */
.close-btn {
  position: absolute;
  top: 10px;
  right: 15px;
  background: #c30010;
  border: none;
  color: white;
  font-size: 18px;
  border-radius: 50%;
  width: 32px;
  height: 32px;
  cursor: pointer;
  transition: background-color 0.2s ease;
}
```

```css
.close-btn:hover {
  background-color: #9e000d;
}

/* Animation */
@keyframes fadeIn {
  from { transform: scale(0.9); opacity: 0; }
  to { transform: scale(1); opacity: 1; }
}

/* Responsive Design */
@media (max-width: 992px) {
  .alerts-container {
    grid-template-columns: repeat(2, 1fr); /* 2 alerts per row on tablet */
  }
}

@media (max-width: 600px) {
  .alerts-container {
    grid-template-columns: 1fr; /* 1 alert per row on mobile */
  }
}
```

5.2 Experimental Results

Hardware Setup

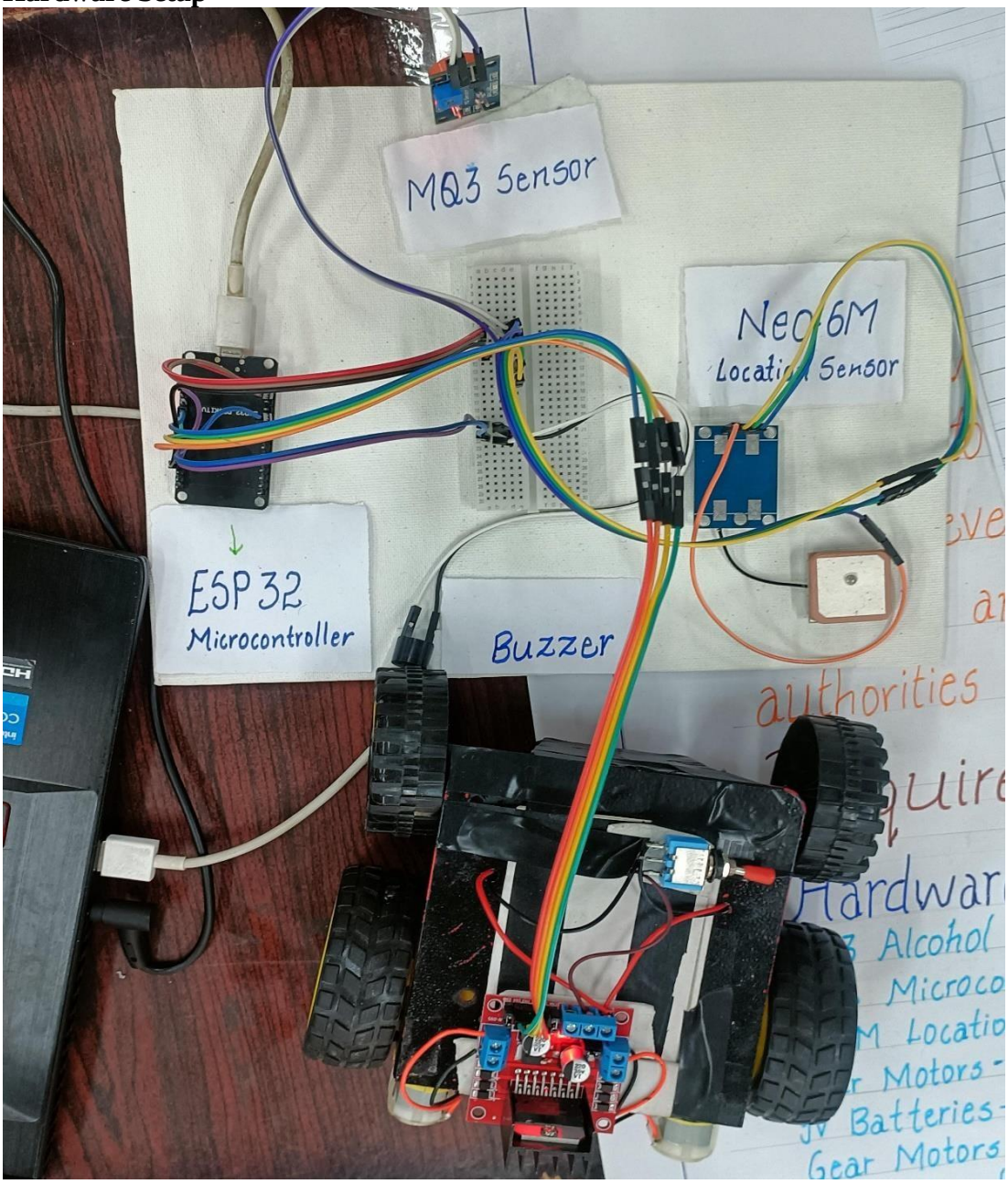


Fig 5.2.1: Hardware Setup

MQ3 Alcohol Sensor- The sensor will constantly check the atmosphere for alcohol particles. Once it detects alcohol, it provides an alert in the system. This is important in detecting intoxication in people or situations like vehicles or public places. Neo6M GPS Module- After the alcohol event is done processing, the GPS module tracks the live activity location of the system. This location data can potentially be sent to authorities and family/emergency contacts to help them track the event for a rapid response.

ESP32 Microcontroller- The ESP32 will connect everything and be the controller. The ESP32 will acquire data from the sensor and be responsible with what to do with that information such as activating the buzzer whilst also transmitting the location data via Wi-Fi or Bluetooth to a connect device or cloud data.Buzzer- If alcohol is detected by the sensor, the buzzer works to provide a noise for nearby people or authorities to hear and warn them. This gives an immediate feedback loop to let others know a danger has been detected.

Motor Driver- A device for a motor driver will be a motor driver like the L298N or L293D which drives gear motors. The motor driver interfaces between the low-power ESP32 and high-power motors. This interface will allow the system to move the vehicle responding to detecting alcohol.

Gear Motors- DC gear motors will drive the wheels of the robotic platform. The gear motors will provide the torque and rotation to move the platform forward, backward, or to turn. The gear motors will connect to the motor driver. The motor driver will be controlled by the ESP32 which will process the measurements taken by the MQ3 sensor.

Web Application:

Fig 5.2.2: Dash Board

The SafeDrive web app is primarily built for authorities to view and manage real-time data for alcohol detection in vehicles. The web app allows law enforcement or safety personnel to be alerted immediately any time that there is any detection of alcohol detected through the smart alcohol detection system. The app has a dashboard feature which gives authorities clear data such as GPS location of the vehicle, the alcohol detection status, and previous incidents, which makes it easy to respond quickly and avoid the drinking and driving.

In addition to real time incident alerts, the web app allows authorities to keep track of trends, safety records

and to see all the previous detections of a certain number of incidents. The app will also communicate directly with backend integrations for storage and processing of the data collected from the system, so the critical information is available to help authorities make critical decisions. The system is an effective tool because it can support the coordination of responses to alcohol detection (e.g., dispatching emergency services, flagging notifications to concerned parties) to help ensure roadways are safe.

5.3 Test Cases:

Test Case1: Alcohol Detection and Vehicle Stop or Not

Test No.	Alcohol Level (Analog)	Vehicle Speed (Kmph)	Stop/ No	Time of test	Comments
1.	2230	40	Stop	10:00 AM	Vehicle Has stopped.
2.	1800	60	No	11:00 AM	Vehicle Has Not stopped.
3.	2200	45	Stop	11:05 AM	Vehicle Has stopped.
4.	1900	55	No	11:10 AM	Vehicle Has Not stopped.
5.	2210	60	Stop	11:35 AM	Vehicle Has stopped.

Table 5.3.1: Alcohol Detection and Vehicle Stop or Not

Test Case2: Remote Monitoring by Law Enforcement

Test No.	Alcohol Level (Analog)	GPS Coordinates (lat , long)	Cloud Data sent (Yes/no)	Time of test	Comments
1.	2230	(17.386052, 78.487705)	Yes	10:00 AM	Data sent successfully.
2.	1800	(17.386052, 78.487705)	No	11:00 AM	No data sent, alcohol levels are safe.
3.	2200	(17.386052, 78.487705)	Yes	11:05 AM	Data sent successfully.
4.	1900	(17.386052, 78.487705)	No	11:10 AM	No data sent, alcohol levels are safe.
5.	2210	(17.386052, 78.487705)	Yes	11:35 AM	Data sent successfully.

Table 5.3.2: Remote Monitoring by Law Enforcement

CHAPTER 6
CONCLUSION AND FUTURE SCOPE

Conclusion

The SafeDrive system offers an intelligent and preventive solution to preventing drunk driving accidents by combining IoT technology with alcohol sensing and car control systems. Through the utilization of devices such as the MQ-3 sensor, ESP32 microcontroller, and NEO-6M GPS module, the system ensures constant monitoring of the driver's alcohol level and initiates preventive action before accidents can happen. The integration of the web application within SafeDrive provides the authorities with the ability to trace the car's location simply, monitor alcohol levels, and react quickly as required. Apart from increasing road safety, the technology also reunites police and technology for cooperative action. SafeDrive is essentially a smart, affordable, and reliable solution that operates to prevent accidents and encourage safe driving.

Future Scope

in the future, SafeDrive can also be made intelligent by directly integrating it with a car's Electronic Control Units (ECUs) so that more efficient and precise control. Adding face recognition will be able to ensure that only trusted, authorized individuals can start and drive the car. The system also is upgradeable to automatically alert parents or other designated emergency contacts of alcohol consumption or dangerous driving behavior, enabling relatives to be notified and even in some cases forestalling life-altering incidents. A dedicated mobile application would provide real-time notification, driving records, and control options for both drivers and the authorities. Also, with AI and predictive analytics, it can detect risky driving habits based on behavior and alcohol consumption data over time. Partnerships with government agencies and police can translate to broader usage of SafeDrive in traffic systems and public transportation, ensuring road safety on a large scale.

CHAPTER 7

REFERENCES

[1] Pradeep Kumar G , Neelam Sanjeev Kumar, and Vijay Kumar P, "Safe transportation system using IoT based alcohol detection," in 2023 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE), Bengaluru, India, 2023, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/10142338

[2] V. Sonule and R. Raman, "In-car breathalyzer systems for enhanced road safety through SVM classification and IoT connectivity," in Proc. 2024 Int. Conf. Advances in Modern Age Technologies in Engineering (AMATHE), Pune, India, May 2024, pp. 506–511. [Online]. Available: https://ieeexplore.ieee.org/document/10142338

[3] N. G. Iyer, M. Arulmozhi, P. Sivakumar, S. Sudharsan, S. Jeny Sophia, and R. Kavitha, "AI-powered driver behavior prediction, drunk driving prevention, accident detection, and insurance integration," in 2023 International Conference on Energy, Materials and Communication Engineering (ICEMCE), Madurai, India, Dec. 2023, pp. 688–693. [Online]. Available: https://colab.ws/articles/10.1109%2Ficemce57940.2023.10434027

[4] V. Malarselvam, A. James Ashish, D. Peranban, S. Arun, and P. Arulmani, "Drunk driving prevention and automatic bus pass tracking by using image processing," *Department of Electrical and Electronics Engineering, Sri Manakula Vinayagar Engineering College*, Puducherry, India, Jul. 2023. [Online]. Available: https://www.researchgate.net/publication/372139124

[5] S. A. Nordin, Z. M. Yusoff, and N. N. Mohammad, "Druken alcohol intelligent detection system IoT based Arduino controller," *School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA*, Segamat, Malaysia, and *Microwave Research Institute, Universiti Teknologi MARA*, Shah Alam, Malaysia, Mar. 2023. [Online]. Available: https://www.researchgate.net/publication/368885194_Druken_alcohol_intelligent_detection_system_IoT_based_Arduino_controller

[6] A. Sinha, N. Kumar, A. Ticku, and M. Shah, "IoT based human biometric system using alcohol level detection for driver security," *School of Computing and Information Sciences, IGNOU*, New Delhi, India; *Department of Computer Science, Lovely Professional University*, Punjab, India; *Bharti Vidyapeeth's College of Engineering*, India; *Department of Computer Science & Engineering, Charotar University of Science & Technology*, Gujarat, India, Dec. 2023. [Online]. Available: https://www.researchgate.net/publication/379842814

CHAPTER 8
PLAGARISM REPORT

iThenticate Page 1 of 48 - Cover Page

CSE-A15 GRIET

A15_Pcheck_doc.pdf

Gokaraju Rangaraju Institute of Engineering and Technology

Document Details

Submission ID
trn:oid:::3618:94523693

Submission Date
May 6, 2025, 12:39 PM GMT+5:30

Download Date
May 6, 2025, 12:40 PM GMT+5:30

File Name
A15_Pcheck_doc.pdf

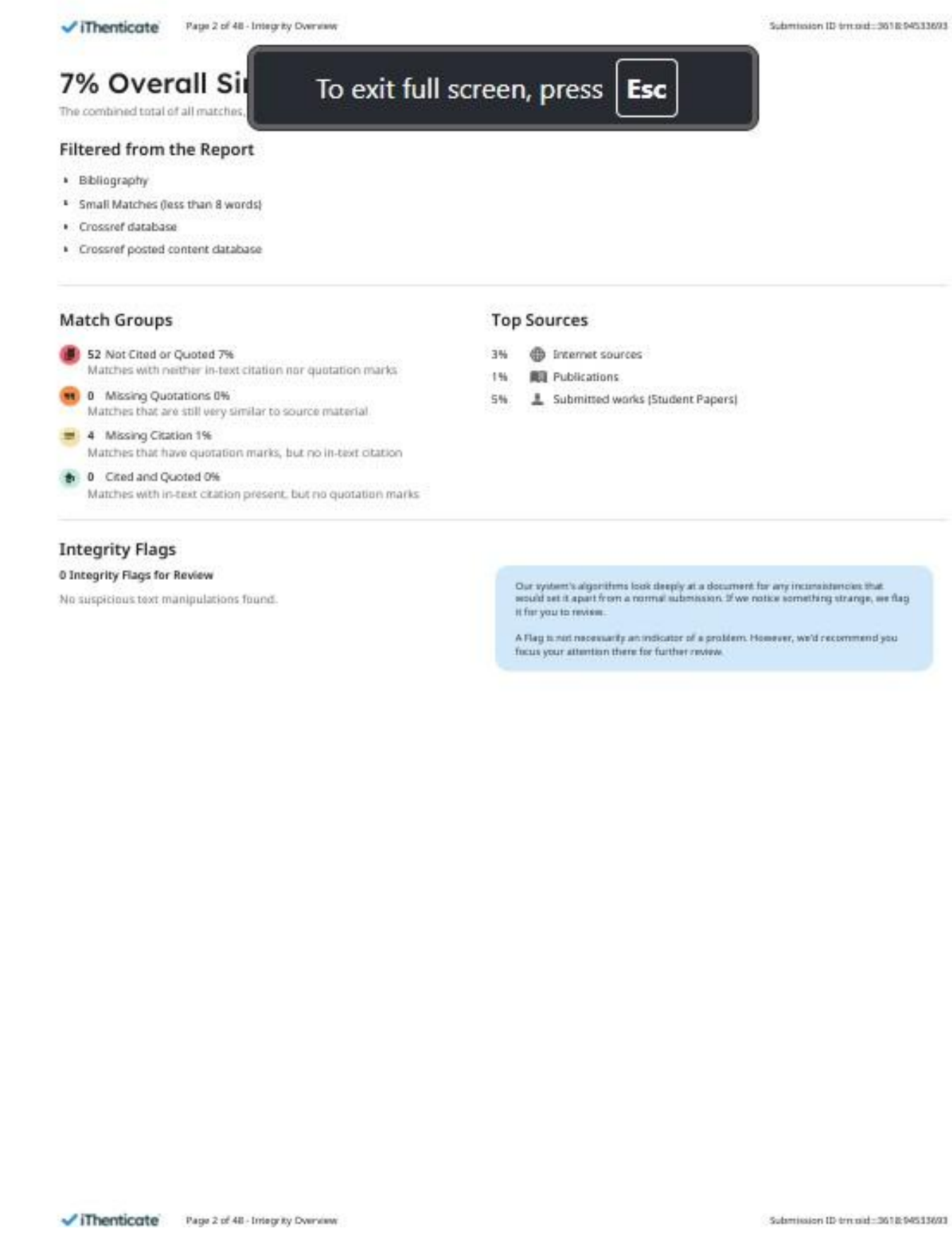
File Size
2.3 MB

42 Pages

7,918 Words

42,969 Characters

iThenticate Page 1 of 48 - Cover Page


iThenticate
Page 2 of 48 - Integrity Overview
Submission ID trn:oid:::3618:94533693
7% Overall Si
To exit full screen, press Esc
The combined total of all matches,
Filtered from the Report
Bibliography
Small Matches (less than 8 words)
Crossref database
Crossref posted content database
Match Groups
52 Not Cited or Quoted 7%
Matches with neither in-text citation nor quotation marks
0 Missing Quotations 0%
Matches that are still very similar to source material
4 Missing Citation 1%
Matches that have quotation marks, but no in-text citation
0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks
Top Sources
3% Internet sources
1% Publications
5% Submitted works (Student Papers)
Integrity Flags
0 Integrity Flags for Review
No suspicious text manipulations found.
Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.
A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.
iThenticate
Page 2 of 48 - Integrity Overview
Submission ID trn:oid:::3618:94533693

Match Groups

- 52 Not Cited or Quoted 7%
  Matches with neither in-text citation nor quotation marks
- 0 Missing Quotations 0%
  Matches that are still very similar to source material
- 4 Missing Citation 1%
  Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

3% Internet sources
1% Publications
5% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	
pluginhighway.ca		<1%
2	Submitted works	
Jabatan Pendidikan Politeknik Dan Kolej Komuniti on 2025-05-04		<1%
3	Submitted works	
Alliance University on 2024-11-12		<1%
4	Internet	
www2.mdpi.com		<1%
5	Internet	
www.scilit.net		<1%
6	Publication	
Arvind Dagur, Dhirendra Kumar Shukla, Nazarov Fayzullo Makhmadiyarovich, Ak...		<1%
7	Submitted works	
Jabatan Pendidikan Politeknik Dan Kolej Komuniti on 2024-11-06		<1%
8	Submitted works	
ABES Engineering College on 2023-05-29		<1%
9	Internet	
rindx.com		<1%
10	Submitted works	
Alliance University on 2025-02-22		<1%

iThenticate Page 3 of 48 - Integrity Overview Submission ID trn:oid:::3618:945133693

CHAPTER 9

APPENDIX

9.1 Research Paper

SafeDrive: Preventing Drunk Driving with IoT Technology

Dr K Anuradha, Md Abdul Abed , Ambati Saiteja , Gonewar Pavan Kumar, Cheepelly Siddhartha
Department of CSE, GRIET, Hyderabad, India.

***Abstract*—Drunk driving is a serious issue, resulting in many accidents regardless of strict regulations and public education campaigns. Classical approaches, such as police roadside breathalyzer screening, identify only drunk drivers when they are on the road. To avoid accidents of this sort, our project presents SafeDrive, an intelligent alcohol detection system using IoT technology. This device integrates the MQ-3 alcohol sensor and an ESP32 microcontroller to continuously monitor the driver's breath for alcohol level. The sensor can be mounted on the seatbelt or wherever else near the driver to efficiently detect. When the alcohol level is above the allowed level before the car is allowed to start, the system will not let the car start. But if the driver drinks and drives and crosses the limit, the system decelerates the vehicle slowly rather than abruptly stopping it, making it safe. Also, the ESP32 provides realtime data transmission, and with a special application, law enforcement officers can monitor the live location of the vehicle, alcohol content, and driver information. This allows the authorities to take action accordingly prior to the occurrence of the accident. Through the incorporation of IoT technology, real-time tracking, and controlled vehicle response, SafeDrive presents a productive and proactive means of curbing accidents caused by drunk driving, in turn encouraging road safety**

I. INTRODUCTION

Drunk driving continues to be one of the most serious dangers on our roads, causing thousands of accidents, injuries, and tragic losses every year—many of which could have been easily prevented. Despite tough laws and constant awareness campaigns, people still get behind the wheel after drinking, often with devastating consequences. The problem with traditional methods, like roadside breath tests, is that they detect the danger only after it’s already in motion—when it may be too late.

That’s exactly why we developed SafeDrive—a smart, IoT-based system that aims to stop drunk driving

before it even begins. At the heart of SafeDrive is an MQ3 alcohol sensor connected to an ESP32 microcontroller. This compact setup is placed close to the driver—like on the seatbelt—and constantly monitors their breath for signs of alcohol. If it detects levels beyond the legal limit, the system steps in automatically: either stopping the engine from starting or, if the vehicle is already moving, safely slowing it down to avoid a sudden stop. But that's not all. SafeDrive is also equipped with a GPS module that tracks the vehicle's location in real time. All this data—alcohol levels, location, alerts—is securely uploaded to the cloud. From there, it can be monitored by authorities through a live dashboard, helping them respond quickly and effectively. In short, SafeDrive is more than just a device—it's a proactive safety solution that blends technology with responsibility. Its goal? To save lives and make our roads safer, one smart decision at a time.

II. **L**iterature **r**eview

With the rapid evolution of IoT technologies, road safety solutions are becoming smarter and more proactive. One promising area is the use of alcohol detection systems integrated directly into vehicles—these systems aim to prevent drunk driving before it becomes a threat. By combining breath analysis, wireless communication, and cloud-based monitoring, researchers are working to build intelligent safety frameworks.

In [1], researchers introduced an IoT-based system that tracks alcohol levels in real-time using biometric sensors. If the driver's breath contains too much alcohol, the system blocks the vehicle from starting. While it includes features like cloud storage and automated control, the system's performance can be impacted by issues like poor weather conditions, blocked sensors, or sensor aging.

Similarly, [2] proposes a lightweight setup using an ESP8266 and an MQ-3 sensor, capable of locking the engine and sending alerts to family or authorities. It's affordable and practical, but vulnerable to tampering and occasionally experiences delays due to weak network signals. In [3], a more versatile system is presented, combining alcohol detection with anti-theft features using GPS and GSM. While it adds extra functionality, its accuracy is hindered by frequent false alarms and unreliable communication links.

Other approaches, like [4], stick strictly to BAC detection but miss out on integrating additional safety tech like automatic braking or fatigue alerts. Reference [5] adds value in public transport by combining alcohol sensing with image-based passenger verification, yet it doesn't adequately handle calibration or interference issues that affect sensor reliability over time.

A more advanced direction is taken in [6], where AI is used to not only detect alcohol but also understand driving behavior patterns and even link incidents with insurance processes. While this approach is powerful, the study overlooks data privacy concerns, which are crucial in a system that processes real-time personal information.

In [7] and [8], lightweight machine learning models like SVMs are used to improve detection accuracy with minimal hardware, making them ideal for commercial vehicles. However, both lack proper user feedback mechanisms and instant alerting capabilities. Reference [9] adds speed control features using Arduino and cloud-based tools like Blynk, but misses key functionalities like accurate location tracking or GPRS integration.

Finally, [10] stands out by using multiple sensors and a shallow neural network to boost detection reliability. Though more accurate, the system still struggles with environmental factors like humidity or air pollution, which can distort readings.

Gaps Identified

- Sensor Sensitivity: Many depend heavily on MQ-3 sensors, which are easily affected by humidity, smoke, and strong odors.
- Easy to Bypass: Some systems can be tricked using external airflow or tampering with sensor placement.
- Slow Alerting: Alert delays are common due to weak GSM/Wi-Fi or lack of backup channels.
- Limited Intelligence: Few solutions use AI to predict or analyze driver behavior or filter out false

positives.

- Narrow Safety Focus: Most systems only detect alcohol but don't integrate with other safety features like fatigue monitoring or emergency braking.
- Privacy Risks: Encryption and data security are often overlooked, leaving personal information vulnerable.
- Environmental Limitations: Few systems compensate for real-world variables like temperature or air quality.
- Scalability: Many are tested in lab settings and not designed for deployment on a large scale or in diverse environments

III. PROPOSED SYSTEM

This system integrates an MQ3 sensor, ESP32 microcontroller, GPS module, and cloud storage to create an alcohol detection and vehicle control mechanism for enhanced road safety. The MQ3 sensor detects alcohol levels in the driver's breath, and the ESP32 microcontroller processes this data. If alcohol is detected above a threshold, the microcontroller signals the motor driver to control the engine, preventing the vehicle from starting. Simultaneously, the GPS module captures the vehicle's location, which, along with the alcohol detection data, is transmitted to cloud storage. This information is then accessible via a police dashboard, providing vehicle details and location for monitoring and enforcement.

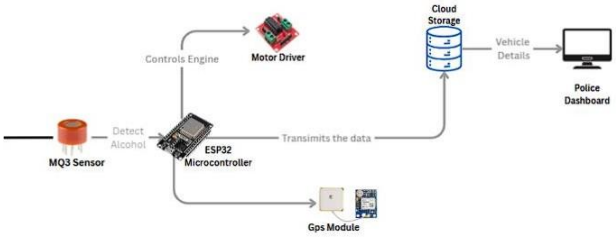


Fig.1. System Architecture

1) Alcohol Detection & Engine Control Flow:

- MQ3 Sensor – Alcohol Detection:
  The system starts by continuously monitoring the driver's breath using an MQ3 alcohol sensor. It is positioned near the driver, ideally on the seatbelt or dashboard.
- ESP32 Microcontroller – Core Processor:
  The ESP32 reads analog values from the MQ3 sensor. If the alcohol level crosses a pre-set threshold, it takes immediate action.
- Motor Driver – Vehicle Control:
  Based on the input from the ESP32, the motor driver module either prevents the engine from starting or initiates a safe and gradual stop if the vehicle is already in motion.

2) Location Tracking & Data Transmission Flow:

GPS Module – Real-time Location:
The GPS module captures the vehicle's location coordinates and feeds them to the ESP32 microcontroller.

Cloud Storage – Centralized Data Upload:
The ESP32 transmits both alcohol level and GPS data to a cloud-based database using Wi-Fi/MQTT protocol. This allows secure and centralized logging.

3) Monitoring and Alerting Flow:

Police Dashboard – Remote Monitoring:
Authorized personnel can access vehicle details, driver status, and location data through a real-time dashboard. This facilitates prompt action and law enforcement intervention if needed.

II. IMPLEMENTATION

The SafeDrive system brings together various technologies like the Internet of Things (IoT), sensors, cloud services, and vehicle control mechanisms to help prevent drunk driving. By detecting alcohol in a driver's breath, the system can either stop the vehicle from starting or slow it down safely if it's already on the move. Both hardware and software components work hand-in-hand to make this real-time safety solution possible.

A. Hardware Implementation

1. MQ3 Alcohol Sensor

At the heart of alcohol detection is the MQ3 sensor, placed near the driver—usually on the seatbelt or dashboard. It continuously monitors the air for traces of alcohol. When alcohol is detected, the sensor sends an analog signal that reflects the concentration level to the system's controller.

2. ESP32 Microcontroller

The ESP32 serves as the brain of the system. It takes the signal from the MQ3 sensor and checks whether the alcohol level is within the safe limit.

- If the level is too high and the car hasn't started yet, the system prevents the ignition.
- If the car is already moving, the system gradually slows it down to avoid sudden braking.

Besides this, the ESP32 also gathers GPS data and sends everything—alcohol readings, location, and alerts—to the cloud for monitoring.

3. GPS Module

The Neo 6M GPS module helps track the vehicle's real-time location. Once alcohol is detected, the system records the coordinates and sends them to the monitoring dashboard so authorities can see where the incident

happened.

4. Motor Driver and Gear Motors

The system uses an L298N motor driver to manage the vehicle's movement through gear motors. If alcohol is detected while driving, the ESP32 activates this module to gently reduce speed and bring the vehicle to a safe stop.

5. Buzzer

A buzzer gives an immediate audio alert when alcohol levels are too high. This not only warns the driver but also people nearby, making the situation instantly noticeable.

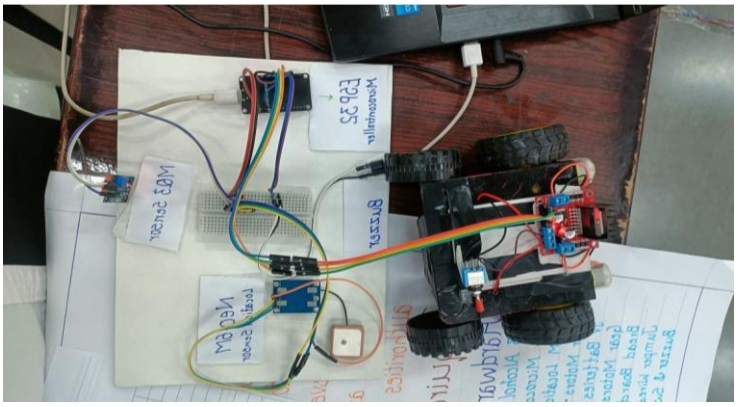


Fig.2. Hardware Setup

Fig. 2. *This image shows the real-world setup of the SafeDrive prototype. The ESP32 microcontroller is wired to the MQ3 alcohol sensor, Neo6M GPS module, buzzer, and motor driver, all mounted on a small vehicle platform. Together, these components work to detect alcohol, track location, and control the vehicle's movement in real time—just like it would in a real driving scenario*

B. Software Implementation

1. Embedded Programming

The SafeDrive hardware is powered by code written in Arduino C++, uploaded via the Arduino IDE. This code handles everything from reading sensor values and controlling motors to pushing data to the cloud over Wi-Fi.

2. Cloud Integration

To keep track of events remotely, the system uses Firebase as its cloud backend. The ESP32 sends data—like alcohol levels and GPS location—using MQTT, a lightweight messaging protocol that ensures real-time updates even on slow networks.

3. Web Dashboard

For law enforcement and monitoring teams, we built a web dashboard using ReactJS. This dashboard gives

a live overview of:

- Alcohol detection alerts
- Vehicle status
- Real-time GPS location
- Historical data logs

This helps authorities respond quickly and keep records for future analysis.

B. System Workflow

Here's how the whole system works step by step:

1. The driver breathes near the MQ3 sensor.
2. If the sensor detects alcohol above the legal limit:
    - The car won't start (if stationary), or
    - It will be slowed down gradually (if already in motion).
3. A buzzer alerts the driver and others nearby.
4. The system sends the alcohol level and GPS coordinates to the cloud.
5. The web dashboard updates in real-time, allowing authorities to respond immediately.

III. RESULTS

Fig. 3(a) shows the real-time alert dashboard interface of the SafeDrive system used by law enforcement. The dashboard displays active alerts for drunk driving, including key details such as the driver's name, contact number, vehicle registration number, alcohol level, and the exact timestamp when the violation was detected. Officers can also view the vehicle's live location, mark the alert as completed, or delete it once action is taken.

This web interface is built using ReactJS and integrated with Firebase as the cloud backend. The ESP32 microcontroller sends data via MQTT protocol in real time whenever alcohol levels exceed the legal limit. The motor driver is immediately activated to stop the vehicle safely, while the buzzer provides an audible warning. The alert is then automatically uploaded to the dashboard to notify authorities, enabling a quick and informed response.

IV. CONCLUSION

The SafeDrive system offers an intelligent and preventive solution to preventing drunk driving accidents by combining IoT technology with alcohol sensing and car control systems. Through the utilization of devices such as the MQ-3 sensor, ESP32 microcontroller, and NEO-6M GPS module, the system ensures constant monitoring of the driver's alcohol level and initiates preventive action before accidents can happen. The integration of the web application within SafeDrive provides the authorities with the ability to trace the car's location simply, monitor alcohol levels, and react quickly as required. Apart from increasing road safety, the technology also reunites police and technology for cooperative action. SafeDrive is essentially a smart, affordable, and reliable solution that operates to prevent accidents and encourage safe driving.

V. REFERENCES

[1] Pradeep Kumar G , Neelam Sanjeev Kumar, and Vijay Kumar P, “Safe transportation system using IoT based alcohol detection,” in 2023 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE), Bengaluru, India, 2023, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/10142338

[2] V. Sonule and R. Raman, “In-car breathalyzer systems for enhanced road safety through SVM classification and IoT connectivity,” in Proc. 2024 Int. Conf. Advances in Modern Age Technologies in Engineering (AMATHE), Pune, India, May 2024, pp. 506–511. [Online]. Available: https://ieeexplore.ieee.org/document/10142338

[3] N. G. Iyer, M. Arulmozhi, P. Sivakumar, S. Sudharsan, S. Jeny Sophia, and R. Kavitha, “AI-powered driver behavior prediction, drunk driving prevention, accident detection, and insurance integration,” in 2023 International Conference on Energy, Materials and Communication Engineering (ICEMCE), Madurai, India, Dec. 2023, pp. 688–693. [Online]. Available: https://colab.ws/articles/10.1109%2Ficemce57940.2023.10434027

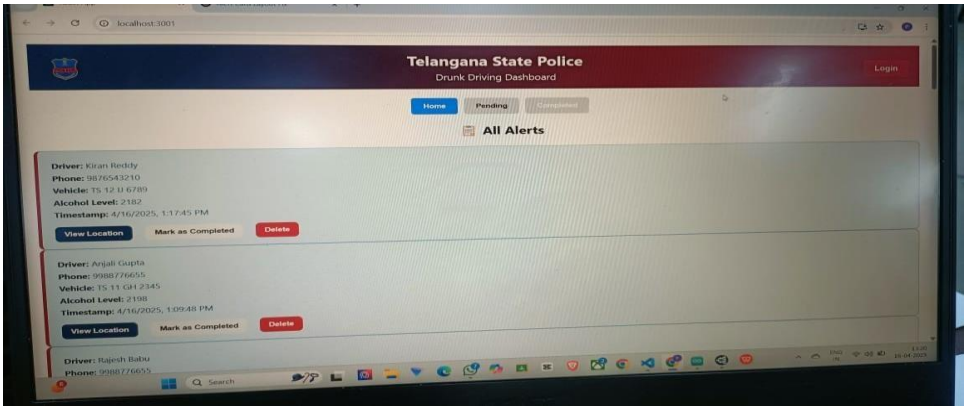
[4] V. Malarselvam, A. James Ashish, D. Peranban, S. Arun, and P. Arulmani, “Drunk driving prevention and automatic bus pass tracking by using image processing,” *Department of Electrical and Electronics Engineering, Sri Manakula Vinayagar Engineering College*, Puducherry, India, Jul. 2023. [Online]. Available: https://www.researchgate.net/publication/372139124

[5] S. A. Nordin, Z. M. Yusoff, and N. N. Mohammad, “Druken alcohol intelligent detection system IoT based Arduino controller,” *School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA*, Segamat, Malaysia, and *Microwave Research Institute, Universiti Teknologi MARA*, Shah Alam, Malaysia, Mar. 2023. [Online]. Available: https://www.researchgate.net/publication/368885194_Druken_alcohol_intelligent_detection_system_IoT_based_Arduino_controller

[6] A. Sinha, N. Kumar, A. Ticku, and M. Shah, “IoT based human biometric system using alcohol level detection for driver security,” *School of Computing and Information Sciences, IGNOU*, New Delhi, India; *Department of Computer Science, Lovely Professional University*, Punjab, India; *Bharti Vidyapeeth’s College of Engineering*, India; *Department of Computer Science & Engineering, Charotar University of Science & Technology*, Gujarat, India, Dec. 2023. [Online]. Available: https://www.researchgate.net/publication/379842814

9.2 Snapshots of the Result

1. Web Application:



2. Notification Alerts:

3. Hardware Setup