A

Project

Report On

“Vehicle Safety Using CAN and IOT”

Submitted in the partial fulfillment of the requirements for the PG Diploma in

EMBEDDED SYSTEMS & DESIGN (PG - DESD)

by

Kapare Krushna	250844230102
Abhishek Pujari	250844230002
Vijayalakshmi Madduru	250844230097

Sunbeam Institute of Information Technology, Pune & Karad.

Year : 2026

Sunbeam Institute of Information Technology,
Pune & Karad.

CERTIFICATE

This is to certify,

Kapare Krushna	250844230102
Abhishek Pujari	250844230002
Vijayalakshmi Madduru	250844230097

This is to certify that the candidate(s) have satisfactorily completed the project and presented a report on the topic titled “Vehicle Safety System using CAN and IoT” at Sunbeam Institute of Information Technology, in partial fulfilment of the requirements for the PG Diploma in Embedded Systems & Design (PG-DESD) for the academic year 2026.

Date : 30/01/2026

ACKNOWLEDGEMENT

This project titled "Vehicle Safety System using CAN and IoT" has been a valuable and enriching learning experience for us. We take this opportunity to submit our project work to Sunbeam Institute of Information Technology as part of the academic requirements of the PG Diploma in Embedded Systems & Design (PG-DESD).

We express our sincere gratitude to Ms. Utkarsha Mam for her constant guidance, technical support, and encouragement throughout the project. Her valuable suggestions and in-depth knowledge helped us understand complex concepts and overcome various technical challenges encountered during the project development.

We would also like to convey our heartfelt thanks to Mr. Devendra Dhande (Course Coordinator, PG-DESD) for his continuous support, motivation, and effective coordination. His efforts in providing the necessary hardware resources, internet facilities, and additional laboratory access played a crucial role in the successful completion of this project. We are thankful for his guidance and support throughout the course duration at Sunbeam Institute of Information Technology, Pune.

Finally, we extend our appreciation to all faculty members, lab staff, and peers who directly or indirectly supported us during the completion of this project.

INDEX

List of Figures

ABSTRACT

This project presents the design and implementation of a Vehicle Safety System using Controller Area Network (CAN) and Internet of Things (IoT) technologies. The primary objective of the system is to enhance vehicle safety by monitoring critical parameters and enabling reliable communication between electronic control units.

The system employs two STM32F407 microcontrollers interconnected through the CAN protocol using MCP2551 transceivers. One STM32F407 is interfaced with an ultrasonic sensor (HC-SR04) for obstacle detection, a temperature sensor (LM35) for monitoring system temperature, and a gas sensor (MQ-2) for detecting hazardous gas leakage. The sensed data is processed and transmitted over the CAN network to the second STM32F407, which is connected to a GPS module (M8N) for real-time vehicle location tracking.

To enable remote monitoring, an ESP8266 Wi-Fi module, programmed using Arduino IDE, transmits sensor and GPS data to the cloud using the MQTT protocol. The data is visualized on the Thing Speak IoT platform, allowing real-time monitoring of vehicle safety parameters from any remote location.

The use of the CAN protocol ensures fast, reliable, and fault-tolerant communication between controllers, which is crucial for safety-critical applications. This project demonstrates that an effective vehicle safety and monitoring system can be developed using standard hardware components and modern IoT technologies. The system can be further enhanced by integrating additional sensors and intelligent control algorithms to improve overall safety and performance.

1. INTRODUCTION

This project aims to enhance vehicle safety by developing a CAN-based vehicle safety and monitoring system that detects potential hazards and provides real-time alerts to the driver. The system improves safety by continuously monitoring critical parameters and enabling reliable communication between electronic control units.

An ultrasonic sensor (HC-SR04) is used to detect obstacles in front of the vehicle. The distance data is processed by an STM32F407 microcontroller, which determines collision risk and transmits safety information to another STM32F407 controller through the Controller Area Network (CAN) protocol using MCP2551 CAN transceivers. The CAN protocol ensures fast and fault-tolerant communication suitable for automotive applications.

The system also monitors temperature using an LM35 sensor and detects gas leakage using an MQ-2 sensor to identify unsafe conditions. Sensor data is transmitted over the CAN network, and a buzzer provides immediate alerts to the driver when abnormal conditions are detected.

To enable remote monitoring, IoT functionality is incorporated using an ESP8266 Wi-Fi module programmed in the Arduino IDE. Sensor and vehicle location data obtained from a GPS module (M8N) are sent to the cloud using the MQTT protocol and visualized on the Thingsboard IoT platform. The proposed system provides an effective and scalable solution for improving vehicle safety.

1.1. Project Scope

The scope of this project is to design and implement a CAN-based vehicle safety and monitoring system integrated with IoT technology to enhance road safety and enable real-time vehicle monitoring. The system focuses on detecting potential hazards such as obstacles, abnormal temperature, and gas leakage, and providing timely alerts to the driver to reduce the risk of accidents.

The project involves the use of two STM32F407 microcontrollers communicating through the Controller Area Network (CAN) protocol using MCP2551 transceivers, ensuring reliable and fault-tolerant data exchange between electronic control units. Safety parameters including obstacle distance, temperature, and gas concentration are monitored using HC-SR04, LM35, and MQ-2 sensors respectively.

In addition to local alerts using a buzzer, the system extends its functionality by integrating IoT features**.** Sensor data and vehicle location information obtained from a GPS module (M8N) are transmitted to the cloud using an ESP8266 Wi-Fi module and the MQTT protocol. The data is visualized on the Thingsboard IoT platform, enabling remote monitoring of vehicle safety conditions.

The project scope also allows for future expansion, such as the addition of more sensors, intelligent decision-making algorithms, and advanced vehicle safety features, making the system suitable for smart vehicles, fleet monitoring, and intelligent transportation systems.

1.2. CAN Protocol

The Controller Area Network (CAN) bus was developed by BOSCH as a multi-master, message-broadcast system capable of operating at a maximum speed of 1 megabit per second (Mbps). Unlike traditional networks such as USB or Ethernet, where data is sent in large blocks from one node to another under the control of a central master, CAN transmits short, prioritized messages—including sensor readings, temperature, engine RPM, and other critical signals—to all nodes simultaneously, ensuring real-time data consistency and synchronization across the network.

Originally designed for the automotive industry, CAN has become an ISO-standardized serial communication bus that replaces complex and bulky wiring harnesses with a simple, reliable two-wire system. Its architecture provides high immunity to electrical noise, robust error detection and correction, and the ability to self-diagnose and isolate faulty nodes, making it highly suitable for safety-critical and real-time applications in vehicles, industrial automation, robotics, and other embedded systems.

By enabling multiple electronic control units (ECUs) to communicate efficiently on a single bus, CAN reduces wiring complexity, enhances system reliability, and supports flexible, scalable, and modular network designs. Its widespread adoption across automotive and industrial applications highlights its effectiveness as a robust, real-time communication standard for modern electronics.

As a result, CAN has achieved widespread adoption across automotive, industrial, and embedded systems, from engine and transmission control to robotics, factory automation, and medical equipment. Its ability to provide fast, reliable, and deterministic communication in complex, safety-critical environments underscores its status as a proven and enduring standard for real-time electronic communication in modern electronics.

CAN Data Frames:

CAN frames are structured packets used to send messages over the network. The main difference between standard and extended frames is the length of the identifier and how the frame accommodates it.

Standard Frame : (11-bit Identifier)

Standard frame is compact, with a short identifier (11 bits), making it suitable for small networks with fewer nodes.

i. Identifier: 11 bits, defines the priority of the message. Lower numbers have higher priority.
ii. Control Field (DLC): Indicates how many bytes of data are being sent (0–8 bytes).
iii. Data Field: Contains the actual payload (sensor readings, commands, etc.), up to 8 bytes.
iv. CRC Field: 15-bit cyclic redundancy check for error detection. It allows receiving nodes to verify that the transmitted data has not been corrupted during transmission, enhancing the reliability of the CAN network.
v. ACK Field: Allows receivers to acknowledge correct reception.
vi. SOF and EOF: Start and end of the frame.

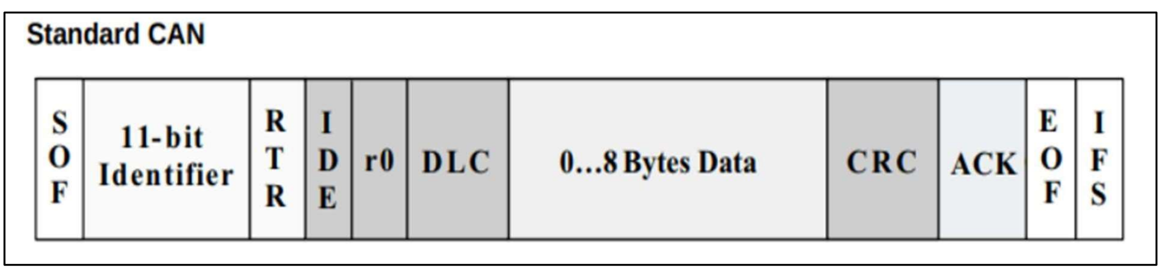

Fig 1.2.1. CAN Standard Frame Format

Extended Frame : (29-bit Identifier)

Extended frames are used when more identifiers are needed, such as in large vehicles with many ECUs or industrial systems. The extra identifier bits allow a larger number of unique messages without losing priority handling.

i. Start of Frame (SOF) – 1 bit: Marks the beginning of a CAN frame and synchronizes all nodes on the bus.
ii. Base Identifier – 11 bits: Forms the first part of the 29-bit identifier; determines the message priority (lower value = higher priority).
iii. Substitute Remote Request (SRR) – 1 bit: Ensures compatibility with standard CAN nodes; always recessive (1) in extended data frames.
iv. Identifier Extension (IDE) – 1 bit: Indicates that the frame uses a 29-bit extended identifier (recessive = extended, dominant = standard).
v. Extended Identifier – 18 bits: Combined with the base identifier to make a 29-bit total ID, uniquely identifying messages in large networks.
vi. Remote Transmission Request (RTR) – 1 bit: Shows whether the frame is data (dominant) or a request for data (recessive).
vii. Reserved Bit 0 (r0) – 1 bit: Reserved for future use; usually set to dominant (0).
viii. Reserved Bit 1 (r1) – 1 bit: Reserved for future use; usually set to dominant (0).
ix. Data Length Code (DLC) – 4 bits: Specifies how many bytes (0–8) are contained in the Data Field.
x. Data Field – 0–64 bits (0–8 bytes): Carries the actual message payload, such as sensor readings or control commands.
xi. Cyclic Redundancy Check (CRC) – 15 bits: Ensures error detection for the transmitted frame.
xii. CRC Delimiter – 1 bit: Marks the end of the CRC field (always recessive).

xiii. Acknowledgment (ACK) Slot – 1 bit: Allows receiving nodes to confirm correct reception by overwriting with dominant (0).

xiv. ACK Delimiter – 1 bit: Marks the end of the ACK field (recessive).

xv. End of Frame (EOF) – 7 bits: Marks the end of the frame, allowing the bus to return to idle state.

xvi. Intermission / Interframe Space – 3 bits: Provides a short idle period between frames for bus synchronization.

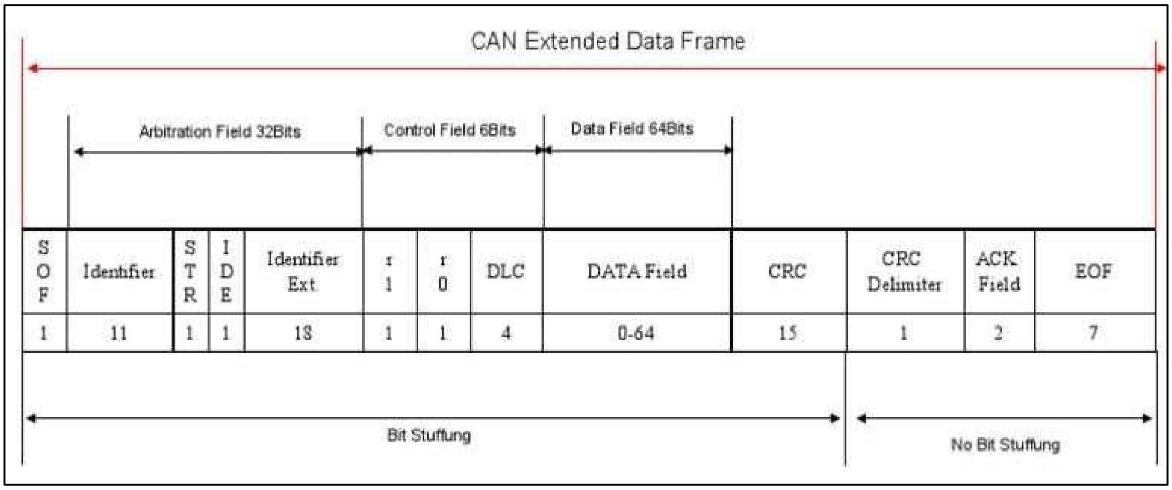

Fig 1.2.2. CAN Extended Frame Format

Integrating the CAN protocol into a project enables deterministic, multi-node communication over a single two-wire bus, reducing wiring complexity and improving system reliability. Its message-oriented architecture with priority-based arbitration ensures that critical signals are transmitted with minimal latency, while built-in CRC, ACK, and fault confinement mechanisms provide robust error detection and system integrity. By supporting real-time data exchange between multiple ECUs or microcontrollers, CAN facilitates scalable, modular, and high-performance embedded system designs, making it ideal for automotive, industrial automation, and robotics applications.

2. System Requirements

The proposed Vehicle Safety System integrates hardware and software components to ensure real-time monitoring and safety alerts. The hardware consists of two STM32F407 microcontrollers interconnected via the CAN protocol using MCP2551 transceivers, along with sensors for obstacle detection (HC-SR04), temperature monitoring (LM35), and gas leakage detection (MQ-2). A GPS module (M8N) provides location tracking, while an ESP8266 Wi-Fi module enables IoT-based remote monitoring through the Thingsboard platform. A buzzer provides immediate alerts in case of hazardous conditions.

The software requirements include STM32CubeIDE for microcontroller programming, Arduino IDE for ESP8266 development, and the MQTT protocol for cloud communication. Together, these hardware and software elements provide a reliable, fault-tolerant, and scalable system for enhancing vehicle safety, with provisions for real-time local and remote monitoring, timely alerts, and future expansion.

1. Hardware Requirements :

i. STM32F407 Microcontrollers (x2)
   a) ARM Cortex-M4, 168 MHz, with built-in CAN controller
   b) ADC channels for sensor interfacing
   c) UART for GPS and ESP8266 communication

ii. CAN Transceivers (MCP2551 x2)
   a) Converts logic-level CAN signals to differential signals for the CAN bus
   b) Ensures robust communication between the two STM32 nodes

iii. Sensors

a) Ultrasonic Sensor (HC-SR04): Obstacle detection, range 2–400 cm

b) Temperature Sensor (LM35): Monitors engine/cabin temperature

c) Gas Sensor (MQ-2): Detects LPG, smoke, and other hazardous gases

iv. GPS Module (M8N): Provides real-time vehicle location.

v. ESP8266 Wi-Fi Module: For IoT-based data transmission to the cloud via MQTT protocol

vi. Buzzer: Provides audible alerts during hazardous conditions

vii. Power Supply: Stable 5V/3.3V supply for microcontrollers, sensors, and modules

viii. USB to TTL UART serial converter - CP2120

ix. Cabling and Termination: Twisted pair wires for CAN bus, 120 Ω resistors at each end

2. Software Requirements :

i. STM32CubeIDE:

a) Embedded C programming for STM32 microcontrollers

b) CAN communication configuration and sensor data processing

ii. Arduino IDE: Programming ESP8266 for IoT and MQTT-based data transmission

iii. MQTT Protocol: Lightweight messaging protocol for IoT communication

iv. Thingsboard IoT Platform: Remote monitoring and visualization of sensor and GPS data

2.1. Hardware Requirements

2.1.1. STM32F407 Discovery Board

The STM32F407 Discovery Board is the central processing platform of this project and is designed to provide a powerful, flexible, and cost-effective development environment for embedded systems. It is based on the STM32F407VGT6 microcontroller, which is part of the STM32F4 series from STMicroelectronics. The STM32F407 is specifically suited for high-performance, real-time applications such as vehicle safety systems due to its combination of speed, memory, and integrated peripherals.

i. Microcontroller Core:

a) Processor: 32-bit ARM Cortex-M4 with floating-point unit (FPU) and Digital Signal Processing (DSP) instructions.

b) Clock Speed: Up to 168 MHz, providing high-speed computation for real-time sensor data processing.

Handles real-time processing of sensor data (HC-SR04, LM35, MQ-2), executes CAN communication, and manages IoT data transmission.

ii. Memory

a) Flash Memory: 1 MB, for storing program code and firmware.

b) SRAM: 192 KB, for temporary data storage, including sensor readings, CAN messages, and GPS coordinates.

Sufficient memory allows simultaneous execution of multiple tasks, such as reading sensors, processing data, and sending messages over CAN and Wi-Fi.

iii. Communication Interface:

a) CAN Controller: Integrated controller for Controller Area Network communication, enabling multi-master, real-time communication between microcontrollers.
b) UART: Serial communication interface used for GPS module (M8N) and ESP8266 Wi-Fi module.
c) I2C / SPI: Interfaces for connecting additional sensors or modules if required.
d) USB OTG FS (Micro-AB connector): Supports USB communication for programming or connecting peripherals.

CAN ensures reliable inter-controller communication, UART enables GPS and IoT data transmission, and USB provides easy programming and debugging.

iv. Analog and Digital Features:

a) ADC: 12-bit Analog-to-Digital Converters (up to 16 channels), for reading analog sensors like LM35 and MQ-2.
b) DAC: 12-bit Digital-to-Analog Converter for audio or analog signal generation.
c) Timers: Multiple timers for PWM generation, time measurement, or scheduling tasks.

ADC reads sensor signals accurately; timers are used for precise timing, e.g., ultrasonic sensor measurement and CAN message.

v. Debugging & Programming:

a) On-board ST-LINK/V2 debugger/programmer
b) Can function as standalone ST-LINK using SWD interface for programming other STM32 boards.

Enables easy firmware development, debugging, and testing without additional external hardware.

vi. Input/Output Features:

a) User LEDs: Eight onboard LEDs including LD1 (red/green) for USB, LD2 for power, and four user-controllable LEDs.
b) Push Buttons: Two push buttons, one for reset and one for user input.
c) GPIOs: Multiple General Purpose Input/Output pins for connecting sensors, buzzer, or other peripherals.

LEDs can indicate system status or warnings; buttons can be used for testing or user input; GPIOs connect all sensors and outputs.

vii. MEMS Sensors:

a) Accelerometer: LIS302DL or LIS3DSH 3-axis MEMS accelerometer.
b) Microphone: MP45DT02 omnidirectional digital MEMS microphone.

Though not used in your current project, these sensors are available for motion detection or audio-based alerts in future enhancements.

viii. Power Supply

a) Input: Powered via USB (5V) or external 5V source.

b) On-board regulators: Provide 3.3V and 5V outputs for the microcontroller and peripheral devices.

Ensures stable and reliable operation of STM32, sensors, CAN transceivers, and ESP8266 Wi-Fi module. Development & IDE Support

ix. Compatible with multiple IDEs:

a) STM32CubeIDE (official ST IDE)

b) Keil MDK-ARM

c) IAR Embedded Workbench

Offers flexibility for firmware development, debugging, and CAN/IoT programming.

Fig 2.1.1. STM32Cube IDE

2.1.2. CAN Transceivers -MCP2551

The MCP2551 is a high-speed, fault-tolerant CAN transceiver used in this project to interface the STM32F407 CAN controller with the physical CAN bus. It converts the digital CAN transmit and receive signals from the microcontroller into differential signals suitable for transmission over the CAN bus and vice versa. The device is fully compliant with the ISO-11898 CAN physical layer standard and supports reliable communication in automotive and industrial environments.

Key Features and Specifications :

i. High-Speed Operation: Supports CAN communication speeds of up to 1 Mb/s, suitable for real-time vehicle safety applications.

ii. ISO-11898 Compliance: Fully compatible with standard CAN physical layer requirements, ensuring interoperability and reliability.

iii. Differential Signaling: Provides differential transmit and receive capability, offering high noise immunity and robust data transmission.

iv. Voltage Compatibility: Designed for 12 V and 24 V systems, making it suitable for automotive environments.

v. Fault Tolerance: Detects ground faults and protects against permanent dominant conditions on the TXD input.

vi. Protection Mechanisms: Includes power-on reset, brown-out protection, and safeguards against short-circuit conditions and high-voltage transients on the CAN bus.

vii. Bus Stability: An unpowered node or brown-out condition does not disturb ongoing CAN bus communication.

viii. Low Power Consumption: Supports low-current standby mode for power-efficient operation.

ix. Scalability: Allows connection of up to 112 nodes on a single CAN network.

x. Wide Temperature Range: Operates reliably in industrial and automotive temperature ranges (–40°C to +85°C / +125°C).

In this project, the MCP2551 acts as the physical layer interface between the two STM32F407 microcontrollers connected over the CAN bus. It ensures reliable and noise-immune communication of sensor data such as obstacle distance, temperature, and gas status between different electronic control units. By protecting the microcontrollers from voltage spikes and electrical disturbances, the MCP2551 enhances system reliability, making it suitable for safety-critical automotive applications.

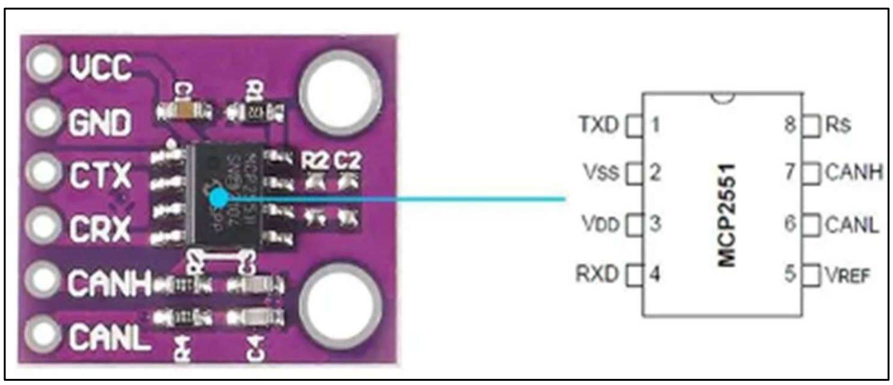

Fig 2.1.2. CAN Transceiver MCP2551

2.1.3. Sensors

2.1.3.1. Ultrasonic Sensor HC- SR04 :

The HC-SR04 ultrasonic ranging module is used in this project for obstacle detection in front of the vehicle. It provides accurate, non-contact distance measurement by transmitting and receiving ultrasonic waves, making it suitable for collision avoidance and safety applications.

Key Features and Specifications :

i. The sensor operates by transmitting ultrasonic sound waves at 40 kHz.
ii. A trigger pulse of at least 10 microseconds is applied to the Trigger pin.
iii. The module automatically transmits eight ultrasonic bursts and waits for the reflected echo.
iv. When the reflected signal is received, the Echo pin goes high.
v. The duration of the high-level pulse on the Echo pin represents the time taken by the ultrasonic wave to travel to the obstacle and return.
vi. Operating Voltage: 5 V
vii. Measurement Range: 2 cm to 400 cm Interface: Digital Trigger and Echo pins

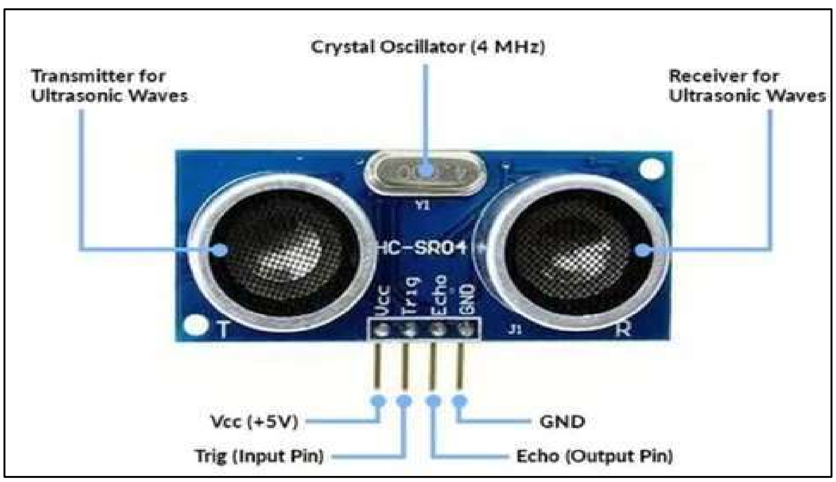

Fig 2.3.1. Ultrasonic Sensor HC-SR04

2.1.3.2. Temperature Sensor LM35

The LM35 is a precision integrated-circuit temperature sensor used in this project to monitor the vehicle/system temperature and detect abnormal thermal conditions. It provides an analog output voltage that is linearly proportional to the temperature in degrees Celsius, eliminating the need for external calibration.

Key Features and Specifications

i. Sensor Type: Analog temperature sensor with linear voltage output
ii. Output Scale Factor: 10 mV/°C
iii. Temperature Range: –55°C to +150°C
iv. Accuracy: ±0.5°C (typical at 25°C)
v. Operating Voltage: 4 V to 30 V
vi. Current Consumption: Very low, typically 60 µA
vii. Output Impedance: Approximately 0.1 Ω
viii. Self-Heating: Less than 0.1°C in still air
ix. Pin Configuration: 3 pins – VCC, Output, Ground

In this Vehicle Safety, the LM35 sensor monitors vehicle temperature through the STM32F407 ADC. The data is transmitted over the CAN network and uploaded to the IoT cloud for remote monitoring. Alerts are generated when temperature exceeds safe limits.

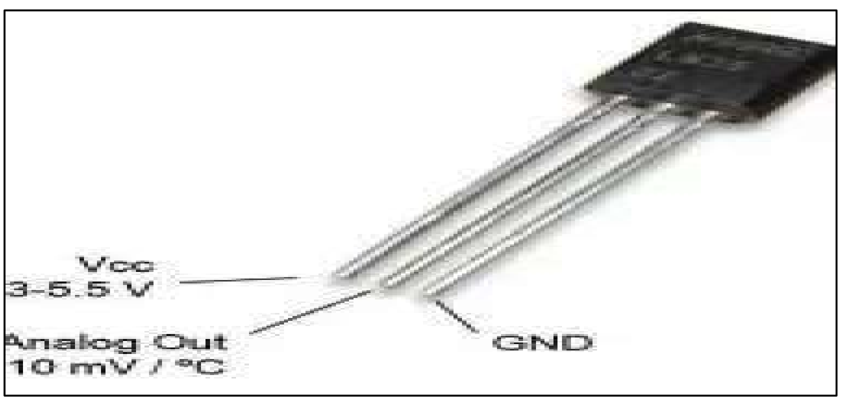

Fig 2.1.3.2. Temperature Sensor

2.1.3.3. Gas Sensor MQ-2

The MQ-2 gas sensor is used in this project to detect the presence of hazardous and combustible gases such as LPG, methane, hydrogen, and smoke. It plays an important role in enhancing vehicle safety by identifying gas leakage or smoke conditions inside or around the vehicle.

Key Features and Specifications:

i. Sensor Type: Semiconductor gas sensor
ii. Detectable Gases: LPG, propane, methane, hydrogen, smoke
iii. Operating Voltage: 5 V
iv. Heater Voltage: 5 V
v. Output Type: Analog and digital output
vi. Sensitivity: Adjustable using onboard potentiometer
vii. Response Time: Fast response to gas concentration changes
viii. Operating Temperature: –10°C to +50°C
ix. Long Life: Stable performance with long operational lifespan

In the proposed Vehicle Safety, the MQ-2 gas sensor is connected to the STM32F407 ADC to monitor hazardous gases or smoke. When gas levels exceed the set threshold, the system detects a dangerous condition and activates a buzzer to alert the driver.

Fig 2.1.3.3. Gas Sensor MQ2

2.1.4 <u>GPS Module M8N</u>

The M8N GPS module is used in this project to provide real-time vehicle location tracking. It receives signals from GPS satellites and calculates the vehicle’s latitude and longitude, which are used for monitoring and tracking purposes.

<u>Key Features and Specifications:</u>

i. Positioning System: GPS / GNSS
ii. Operating Voltage: 3.3 V – 5 V
iii. Accuracy: Approximately 2.5 meters
iv. Interface: UART (Serial communication)
v. Data Output: NMEA sentences (GPRMC, GPGGA, etc.)
vi. Update Rate: Up to 10 Hz

In the Vehicle Safety, the GPS module provides real-time location data to the STM32F407 microcontroller. The location information is transmitted along with sensor data to the IoT cloud platform via the ESP8266 module, enabling remote vehicle tracking and monitoring.

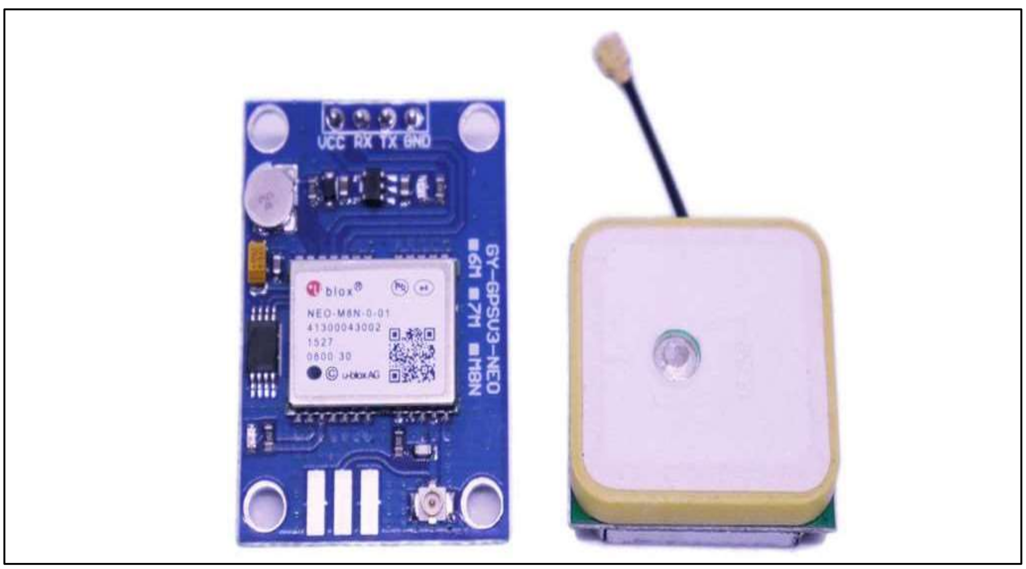

Fig 2.1.4. GPS M8N

2.1.5 Wi-Fi Module ESP2866

The ESP8266 Wi-Fi module is used to enable internet connectivity and IoT functionality in the system. It allows sensor and GPS data to be transmitted to the cloud for remote monitoring.

Key Features and Specifications:

i. Operating Voltage: 3.3 V
ii. Wi-Fi Standard: IEEE 802.11 b/g/n
iii. Communication Interface: UART
iv. Protocols Supported: TCP/IP, MQTT
v. Programming Platform: Arduino IDE
vi. Low Power Consumption: Suitable for embedded systems

In this project, the ESP8266 sends sensor data and GPS coordinates to the Thingsboard IoT platform using the MQTT protocol. This enables real-time remote monitoring and data visualization of vehicle safety parameters.

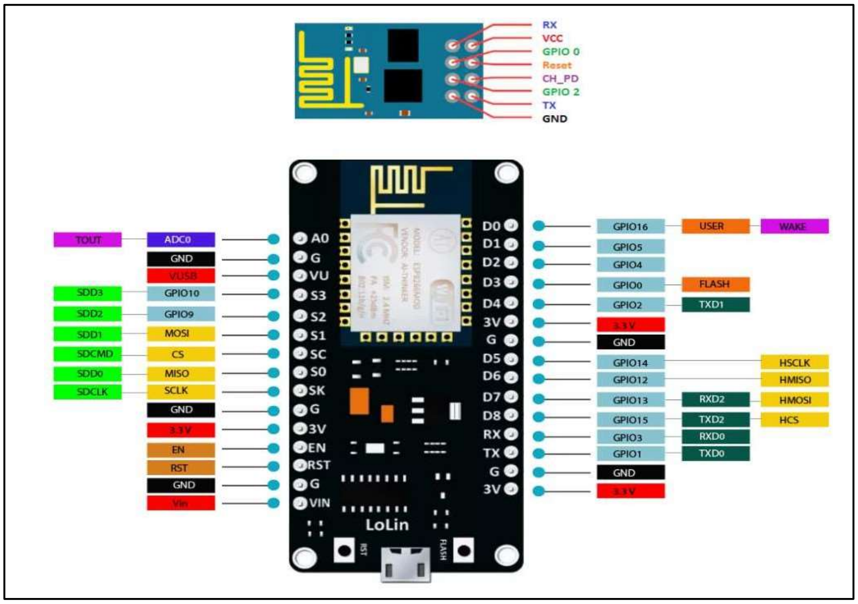

Fig 2.1.5. GPS Module M8N

2.1.6 Buzzer

The buzzer is used as an audible alert device to warn the driver about hazardous conditions detected by the system.

Key Features and Specifications:

i. Operating Voltage: 3.3 V / 5 V
ii. Type: Active buzzer
iii. Control: GPIO controlled

In this project, the buzzer is activated when critical conditions such as obstacle detection, gas leakage, or excessive temperature are identified. The audible warning helps ensure that the driver is instantly alerted, even without visual monitoring of the system. By providing real-time alerts, the buzzer enhances system responsiveness and contributes to improved vehicle safety.

Fig 2.1.6. Buzzer

2.1.7 Power Supply

The power supply unit provides the required and stable operating voltages to all components of the Vehicle Safety System. A regulated 5 V supply is used as the primary input, which is distributed to the STM32F407 microcontrollers, sensors, CAN transceivers, and communication modules.

2.1.8 USB to TTL UART Serial Converter (CP2102)

The CP2102 USB to TTL UART serial converter is used in this project as a communication and debugging interface between the computer and embedded devices. It converts USB signals from a PC into TTL-level UART signals (TX and RX), enabling serial communication with microcontrollers and modules.

Key Features and Specifications:

i. Interface: USB to UART (TTL level)
ii. Operating Voltage: 5 V (USB powered)
iii. Logic Level: 3.3 V / 5 V TTL compatible
iv. Communication Signals: TX, RX, GND ,
v. Baud Rate Support: Up to 1 Mbps
vi. USB Standard: USB 2.0 compliant

The CP2102 converter provides a UART interface to program, configure, and monitor modules like the ESP8266, enabling real-time data observation for testing and debugging.

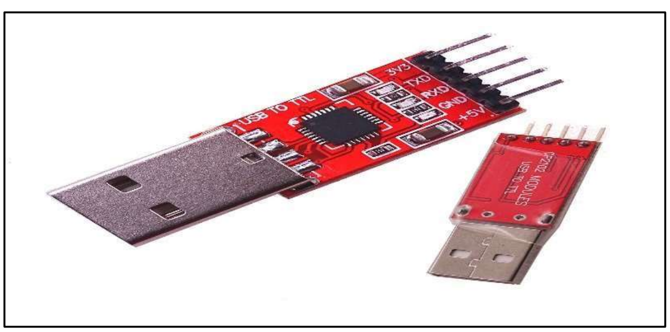

Fig 2.1.8. CP2102 USB to TTL UART serial converter

2.2. Software Requirements

2.2.1. STM32CubeIDE :

i. Used for programming the STM32F407 microcontrollers.

ii. Supports embedded C development, peripheral configuration, and debugging.

iii. Enables CAN protocol configuration and testing.

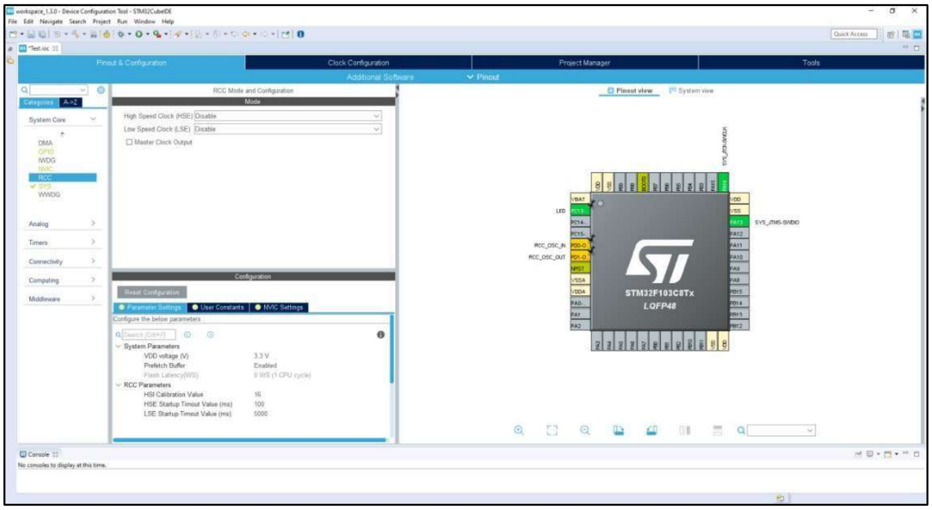

Fig 2.2.1. STM32 CubeIDE

2.2.2. Arduino IDE:

i. Used for programming the ESP8266 Wi-Fi module.

ii. Supports MQTT communication protocol implementation for cloud data transmission.

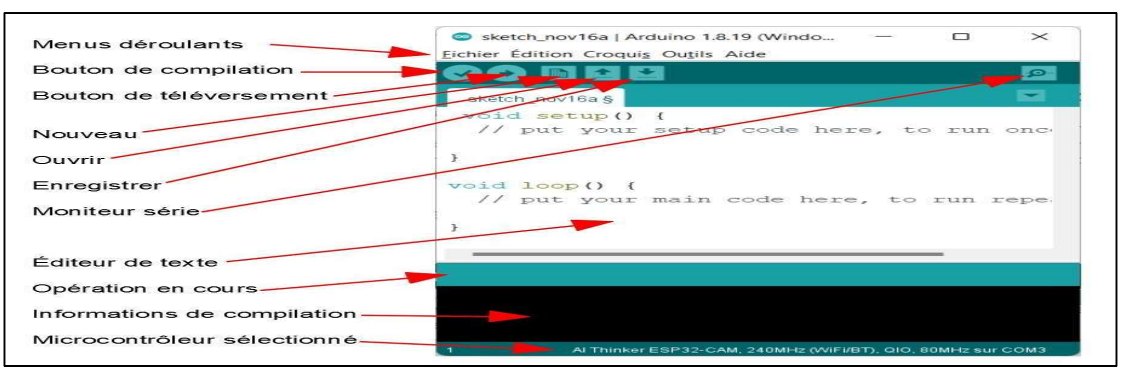

Fig 2.2.2. Arduino IDE

2.2.3. Thingsboard IoT Platform:

i. Cloud platform for real-time monitoring and visualization of sensor and GPS data.

ii. Provides dashboards for distance, temperature, gas status, and vehicle location tracking.

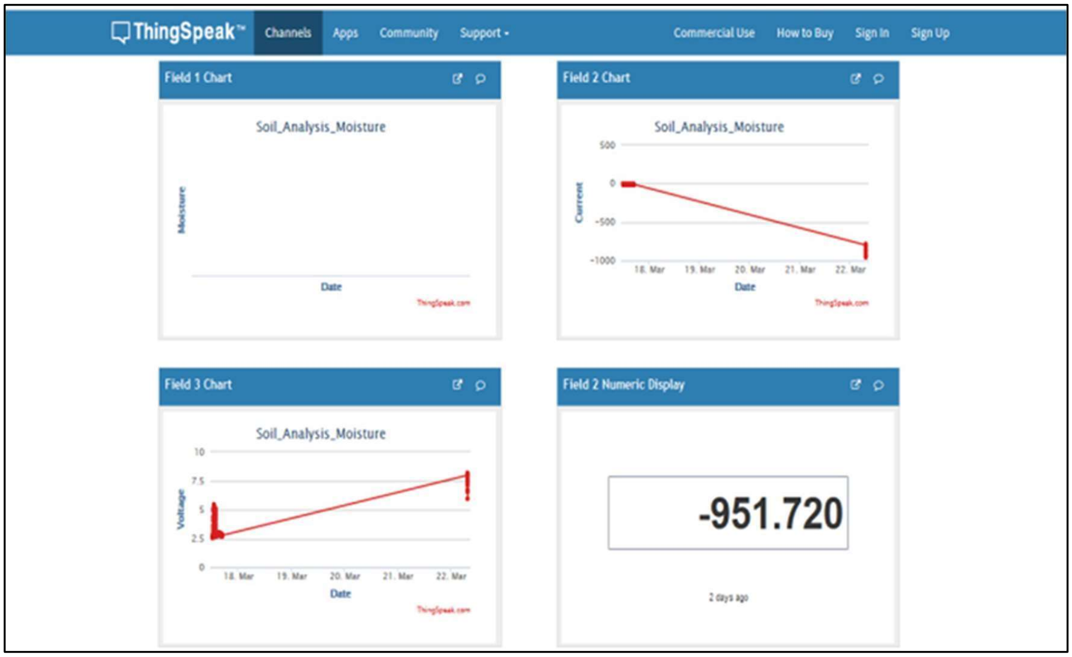

Fig 2.2.3. Thingsboard IoT Platform

3. Design and Implementation

3.1. Block Diagram

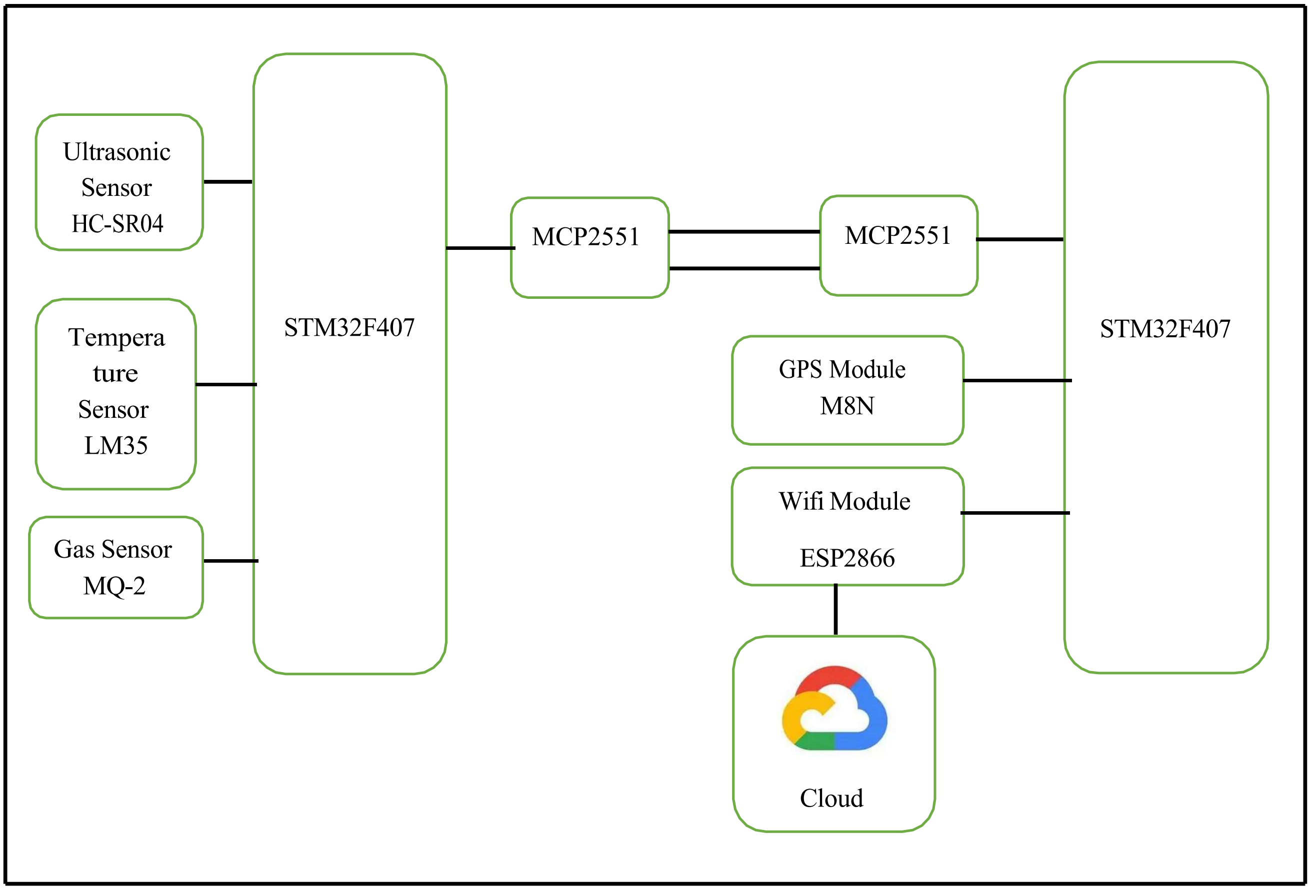

Fig 3.1. Block Diagram

The system consists of two STM32F407 microcontrollers communicating through the CAN protocol using MCP2551 transceivers. One controller collects data from ultrasonic, temperature, and gas sensors, while the other receives this data along with GPS information. The processed data is transmitted to the cloud via an ESP8266 Wi-Fi module using MQTT and monitored remotely on the Thingsboard IoT platform.

Block Diagram Description

STM32F407 [Node 1 – Sensor Node / CAN Transmitter]:

This microcontroller acts as the sensor and data acquisition unit. It is responsible for interfacing with the Ultrasonic Sensor (HC-SR04), Temperature Sensor (LM35), and Gas Sensor (MQ-2). The STM32F407 processes the sensor data and transmits the safety-related information onto the CAN bus. This node primarily functions as a CAN data transmitter.

STM32F407 [Node 2 – IoT & GPS Node / CAN Receiver]:

This microcontroller acts as the receiving and data processing unit. It receives sensor data from the CAN bus and interfaces with the GPS module (M8N) to obtain real-time vehicle location. It also communicates with the ESP8266 Wi-Fi module to forward the received data to the cloud. This node primarily functions as a CAN data receiver and IoT gateway.

MCP2551 CAN Transceivers:

The MCP2551 devices act as the physical layer interface for CAN communication. They convert the digital CAN signals from the STM32F407 microcontrollers into differential signals required on the CAN bus and vice versa. The CAN High (CANH) and CAN Low (CANL) lines enable reliable and noise-immune communication between the two nodes.

Ultrasonic Sensor (HC-SR04):

This sensor is used to measure the distance between the vehicle and obstacles in front. It works by transmitting ultrasonic pulses and measuring the echo return time. The distance information is used to detect potential collision risks.

Temperature Sensor (LM35):

The LM35 sensor is used to monitor the system or environmental temperature. It provides a linear analog output proportional to temperature, allowing detection of overheating conditions.

Gas Sensor (MQ-2):

The MQ-2 sensor detects the presence of hazardous gases such as LPG, smoke, or methane. It helps identify unsafe environmental conditions inside or around the vehicle.

GPS Module (M8N):

The GPS module provides real-time vehicle location in the form of latitude and longitude coordinates. This data is used for vehicle tracking and remote monitoring.

Wi-Fi Module (ESP8266):

The ESP8266 module provides internet connectivity to the system. It is programmed using Arduino IDE and transmits sensor and GPS data to the cloud using the MQTT protocol.

Cloud Platform (Thingsboard):

The Thingsboard IoT platform is used to store, visualize, and monitor sensor and GPS data remotely. It displays real-time graphs and values for distance, temperature, gas levels, and location.

3.2. System Functionality

i. The STM32F407 Sensor Node (Node 1) continuously reads data from the Ultrasonic Sensor (HC-SR04), Temperature Sensor (LM35), and Gas Sensor (MQ-2).

ii. The sensor data is processed to detect unsafe conditions such as obstacle proximity, abnormal temperature, or gas leakage.

iii. The processed sensor information is transmitted over the CAN bus using the MCP2551 CAN transceiver.

iv. The STM32F407 IoT & GPS Node (Node 2) receives the sensor data via the CAN bus.

v. Simultaneously, the GPS module (M8N) provides real-time vehicle location data to the second STM32F407.

vi. The STM32F407 sends both sensor data and GPS information to the ESP8266 Wi-Fi module.

vii. The ESP8266 publishes the data to the cloud using the MQTT protocol.

viii. The Thingsboard IoT platform displays the received data, enabling real-time remote monitoring of vehicle safety parameters.

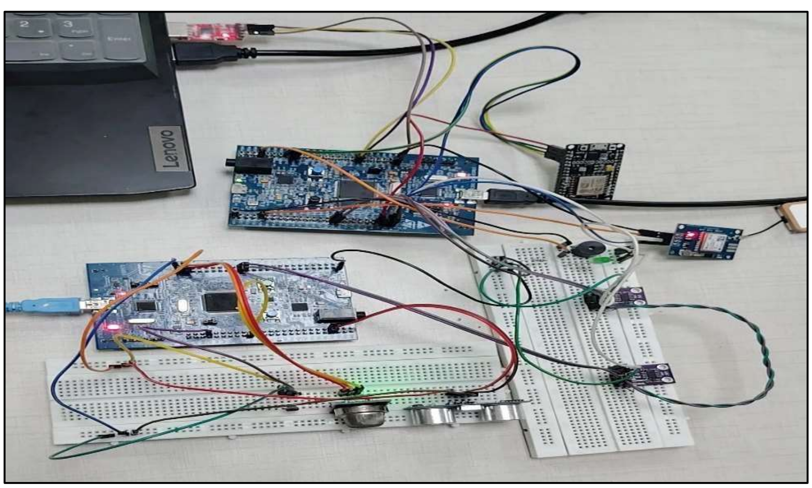

Fig 3.2. System Architecture

4. Advantages and Disadvantages

4.1. Advantages

i. The system provides continuous and real-time monitoring of critical vehicle safety parameters such as obstacle distance, temperature, and gas leakage.

ii. CAN protocol ensures fast, reliable, and fault-tolerant communication between multiple STM32F407 controllers, which is essential for automotive applications.

iii. IoT integration enables remote monitoring and data visualization through the Thingsboard cloud platform from any location.

iv. Reduced wiring complexity due to CAN bus architecture improves system reliability and maintainability.

v. The modular design allows easy expansion and integration of additional sensors and safety features.

4.2. Disadvantages

i. The accuracy of sensor data may be affected by environmental conditions such as dust, humidity, or extreme temperatures.

ii. Dependence on internet connectivity limits real-time cloud monitoring in low-network areas.

iii. System complexity increases due to the integration of CAN communication and IoT modules.

iv. Ultrasonic sensors have limited range and may not detect certain types of obstacles effectively.

5. Future Scope

i. The proposed Vehicle Safety System using CAN and IoT can be further enhanced by incorporating advanced technologies and additional safety features to meet the requirements of modern intelligent vehicles. More sensors such as cameras, radar, LiDAR, rain sensors, and vibration sensors can be integrated to improve obstacle detection accuracy and environmental awareness. Sensor fusion techniques can be applied to combine data from multiple sensors for more reliable decision-making.

ii. Advanced driver assistance features such as automatic emergency braking, adaptive cruise control, lane departure warning, and collision avoidance systems can be implemented using more powerful control algorithms. The system can also be extended to include vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication to enable cooperative safety systems and smart traffic management.

iii. IoT functionality can be further enhanced by developing a dedicated mobile or web application to provide real-time alerts, vehicle tracking, and historical data analysis. Cloud-based analytics and machine learning algorithms can be used to predict potential failures, analyze driving patterns, and improve overall safety performance. Additionally, cybersecurity measures can be implemented to protect vehicle data and communication channels from unauthorized access.

iv. The system can also be adapted for fleet management, emergency response vehicles, and commercial transportation systems. With further development, the proposed architecture can serve as a scalable platform for future smart vehicles and intelligent transportation systems.

6. Conclusion

The Vehicle Safety System using CAN and IoT successfully demonstrates an effective approach to enhancing vehicle safety through real-time monitoring and reliable communication. The system integrates multiple sensors with STM32F407 microcontrollers and uses the CAN protocol to ensure fast, robust, and fault-tolerant data exchange between electronic control units. The inclusion of IoT technology enables remote monitoring of vehicle safety parameters through cloud connectivity, providing improved situational awareness and accessibility.

The project highlights the practical implementation of embedded systems, CAN-based communication, and IoT integration using standard hardware components. The modular and scalable design allows for future enhancements such as the addition of advanced sensors and intelligent control algorithms. Overall, the proposed system provides a strong foundation for developing advanced vehicle safety solutions and intelligent transportation systems.

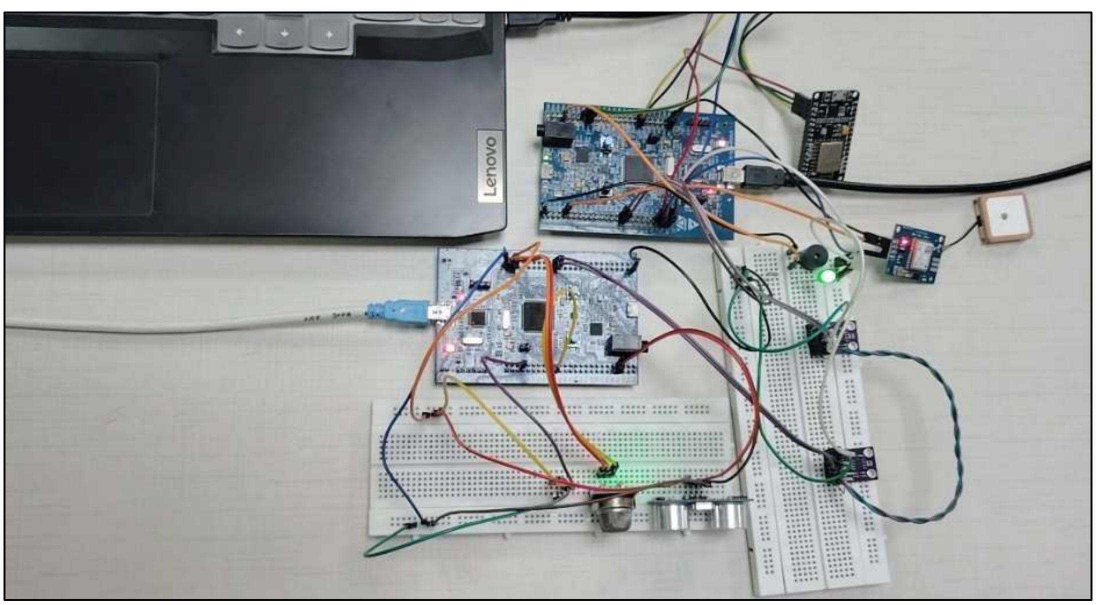

Fig 6. Hardware Output

6.1. <u>OutPut</u>

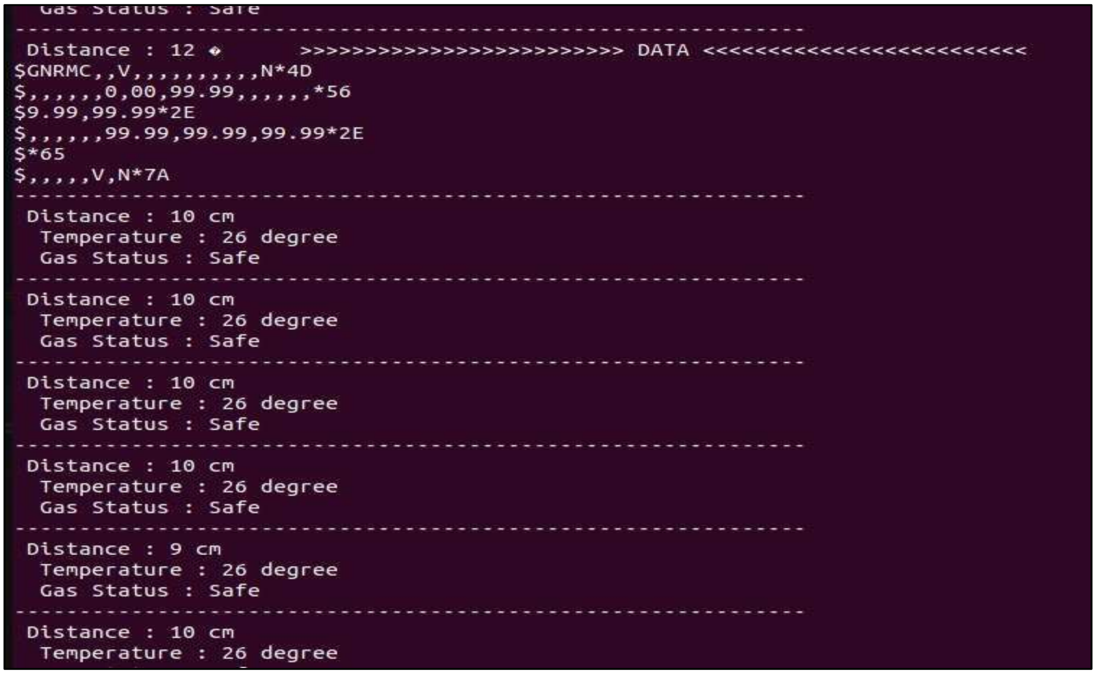

Fig 6.1.1. UART Output

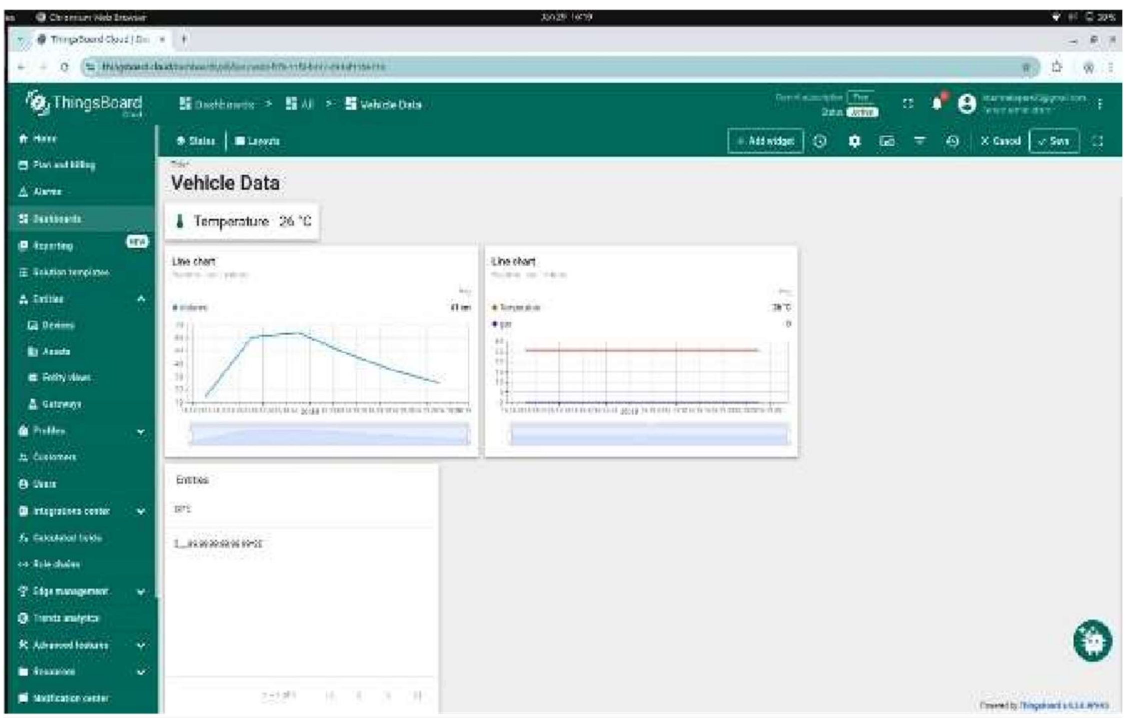

Fig 6.1.2. IOT Cloud Monitoring Output

7. References

[1] R. Bosch GmbH, "Controller Area Network (CAN): An Overview," Automotive Electronics Journal, Germany.

[2] S. Sharma and R. Kumar, "Design and Implementation of Vehicle Safety System Using CAN Protocol," International Journal of Engineering Research and Technology (IJERT), vol. 8, no. 6, pp. 512–517.

[3] A. Patil, S. Deshmukh, and P. Kulkarni, "Collision Avoidance System for Automobiles Using CAN Communication," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol. 7, no. 4.

[4] M. Singh and V. Verma, "IoT Based Smart Vehicle Monitoring and Accident Prevention System," International Journal of Computer, vol. 178, no. 9.

[5] P. Raut and S. Bhosale, "Real-Time Vehicle Safety Monitoring Using Embedded Systems and CAN Bus," International Conference on Smart Systems and Inventive Technology (ICSSIT).

[6] N. Gupta and A. Jain, "Automatic Braking and Obstacle Detection System Using Ultrasonic Sensor," International Journal of Emerging Technologies in Engineering Research (IJETER).

[7] J. Lee and K. Park, "Design of CAN-Based Embedded System for Automotive Safety Applications," IEEE International Conference on Vehicular Electronics and Safety.

[8]A. Mishra et al., "IoT-Based Vehicle Tracking and Safety System," International Journal of Engineering and Advanced Technology (IJEAT).

[9] S. Kumbhar and R. Patil, "Vehicle Safety and Monitoring System Using IoT and Embedded Controller," International Journal of Scientific Research in Engineering and Technology (IJSRET)*.

[10] Sunbeam Institute of Information Technology, "PG-DESD Project Guidelines and Reference Manual," Pune.