

“PyroSense: An Intelligent Multi-Sensor Fire Safety Network”

A

Project Report

*submitted in partial fulfillment of the
requirements for the award of the degree of*

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

Name

Piyush Kumar

Sonali Bhadra

Dhruv Tandon

Roll no.

R2142220342

R2142220179

R2142220067

under the guidance of

Dr. Rohit Tanwar



School of Computer Science

University of Petroleum & Energy Studies

Bidholi, Via Prem Nagar, Dehradun, Uttarakhand

November – 2024

CANDIDATE'S DECLARATION

I/We hereby certify that the project work entitled “**PyroSense: An Intelligent Multi-Sensor Fire Safety Network**” in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in Cyber Security and Forensics and submitted to the System Department, School of Computer Science, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/our work carried out during a period from **August, 2024** to **November, 2024** under the supervision of **Dr. Rohit Tanwar, Associate Professor, School of Computer Science, UPES Dehradun**.

The matter presented in this project has not been submitted by me/us for the award of any other degree of this or any other University.

Piyush Kumar (R2142220342)
Sonali Bhadra (R2142220179)
Dhruv Tandon (R2142220067)

This is to certify that the above statements made by the candidate is correct to the best of my knowledge.

Date: -11-2024

Dr. Rohit Tanwar
Project Guide

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide Dr. Rohit Tanwar, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thanks to our respected **Dr. Ajay Prasad, Cluster Head, School of Computer Science** for his great support in doing our project.

We are also grateful to Dean SoCS UPES for giving us the necessary facilities to carry out our project work successfully. We also thanks to our Course Coordinator and our Activity Coordinator Dr. Gaytri Bakshi for providing timely support and information during the completion of this project.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us.

Name	Piyush Kumar	Sonali Bhadra	Dhruv Tandon
Roll No.	R2142220342	R2142220179	R2142220067

ABSTRACT

Fire safety is a critical concern across residential, commercial, and industrial domains. Traditional fire alarm systems often rely on single-sensor mechanisms, which can lead to false alarms, delayed responses, and limited reliability. In this context, PyroSense: An Intelligent Multi-Sensor Fire Safety Network offers a scalable, IoT-based solution to these challenges.

PyroSense integrates multiple sensors, including MQ2 for smoke detection, MQ135 for gas monitoring, flame sensor for detecting flame, and DHT11 for temperature and humidity measurement. These sensors are controlled by ESP8266 modules, enabling distributed intelligence for real-time monitoring. The system is enhanced with SIM800L module that provide instant alerts via SMS and phone calls. Data visualization is implemented through Home Assistant running on a Raspberry Pi 3B, offering actionable insights and historical logging.

Key features of PyroSense include multi-sensor data fusion, reducing false positives by combining readings from various sources; scalability for diverse deployment scenarios; and automated alerts to ensure timely intervention. The system is designed with affordability and ease of implementation in mind, making it suitable for both small-scale and large-scale fire safety applications. PyroSense sets a benchmark for intelligent fire safety by combining IoT technology with a robust, multi-sensor architecture to minimize risks and protect lives and assets.

TABLE OF CONTENTS

S.No.	Contents	Page No
1.	Introduction	1
1.1.	History	1
1.2.	Requirement Analysis	1
1.3.	Main Objective	1
1.4.	Sub Objectives	1
1.5.	Pert Chart Legend	1
2.	System Analysis	2
2.1.	Existing System	2
2.2.	Motivations	2
2.3.	Proposed System	3
2.4.	Modules	3
2.4.1.	Sensor Data Acquisition and Processing	3
2.4.2.	Fire Detection and Alarm System	3
2.4.3.	Decision-Making and Alert System	3
2.4.4.	Real-time Monitoring and Visualization	4
2.4.5.	Communication and Response System	4
2.4.6.	System Configuration and User Management	4
3.	Design	4
3.1.	Modelling in PyroSense	5
3.1.1.	System Architecture	5
3.1.2.	Block Diagram	6
4.	PyroSense	8
4.1.	PyroSense Overview	8
4.2.	ESP8266 – PyroSense’s Brain	8
4.2.1	Technical Specifications	8
4.2.2	Features	8
4.2.3	Architecture-memory organization	9

4.3. Programming the ESP8266	9
4.3.1. Quick Start Guide	9
4.4. PyroSense's Sensors	9
4.4.1. Smoke Sensor (MQ2)	9
4.4.2. Flame Sensor	10
4.4.3. CO2 Sensor (MQ135)	10
4.4.4. Temperature and Humidity Sensor (DHT11)	10
4.5. SIM800L Module for Alerts	10
4.6. Real-time Monitoring and Visualization (Home Assistant)	10
5. Implementation	11
5.1. Sensor Data Collection and Processing	11
5.2. Scenarios	11
5.3.1. Scenario 1: Normal Operation (No Fire Detected)	11
5.3.2. Scenario - 2 Fire Detection Scenario	11
5.3. Algorithms	12
5.3.1. Scenario - 1 Threshold-based Detection Algorithm	12
5.3.2. Scenario - 2 Real-Time Monitoring Algorithm	13
6. Output screens	13
7. Limitations and Future Enhancements	17
8. Conclusion	18
Appendix A: Hardware Components and Features	18
Appendix B: Breadboarding Rules	19
References	19

LIST OF FIGURES

S.No.	Figure	Page No
1.	Fig. 1.1 Pert Chart	1
2.	Fig. 3.1 System Architecture	5
3.	Fig. 3.2 Block Diagram	6
4.	Fig. 5.1 Fire Detection Scenario	12
5.	Fig. 6.1 CO2 Concentration	13
6.	Fig. 6.2 Gas Sensor Output	14
7.	Fig. 6.3 Flame Sensor Output	14
8.	Fig. 6.4 Temperature and Humidity Sensor Output	14
9.	Fig. 6.5 Dashboard	15
10.	Fig. 6.6 Local Alarm(LED)	15
11.	Fig. 6.7 SMS Notification	16
12.	Fig. 6.8 Phone Call	16
13.	Fig. 6.9 Residential Monitoring	16
14.	Fig.6.10 Logs	17

1. INTRODUCTION

1.1. History

Fire safety has historically been addressed using single-sensor alarm systems that detect smoke or heat to trigger alarms. While these systems have been effective in raising awareness, they often suffer from limitations such as high false alarm rates and inadequate response mechanisms. Over the years, advancements in IoT and sensor technology have provided opportunities to enhance fire detection systems by integrating multiple sensors and enabling real-time decision-making. **PyroSense** builds on this progress, leveraging a multi-sensor network to address the limitations of traditional systems and provide a robust, intelligent fire safety solution.

1.2. Requirement Analysis

Developing PyroSense involved analyzing the following key requirements:

- Real-time monitoring of fire-prone environments using multi-sensor data.
- Reduction in false alarm rates by implementing data fusion techniques.
- Immediate alert mechanisms, including SMS and call-based notifications.
- Scalability to adapt to different environments, from residential spaces to industrial facilities.
- Cost-effectiveness to ensure accessibility for diverse users.
- Easy-to-deploy and maintain architecture for non-technical users.

1.3. Main Objective

To design and implement an **IoT-based multi-sensor fire safety system** that enhances reliability and reduces response time during fire incidents by integrating smoke, heat, flame and gas sensors with advanced alert mechanisms.

1.4. Sub Objectives

1. Implement a distributed sensor network for real-time data collection.
2. Minimize false positives using data fusion algorithms.
3. Develop a centralized visualization system through Home Assistant.
4. Enable automated emergency notifications using SIM800L module.
5. Ensure scalability and affordability for widespread adoption.
6. Incorporate historical data logging for performance analysis and improvements.

1.5. PERT Chart Legend

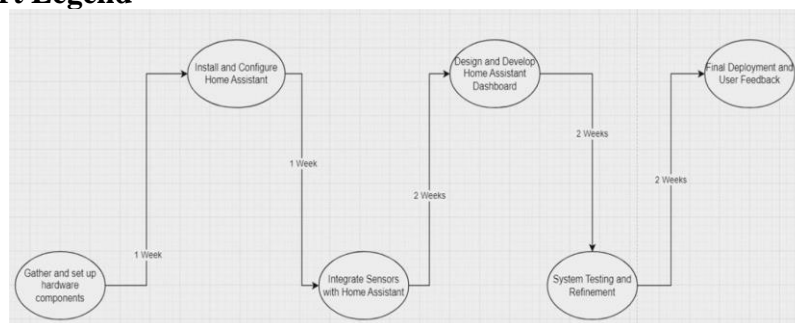


Fig 1.1: PERT Chart

- **Tasks:** Major activities involved in system design, development, and deployment.
- **Dependencies:** Logical sequence of tasks to ensure successful project completion.
- **Duration:** Estimated time for each task, including delays and contingencies.
- **Milestones:** Critical points in the project that signify the completion of a major deliverable.
- **Critical Path:** Sequence of dependent tasks determining the minimum project duration.

2. SYSTEM ANALYSIS

2.1 Existing System

Several researchers have proposed innovative IoT-based fire detection and response systems tailored to various applications. A. Rehman, M. A. Qureshi et al. developed a Smart Fire Detection and Deterrent System that uses flame sensors to monitor small indoor spaces and activates sprinklers and ventilation systems for fire control. Similarly, M. Arefin, A. Rahman et al. integrated flame sensors with alarms and water supply mechanisms to provide early warnings and coordinated responses to fire hazards. Yadav and Rani (2022) introduced a sensor-based system combining temperature and smoke sensors for real-time, scalable fire detection, while Komalapati et al. (2021) employed multi-sensor data and video surveillance for enhanced situational awareness. Hossain et al. (2022) focused on accuracy and robustness in residential settings by integrating temperature, gas, and smoke sensors with wireless protocols. Hasan Shuvo and Chowdhury (2024) utilized MQ-4 gas and LM35 temperature sensors for cost-effective monitoring in high-risk zones, and Kumaran et al. (2023) enhanced smoke detection accuracy with machine learning algorithms using temperature and humidity data. Alqourabah (2020) designed a system with automatic water sprinklers activated by temperature and smoke sensors, reducing response times. Sarkar developed a scalable framework leveraging cloud platforms for efficient monitoring, while Ajith (2018) incorporated GPS-based path navigation for safe evacuation during emergencies. Finally, Hassin (2021) presented a ubiquitous IoT-based fire detection and security system optimized for smart buildings, ensuring rapid hazard detection and mobile notifications.

2.2 Motivations

Fire safety systems have evolved significantly, yet they often rely on conventional mechanisms that fail to address challenges like real-time data analysis, early detection, and multi-sensor integration. Modern fires, caused by industrial hazards or environmental conditions, demand advanced solutions. The PyroSense project aims to fill this gap by offering a highly integrated IoT-based fire detection and response system that not only provides early warning signals but also enhances decision-making and response efficiency. The motivation stems from the need for:

- Improved safety measures in high-risk zones.
- Integration of real-time environmental monitoring and fire detection.
- Scalability and adaptability for both residential and commercial applications.
- An intelligent approach to minimize false alarms and ensure rapid communication with emergency services.

2.3 Proposed System

The PyroSense system is an intelligent multi-sensor network designed to monitor environmental parameters, detect fire incidents, and send alerts for immediate action. It employs an IoT-based infrastructure that includes sensors for smoke, temperature, gas, and heat, managed by ESP8266 modules and Raspberry Pi. The system integrates with Home Assistant for real-time data visualization and uses SIM800L modules for automatic SMS and phone call alerts. Key features include:

- Multi-sensor data fusion for accurate fire detection.
- Real-time monitoring through a user-friendly interface.
- Configurable alert thresholds to minimize false alarms.
- Scalable design supporting residential, industrial, and commercial use cases.

2.4 Modules

2.4.1. Sensor Data Acquisition and Processing

This module handles data collection from sensors like:

- **MQ2 (smoke)**: Detects smoke particles.
- **MQ135 (CO2)**: Monitors air quality for carbon dioxide levels.
- **Flame Sensor**: Detects infrared radiation to measure heat levels.
- **DHT11 (temperature & humidity)**: Tracks environmental temperature and humidity.

The data from these sensors is processed by ESP8266 microcontrollers, ensuring precise measurement and filtering of noise for accurate fire detection.

2.4.2. Fire Detection and Alarm System

This module processes the acquired sensor data using threshold-based algorithms to determine the likelihood of a fire. If sensor readings surpass predefined safety limits, the alarm system is triggered. Features include:

- **Visual (LEDs) and auditory (buzzer) alarms** for immediate notification to occupants in the vicinity.
- A **multi-sensor decision approach** to minimize false positives by combining data from multiple sensors.
- Automatic **SMS and phone call alerts** sent via SIM800L modules to pre-configured emergency contacts, ensuring rapid communication with homeowners, facility managers, or emergency responders.
- Prioritization of alerts based on fire severity, ensuring critical situations are addressed first.

2.4.3. Decision-Making and Alert System

The decision-making module leverages processed data to classify fire severity levels. In case of an emergency:

- SMS and phone call alerts are sent via SIM800L modules to pre-configured contacts.
- Alerts are prioritized based on the risk severity.
- Advanced algorithms allow distributed intelligence to ensure decisions are made locally at each node for faster response.

2.4.4. Real-time Monitoring and Visualization

The system integrates with **Home Assistant** running on a Raspberry Pi for real-time data visualization. Features include:

- A dashboard displaying live sensor readings, alert levels, and system status.
- Historical data storage for analyzing trends and system performance.
- User-configurable settings to adjust thresholds and notification preferences.

2.4.5. Communication and Response System

This module ensures seamless communication between the system and external entities:

- SIM800L send notifications to emergency responders, homeowners, or facility managers.
- MQTT protocol facilitates communication between IoT devices, ensuring low-latency alerts.
- The system can be integrated with automated sprinkler systems for proactive response.

2.4.6. System Configuration and User Management

- Allows users to configure sensor thresholds, add or remove contacts, and define alert escalation protocols.
- Offers role-based access for administrators, operators, and end-users.
- Ensures secure communication through encrypted channels to prevent unauthorized access.

3. DESIGN

The PyroSense system is designed to detect fire hazards in real time and provide immediate alerts to stakeholders, ensuring safety and damage prevention. The design focuses on scalability, efficiency, and modularity, allowing for easy integration of additional sensors and communication modules.

The system is divided into layers, each responsible for a specific function, from sensing environmental parameters to processing data and communicating with external systems. The design also incorporates fault tolerance, ensuring reliable operations under various environmental conditions.

Key design considerations include:

- **Accuracy and Sensitivity:** The system integrates high-quality sensors to detect gas levels, flames, smoke, temperature, and humidity with precision.
- **Real-Time Processing:** ESP8266 and Raspberry Pi 3B ensure real-time monitoring and alert generation.
- **Scalability:** The system is modular, allowing additional sensors or communication methods to be added as needed.
- **Cost-Effectiveness:** Open-source platforms like Home Assistant, paired with affordable hardware, make the system economically viable.

The design also emphasizes user accessibility, providing a dashboard for intuitive data visualization and control.

3.1 Modelling in PyroSense

The PyroSense system is modeled as a multi-layered IoT solution that integrates environmental sensing, local processing, centralized monitoring, and emergency communication. The modeling approach ensures a clear flow of data and actions, from sensing to output generation.

3.1.1 System Architecture

The PyroSense system is designed with a layered architecture to efficiently monitor and manage fire detection, environmental sensing, and communication with stakeholders. The architecture consists of the following layers:

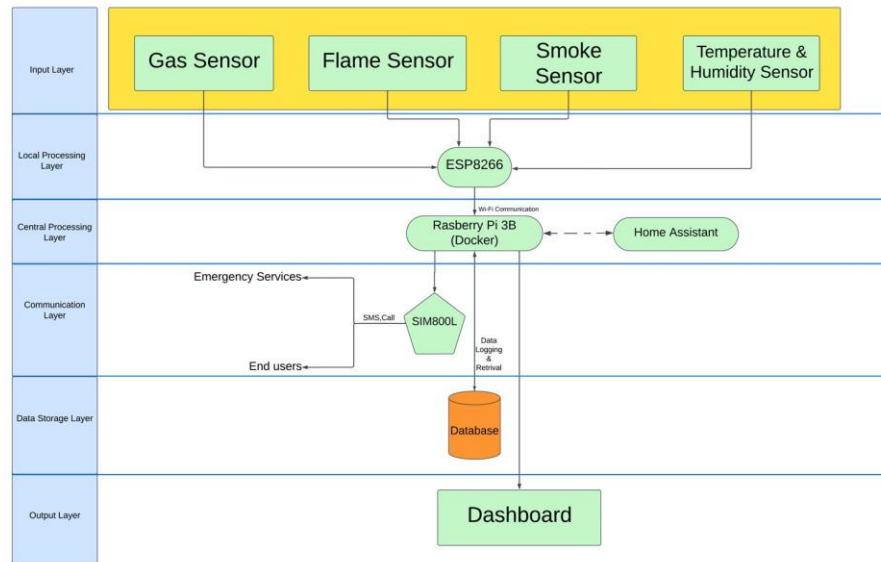


Fig 3.1: System Architecture

1. Input Layer:

This layer consists of multiple sensors integrated for collecting data from the environment. These sensors include:

- **Gas Sensor:** Detects harmful gases in the environment.
- **Flame Sensor:** Identifies the presence of flames to detect fire.
- **Smoke Sensor:** Detects smoke particles to ensure early fire detection.
- **Temperature and Humidity Sensor:** Monitors environmental conditions for anomalies.

2. Local Processing Layer:

The data collected from the sensors is sent to the **ESP8266 microcontroller**, which

acts as the local processing unit. The ESP8266 preprocesses the data and communicates critical information to the central system using Wi-Fi.

3. **Central Processing Layer:**

The central processing unit is a **Raspberry Pi 3B**, running a **Dockerized Home Assistant** instance. This layer serves as the core control system for managing sensor data, triggering alerts, and maintaining communication with external systems. The Raspberry Pi interfaces with:

- **SIM800L Module** for emergency notifications via SMS and calls.
- **Database** for storing sensor data and logs for future analysis and retrieval.

4. **Communication Layer:**

This layer ensures reliable communication between the system and external stakeholders:

- **Emergency Services** are notified in case of a fire.
- **End Users** receive alerts and can monitor real-time data through a user-friendly dashboard.

5. **Data Storage Layer:**

Data is logged into a centralized **database** for historical tracking, analysis, and further processing. This enables trends to be identified and system performance to be evaluated.

6. **Output Layer:**

This layer includes a **dashboard** for visualizing sensor readings, alerts, and system performance. Users can access the dashboard to monitor environmental conditions and system status.

3.1.2 Block Diagram

The block diagram of PyroSense provides a detailed visual representation of its components and their interconnections. The major components include:

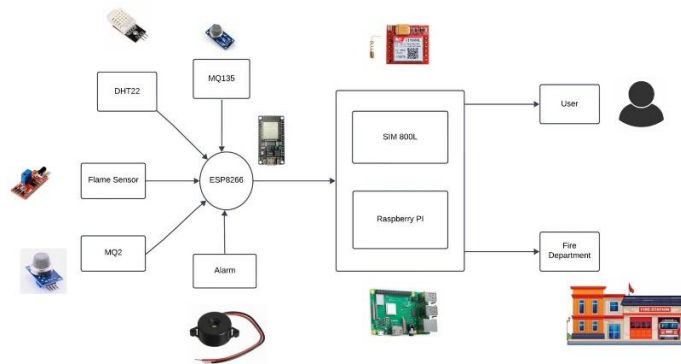


Fig 3.2: Block Diagram

1. **Sensors:**

- **MQ135:** Detects air quality and harmful gases like CO₂.
- **MQ2:** Senses gases such as LPG and propane for fire detection.
- **Flame Sensor:** Detects flame presence for immediate alert.
- **DHT22:** Measures temperature and humidity to identify environmental anomalies.

2. **ESP8266 Microcontroller:**

Acts as the central hub for collecting data from sensors, preprocessing it, and forwarding critical alerts to the Raspberry Pi.

3. **Raspberry Pi 3B:**

- Processes data from ESP8266.
- Hosts Home Assistant for central monitoring and control.
- Interfaces with the SIM800L module for communication with stakeholders.

4. **SIM800L Module:**

Sends SMS and makes calls to emergency services and end users during critical situations.

5. **Output Systems:**

- **Alarm System:** Activated upon detecting a critical hazard to alert nearby individuals.
- **Dashboard:** Displays real-time data for users and system administrators.

6. **Communication with Emergency Services:**

In case of fire detection, the system automatically alerts the fire department and end users via calls and SMS.

This layered and modular architecture ensures efficient sensing, processing, communication, and monitoring, making PyroSense a reliable fire detection and management system.

4. PYROSENSE

PyroSense is a smart, IoT-based fire detection and prevention system that integrates multiple environmental sensors to monitor fire-related hazards in real time. By combining precise sensing technologies, reliable communication modules, and intuitive dashboards, PyroSense provides a comprehensive solution for ensuring safety in residential, commercial, and industrial environments.

4.1 PyroSense Overview

The PyroSense system leverages advanced sensing, processing, and communication technologies to detect fire hazards like smoke, gas, flame, and abnormal temperature or humidity levels. It uses an ESP8266 microcontroller for local data processing, a Raspberry Pi for central monitoring, and a SIM800L module for emergency alerts.

The system aims to provide:

- Early warning alerts via SMS and calls.
- Continuous monitoring with data visualization through Home Assistant.
- Scalability for integration of additional sensors.
- High reliability in diverse environmental conditions.

4.2 ESP8266 - PyroSense's Brain

The ESP8266 serves as the primary processing unit for PyroSense, handling real-time data from the connected sensors and communicating with the Raspberry Pi. Its low cost, Wi-Fi capabilities, and ease of programming make it ideal for IoT applications.

4.2.1 Technical Specifications

- **Processor:** 32-bit Tensilica Xtensa LX106 microcontroller.
- **Clock Speed:** 80 MHz (can be overclocked to 160 MHz).
- **Flash Memory:** 4 MB (varies by module).
- **RAM:** 50 kB (for instructions) and 80 kB (data).
- **Wi-Fi:** 802.11 b/g/n with integrated TCP/IP stack.
- **GPIO Pins:** Up to 16, supporting PWM, ADC, I2C, SPI, and UART.

4.2.2 Features

- Built-in Wi-Fi support for wireless communication.
- Ultra-low power consumption in deep-sleep mode.
- Integrated ADC for analog signal processing.

- Supports multiple communication protocols (I2C, SPI, UART).
- Compact design, making it suitable for space-constrained applications.

4.2.3 Architecture and Memory Organization

The ESP8266 architecture is optimized for IoT applications, with:

- A Harvard architecture that separates instruction and data memory for faster processing.
- Integrated flash memory for storing firmware and user applications.
- SRAM used for real-time data operations and buffering.
- A memory map that organizes user code, stack, heap, and system functions.

4.3 Programming the ESP8266

The ESP8266 can be programmed using various IDEs, with the Arduino IDE being the most common due to its simplicity and vast community support.

4.3.1 Quick Start Guide

- **Install Arduino IDE:** Download and install the Arduino IDE from the official website.
- **Add ESP8266 Board:** Go to File > Preferences and add the ESP8266 board manager URL. Then, install the ESP8266 board from Tools > Board Manager.
- **Connect Hardware:** Use a USB-to-TTL converter or a development board like NodeMCU to connect the ESP8266 to your computer.
- **Write Code:** Load a basic program like the "Blink" example or Wi-Fi connection sketch to test functionality.
- **Upload Code:** Select the correct COM port and board type, then upload the code.

4.4 PyroSense's Sensors

PyroSense incorporates multiple sensors to monitor environmental parameters and detect fire-related hazards.

4.4.1 Smoke Sensor (MQ2)

The MQ2 sensor detects smoke and flammable gases like LPG, propane, methane, and hydrogen.

- **Operating Voltage:** 5V DC.
- **Output:** Analog and digital signals.
- **Application:** Detects smoke levels and triggers alerts when thresholds are crossed.

4.4.2 Flame Sensor

The flame sensor detects infrared light emitted by flames.

- **Operating Voltage:** 3.3V to 5V DC.
- **Output:** Digital signal.
- **Features:** Adjustable sensitivity via potentiometer.

4.4.3 CO2 Sensor (MQ135)

The MQ135 measures air quality and CO2 concentration.

- **Operating Voltage:** 5V DC.
- **Output:** Analog signal indicating gas concentration.
- **Application:** Detects CO2 levels to identify combustion or hazardous air quality.

4.4.4 Temperature and Humidity Sensor (DHT11)

The DHT11 provides temperature and humidity readings.

- **Operating Voltage:** 3.3V to 5V DC.
- **Output:** Digital signal.
- **Specifications:**
 - Temperature Range: 0°C to 80°C.
 - Humidity Range: 20% to 90%.

4.5 SIM800L Module for Alerts

The SIM800L handles communication with emergency services and end users by sending SMS or making calls when hazards are detected.

Features:

- Quad-band GSM for worldwide use.
- SMS, call, and GPRS capabilities.
- Low power consumption, ideal for IoT.

Application in PyroSense: The module is programmed to alert users and emergency services with location-based details and the type of detected hazard.

4.6 Real-Time Monitoring and Visualization (Home Assistant)

Home Assistant serves as the central dashboard for PyroSense, allowing users to monitor sensor readings in real time.

Features:

- Easy integration with ESP8266 via MQTT.
- Customizable dashboards for visualizing temperature, humidity, gas concentration, and flame detection.
- Automation triggers for specific events, such as turning on alarms or notifying users.

With its open-source nature, Home Assistant provides flexibility for future upgrades, such as adding more sensors or advanced analytics.

5. IMPLEMENTATION

The implementation phase involves integrating all components of the PyroSense system, processing sensor data, and handling specific scenarios through predefined algorithms. It focuses on real-time data collection, decision-making, and notification delivery to ensure reliable fire detection and response.

5.1 Sensor Data Collection and Processing

Sensor data collection is the backbone of PyroSense, where the system continuously gathers real-time inputs from multiple sensors. The collected data is processed using the ESP8266 microcontroller, which analyzes and transmits the results for visualization or emergency action.

Data Collection Steps:

- **Reading Sensor Outputs:** Each sensor (MQ2, MQ135, DHT11, Flame Sensor) sends analog or digital signals representing environmental conditions.
- **Normalization:** Raw sensor data is normalized to match threshold values for fire detection.
- **Filtering:** Noisy sensor readings are filtered using techniques like moving average or Kalman filters.
- **Threshold-Based Decisions:** The processed data is compared against predefined thresholds to identify potential fire hazards.

Data Flow:

- Sensors → ESP8266 (Processing Unit)
- ESP8266 → Raspberry Pi (Central Node for Real-Time Monitoring)
- Raspberry Pi → Home Assistant (Visualization) and SIM800L (Alert Mechanism).

5.2 Scenarios

To evaluate PyroSense's effectiveness, it is tested in real-world scenarios simulating fire hazard situations. These scenarios validate the system's reliability and response time.

5.3.1 Scenario 1: Normal Operation (No Fire Detected)

Description: In this scenario, all sensors read normal environmental parameters. The system continues to monitor the environment without triggering any alerts. Data is transmitted regularly to the Home Assistant platform for real-time monitoring.

5.3.2 Scenario 2: Fire Detection Scenario

This scenario simulates a situation where multiple fire parameters are simultaneously detected.

Implementation:

- The MQ2 detects smoke or gas, while the flame sensor identifies direct flames.
- DHT11 monitors the rise in temperature and humidity drop, which are common during fires.
- The MQ135 measures CO2 levels to confirm combustion.

- If all conditions match the fire detection criteria, the system activates the alarm and sends SMS/call notifications.



Fig 5.1: Fire Detection Scenario

Applications:

- Residential and commercial spaces where early fire detection is crucial.

5.3 Algorithms

The following algorithms are used for implementing the fire detection scenarios and ensuring accurate real-time monitoring.

5.3.1 Threshold-Based Detection Algorithm

This algorithm uses predefined threshold values for each sensor to identify fire-related hazards.

Steps:

1. Read sensor values:
 - MQ2: Smoke or gas level.
 - MQ135: CO2 concentration.
 - Flame Sensor: Presence of flames.
 - DHT11: Temperature and humidity levels.
2. Compare each value with its threshold:
 - **Smoke Threshold:** > 300 PPM (parts per million).
 - **CO2 Threshold:** > 1000 PPM.
 - **Temperature Threshold:** > 50°C.
 - **Humidity Drop:** Below 20%.
3. If any two or more thresholds are exceeded, trigger the alarm.
4. Send alerts using the SIM800L module.
5. Log the data and display it on the Home Assistant dashboard.

Pseudo-code:

```
if smoke_level > smoke_threshold or co2_level > co2_threshold:
    if temperature > temperature_threshold or humidity < humidity_threshold:
        trigger_alarm()
        send_sms_alert()
        log_data_to_dashboard()
```

5.3.2 Real-Time Monitoring Algorithm

This algorithm ensures continuous monitoring and visualization of sensor data on Home Assistant.

Steps:

- Initialize the ESP8266 and establish communication with all sensors.
- Collect sensor data every 2 seconds and send it to the Raspberry Pi.
- Raspberry Pi processes the data and updates the Home Assistant dashboard via MQTT.
- If any sensor value exceeds its threshold:
 - Activate alerts (SMS, call).
 - Log the event in the system.
- Repeat the process continuously.

Pseudo-code:

```
while True:
    sensor_data = read_all_sensors()
    process_data(sensor_data)
    send_to_dashboard(sensor_data)
    if any(sensor_value > threshold for sensor_value in sensor_data):
        trigger_alert()
        log_event()
```

6. OUTPUT SCREENS

6.1 Sensor Monitoring and Alerts

The PyroSense system uses multiple sensors to monitor environmental conditions and trigger alerts when thresholds are exceeded.

Below are the outputs from each sensor:

1. MQ135 Air Quality Sensor

- **Description:** Monitors CO₂ levels between 10–2,000 PPM. Alerts are triggered when levels exceed 800 PPM.
- **Sample Output:** Real-time CO₂ concentration displayed on the Home Assistant dashboard.



Fig 6.1: CO₂ Concentration

2. MQ2 Gas Sensor

- **Description:** Detects combustible gases like methane, LPG, and hydrogen within the range of 200–2,000 PPM.
- **Sample Output:** Alarm activation upon gas concentration exceeding 300 PPM.



Fig 6.2: Gas Sensor Output

3. Flame Sensor

- **Description:** Detects flame presence based on infrared light in the wavelength range of 760–1100 nm.
- **Sample Output:** Immediate alarm triggered for detected flames.



Fig 6.3: Flame Sensor Output

4. DHT22 Temperature and Humidity Sensor

- **Description:** Monitors temperature (-40°C to 80°C) and humidity (0%–100%).
- **Sample Output:** Temperature or humidity threshold breach displayed on the dashboard.



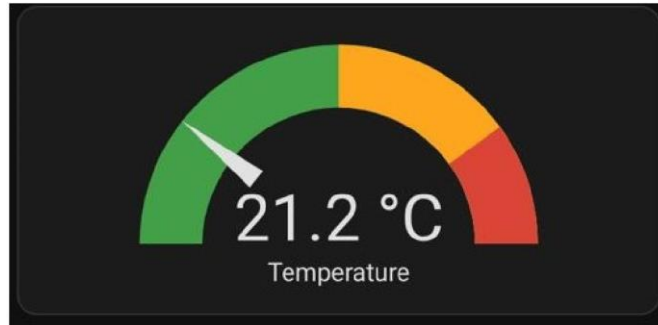


Fig 6.4: Temperature and Humidity Sensor Output

6.2 System Integration

The data from all sensors is processed in real-time and visualized using Home Assistant. Alerts and notifications are triggered as per the configured thresholds.

- **Home Assistant Dashboard:** Displays sensor readings, system status, and alert history.



Fig 6.5: Dashboard

6.3 Alert Mechanisms

The PyroSense system includes multiple layers of alerts to ensure rapid response to fire hazards.

1. Local Alarms:

- **Description:** A buzzer and flashing light are activated immediately when fire is detected.



Fig 6.6: Local Alarm (LED)

2. SMS Notifications:

- **Description:** Emergency alerts sent to users and emergency services with sensor details.

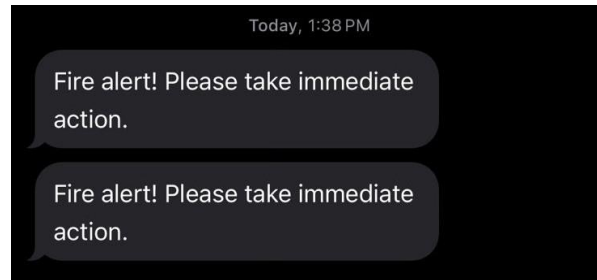


Fig 6.7: SMS Notification

3. Phone Calls:

- **Description:** Automated phone calls made to alert emergency contacts.



Fig 6.8: Phone Call

6.4 Use Case Scenarios

1. Residential Monitoring:

- **Scenario:** Fire detected in a living room. The system activates a local alarm and sends alerts.



Fig 6.9: Residential Monitoring

2. Industrial Application:

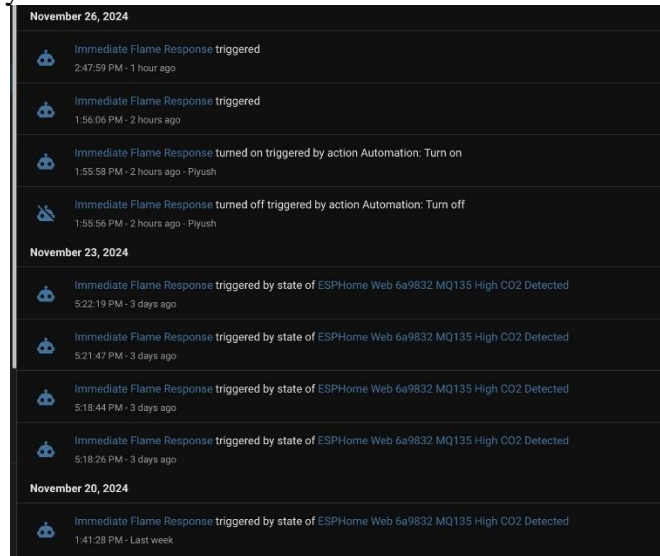
- **Scenario:** CO2 levels exceed thresholds in a warehouse.

3. Forest Fire Monitoring:

- **Scenario:** Flame sensor detects fire in a monitored forest area.

6.5 Alert Visualization and Logs

- **Historical Data:** The Home Assistant interface provides logs of past events for performance analysis and audits.



The screenshot displays a list of logs from the Home Assistant interface. The logs are organized by date, with sections for November 26, 2024; November 23, 2024; and November 20, 2024. Each log entry includes a flame icon, a title, and a timestamp. The events are categorized as 'Immediate Flame Response' triggered, 'turned on', or 'turned off', with some entries specifying triggers like 'Automation: Turn on' or 'ESPHome Web 6a9832 MQ135 High CO2 Detected'.

Date	Event	Timestamp
November 26, 2024	Immediate Flame Response triggered	2:47:59 PM - 1 hour ago
	Immediate Flame Response triggered	1:56:06 PM - 2 hours ago
	Immediate Flame Response turned on triggered by action Automation: Turn on	1:55:58 PM - 2 hours ago - Piyush
	Immediate Flame Response turned off triggered by action Automation: Turn off	1:55:56 PM - 2 hours ago - Piyush
November 23, 2024	Immediate Flame Response triggered by state of ESPHome Web 6a9832 MQ135 High CO2 Detected	5:22:19 PM - 3 days ago
	Immediate Flame Response triggered by state of ESPHome Web 6a9832 MQ135 High CO2 Detected	5:21:47 PM - 3 days ago
	Immediate Flame Response triggered by state of ESPHome Web 6a9832 MQ135 High CO2 Detected	5:18:44 PM - 3 days ago
	Immediate Flame Response triggered by state of ESPHome Web 6a9832 MQ135 High CO2 Detected	5:18:26 PM - 3 days ago
November 20, 2024	Immediate Flame Response triggered by state of ESPHome Web 6a9832 MQ135 High CO2 Detected	1:41:28 PM - Last week

Fig 6.10: Logs

7. LIMITATIONS AND FUTURE ENHANCEMENTS

Limitations:

- **Sensor Accuracy:** The sensors used in PyroSense, such as MQ2 and MQ135, are cost-effective but lack high precision, potentially leading to false positives or delayed detection in certain cases.
- **Power Dependency:** The system requires uninterrupted power supply, which might be a challenge in remote or disaster-prone areas.
- **Scalability Constraints:** Although designed for scalability, the system's performance may degrade with a significant increase in the number of sensors or nodes.
- **Network Dependency:** The reliance on Wi-Fi and GSM for communication may lead to delays or failures in notifications in low-network coverage areas.
- **Environment-Specific Calibration:** Sensors require recalibration based on the deployment environment, which can be cumbersome.

Future Enhancements:

- **Integration with AI:** Employ machine learning algorithms to improve detection accuracy and predict fire hazards based on environmental data trends.
- **Battery Backup:** Incorporate battery support with solar charging capabilities to ensure the system remains operational during power outages.
- **Enhanced Sensors:** Use advanced sensors with higher sensitivity and reliability to reduce false alarms and improve detection speed.
- **Edge Computing:** Implement edge-based analytics for faster decision-making and reduced reliance on central processing.
- **Mobile App Integration:** Develop a dedicated mobile application for real-time monitoring, remote configuration, and alert notifications.
- **Automated Fire Suppression:** Integrate with water sprinklers or fire extinguishing systems for immediate response to fire incidents.

8. CONCLUSION

The PyroSense project demonstrates a robust and intelligent fire safety solution using IoT-based technologies. By integrating multiple sensors, real-time monitoring, and automated alert mechanisms, it significantly improves fire detection accuracy and response times. The system's modular design ensures scalability, making it adaptable to residential, commercial, and industrial applications. While there are limitations, the proposed future enhancements outline a clear path for addressing these challenges and expanding the system's capabilities. PyroSense not only sets a benchmark in fire safety systems but also highlights the potential of IoT in addressing critical safety concerns.

Appendix A: Hardware Components and Features

1. Sensors:

- **MQ2 Smoke Sensor:** Detects smoke and flammable gases.
 - Operating Voltage: 5V DC.
 - Output: Analog/Digital.
- **MQ135 Gas Sensor:** Measures air quality and CO2 levels.
 - Operating Voltage: 5V DC.
 - Output: Analog.
- **Flame Sensor:** Identifies infrared radiation from flames.
 - Operating Voltage: 3.3V to 5V DC.
 - Output: Digital.
- **DHT11 Sensor:** Measures temperature and humidity.
 - Operating Voltage: 3.3V to 5V DC.
 - Output: Digital.

2. Microcontrollers and Modules:

- **ESP8266:** Central processing unit for sensor data.
 - Features: Wi-Fi support, low power consumption.
- **Raspberry Pi 3B:** Hosts the Home Assistant dashboard.
 - Features: Decent processing power, modularity.
- **SIM800L:** Sends SMS and calls for alerts.
 - Features: Quad-band GSM, low power usage.

Appendix B: Breadboarding Rules

- **Avoid Overcrowding:** Space out components to prevent accidental short circuits and ease troubleshooting.
- **Use Proper Wire Lengths:** Avoid excessively long or tight wires to reduce signal interference and maintain neatness.
- **Power Supply Management:** Ensure all components receive the correct voltage and current ratings to prevent damage.
- **Color Code Wires:** Use different colors for VCC, GND, and signal wires for easy identification.
- **Secure Connections:** Double-check all connections to ensure they are firm and not loose, which can cause intermittent issues.
- **Test Incrementally:** Test each module or sensor separately before integrating them into the system.
- **Document Connections:** Maintain a clear diagram or chart of connections to refer back during debugging.

References

- [1] A. Rehman, M. A. Qureshi, T. Ali, M. Irfan, S. Abdullah, S. Yasin, U. Draz, A. Glowacz, G. Nowakowski, A. Alghamdi, A. A. Alsultan, "Multi-Sensor Fire Detection System Using IoT," IEEE Transactions on Industry Applications, vol. 58, no. 6, pp. 6441-6450, 2022.
- [2] M. Arefin, M. A. Hussen Wadud, A. Rahman, F. I. Islam "An IoT-based Integrated Solution for Fire Detection Alarm System and Water Supply Management" ResearchGate, 2023.
https://www.researchgate.net/publication/375058126_An_IoT-based_Integrated_Solution_for_Fire_Detection_Alarm_System_and_Water_Supply_Management
- [3] R. Yadav, P. Rani, "Sensor based smart fire detection and fire alarm system," SSRN Electronic Journal, 2022. <https://doi.org/10.2139/ssrn.3724291>
- [4] N. Komalapati, V. C. Yarra, L. A. V. Kancharla, T. N. Shankar, "Smart fire detection and surveillance system using IoT," in 2021 International Conference on Artificial Intelligence and

Smart Systems (ICAIS), 2021, pp. 1208-1213. IEEE.
<https://doi.org/10.1109/ICAIS50930.2021.9395841>

[5] T. Hossain, M. Ariful Islam, A. B. R. Khan, M. Sadekur Rahman, "A Robust and Accurate IoT-Based Fire Alarm System for Residential Use," in A. P. Pandian, X. Fernando, W. Haoxiang (Eds.), *Computer Networks, Big Data and IoT*, Springer Singapore, 2022, pp. 479–492.
https://doi.org/10.1007/978-981-19-0898-9_37

[6] M. M. Hasan Shuvo, S. Chowdhury, "IoT Based Fire Detector System Using MQ-4 and LM35 Sensor," in *2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS)*, 2024, pp. 1-6. IEEE. <https://doi.org/10.1109/ICSCSS60660.2024.10625187>

[7] S. Kumaran, S. Arunachalam, V. Surendar, T. Sudharsan, "IoT based Smoke Detection with Air Temperature and Air Humidity; High Accuracy with Machine Learning," in *2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, 2023, pp. 1–6. IEEE. url <https://doi.org/10.1109/ICAIS56108.2023.10073920>

[8] H. Alqourabah, A. Muneer, S. M. Fati, "A Smart Fire Detection System using IoT Technology With Automatic Water Sprinkler," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. xx-xx, 2020. <https://doi.org/10.11591/ijece.v9i4.ppxx-xx>

[9] S. Sarkar, "Smart Fire Detection System Using IoT," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. xx-xx.
<https://doi.org/10.11591/ijece.v9i4.ppxx-xx>

[10] G. Ajith, J. Sudarsaun, S. Dhilipan Arvind, R. Sugumar, "IoT Based Fire Deduction and Safety Navigation System," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 7, Special Issue, Mar. 2018, pp. xx-xx. ISSN(Online): 2319-8753. ISSN (Print): 2347-6710

[11] M. E. Hassin, A. Al Neon, S. Sabila, R. M. Rahman, "Smart Fire Detection and Security System," in *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2021, pp. 0666-0671. IEEE.
<https://doi.org/10.1109/UEMCON53757.2021.9666623>