Journal of Information Technology Management
ISSN-OnLine: 2980-7972
University of Tehran

Secure Multi-Sensor IoT-Based Fire Detection System Using MQTT and TLS Encryption

Sara Seyam *

*Corresponding author, Student, Department of General Education, Liwa University, Abu Dhabi, UAE. E-mail: 1080365@students.adu.ac.ae

Sima Sabouni

Student, Department of General Education, Liwa University, Abu Dhabi, UAE. E-mail: 1079867@students.adu.ac.ae

Rasha Hassan

Assistant Prof., Senior Lecturer, Department of General Education, Liwa University, Abu Dhabi, UAE. E-mail: rasha.hasan@lc.ac.ae

Journal of Information Technology Management, 2025, Vol. 18, Issue 2, pp. 167-175
Published by the University of Tehran, College of Management
doi: https://doi.org/10.22059/jitm.2026.107236
Article Type: Research Paper

Received: December 07, 2025
Received in revised form: January 17, 2026
Accepted: February 21, 2026
Published online: March 01, 2026

Abstract

This paper presents an IoT-based fire detection system that integrates multiple sensors (MQ2, DHT11, and PIR) with an MQTT communication protocol to enhance detection accuracy and reduce false alarms. By combining multiple sensors, manual override mechanisms, and a real-time dash-board interface, the system allows emergency response while providing flexibility for user interven-tion in case of failures. The system is protected with TLS encryption in order to protect the data in-tegrity in such a crucial system. The tests taken regarding this system confirm the reliability of the alerts and the robustness of the system performance in simulated scenarios.

Keywords: Fire Detection, MQTT, PIR Sensor, MQ2 Sensor, TLS Encryption.

Introduction

Fire detection systems are a crucial part of every building, where the early detection of fire hazards plays an important role in mitigating damage, preventing loss of life, and minimizing property damage (Alshareef & Naif, 2023). Traditional fire protection systems based on single sensors, such as smoke detectors, tend to cause false alarms and lack features such as human presence confirmation and manual override capabilities (Hambarde & Proenca, 2023). Further challenges arise from the need for reliable and secure communication protocols and intuitive monitoring interfaces capable of providing real-time updates (Hassan & Tarig, 2022).

The main goal of the proposed system in this paper is to enhance current fire safety systems by integrating MQ2, DHT11, and PIR sensors with MQTT communication and a real-time dashboard (Hu, 2024). The system aims to improve fire detection accuracy by reducing false alarms, enhancing system reliability and flexibility through the implementation of a manual override mechanism, and streamlining data communication and visualization using MQTT protocols to transmit sensor data to a central Raspberry Pi broker (Kekevi & Arif, 2022).

Literature Review

In recent years, advancements in IoT technologies have encouraged the development of innovative fire safety monitoring solutions. However, limitations still exist in the integration of sensors, communication protocols, user interfaces, and cybersecurity measures (Tariq et al., 2022).

Palamar and Palamar (2023) proposed a system capable of real-time detection using smoke, CO, temperature, and flame sensors; however, the system lacked motion sensing and a centralized dashboard, which increased the possibility of false alarms and reduced user control. The proposed system in this paper addresses these limitations by integrating a PIR sensor for occupancy detection, an interactive dashboard, and manual override facilities.

Similarly, Mahgoub et al. (2019) presented an IoT-based system consisting of wireless nodes, MQTT communication, and a bridge node connected to a central Raspberry Pi. Although their system demonstrated successful functionality, it suffered from SMS notification delays of up to 27 seconds and limited security mechanisms. The proposed system improves upon this by implementing MQTT communication with TLS encryption to provide faster and more secure communication.

Lee et al. (2013) developed a fire monitoring solution that relied on costly infrastructure such as servers and VPNs while lacking manual override and encryption features. In contrast,

the proposed approach utilizes lightweight MQTT communication secured with TLS encryption, reducing infrastructure costs while supporting secure real-time overrides.

Furthermore, Udurume et al. (2025) integrated MQTT, ESP-NOW, and radio environment mapping for industrial applications but did not incorporate motion sensing or manual override capabilities. Likewise, Muheden et al. (2016) depended mainly on Wi-Fi communication and lacked sufficient security measures, motion sensing, and manual override mechanisms. The proposed system overcomes these limitations through PIR-based occupancy detection, TLS-secured MQTT communication, multi-sensor fusion, and emergency override controls..."

Arduino-based solutions proposed by Mahzan et al. (2018) and Patange and Yadav (2015) mainly relied on temperature sensors and SMS notifications while lacking robust interfaces and cybersecurity features. Such systems remain vulnerable to false alarms and lack occupant awareness and override facilities. Conversely, the proposed system integrates MQ2, DHT11, and PIR sensors to improve detection accuracy, reduce false alarms, and enhance occupant safety. Additionally, it provides secure real-time communication, a user-friendly dashboard, and advanced override mechanisms.

Methodology

System Design

Our fire alarm system consists of two main components to which all other hardware components are connected. First, the Raspberry Pi serves as the central hub for data collection and as the MQTT broker. As shown in Figure 6, three key sensors are directly connected to the Raspberry Pi: a temperature sensor (DHT11), a gas sensor (MQ2), and a PIR motion sensor. They all provide data constantly to the MQTT broker, with each publishing to a different topic. Second, the Node MCU (ESP8266) is a node that subscribes to the topic Fire Sensor/# to receive all messages that are published under the Fire Sensor category. To facilitate the logic of our fire alarm system, we included several visual and audio indicators. Four LEDs are connected, each resembling one of the system's statuses. In addition, a buzzer and sprinklers are attached to be activated in case of an emergency. Moreover, two buttons are connected. One is an emergency switch for manual activation of undetected emergencies, and another is a reset button to prevent unnecessary alerts / false alarms. Finally, we developed an interface to allow real-time monitoring of sensor readings and emergency conditions. This enables us to access it remotely and act in real time.

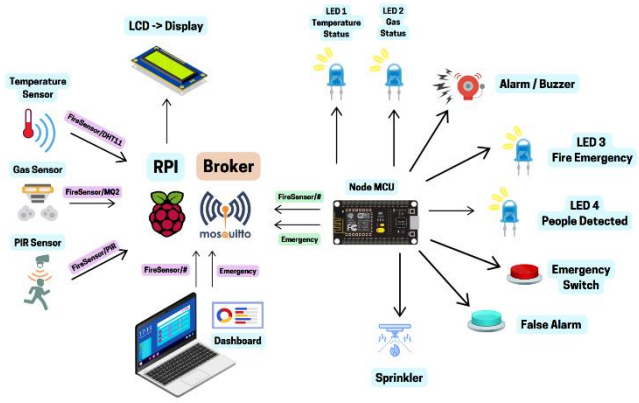

Fig. 1. System Architecture diagram

State Machine Diagram

As illustrated in Figure 1, our proposed Fire alarm system is composed of 3 main subsystems: the Raspberry Pi, Node MCU, and Dashboard. The Raspberry Pi keeps monitoring data from the sensors connected. Then continuously publish three key parameters, which are: smoke level, temperature, and motion detection, each through its respective MQTT topic. At the same time, the dashboard will receive updates for remote monitoring. The Node MCU subscribes to these topics and triggers different emergency protocols based on assigned thresholds. The logic goes if the smoke level exceeds 10/100 smoke level, the system enters a SMOKE_DETECTED state, turning on LED_1. Similarly, if the temperature surpasses 35°C, it enters the HEAT_DETECTED state, activating another LED_2. However, for a fire to be confirmed, we rely on the values from two sensors simultaneously. If both the temperature and smoke levels exceeded the assigned threshold, the system enters HOUSE_ON_FIRE mode, which turns on a loud buzzer, emergency LED_3, and a DC motor (simulating sprinklers). To prevent false alarms, a 60-second timeframe is set, where if the reset button is pressed, the system enters monitoring mode again. Otherwise, the system checks if motion is detected inside the premises. If evaluated to true, the system enters the PEOPLE_PRESENT state, issuing an evacuation alert to authorities. The system is also equipped with a manual override button, which causes emergency responses to be triggered instantly when activated. The final component integrated is the dashboard, which will provide real-time visualization of all sensor data and system states, aiming at remote monitoring and quick emergency responses.

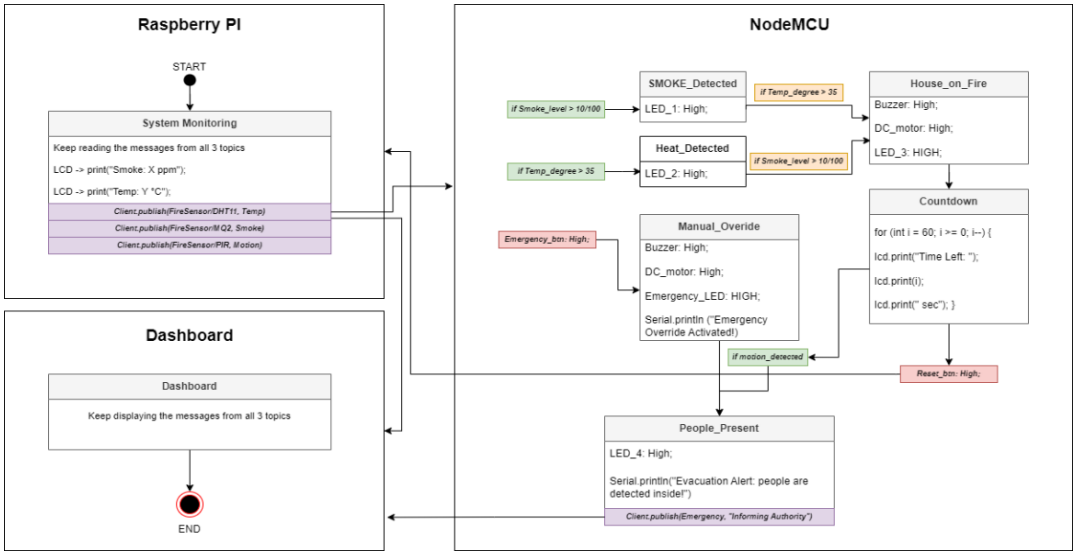

Fig. 2. State Machine Diagram

Hardware Schematics

To assemble the hardware components, we used Fritzing to simulate the configuration and check compatibility. The Raspberry Pi is wired to the DHT11, MQ2 (via an MCP3008 ADC), and PIR motion sensor while properly distributing power and signal wiring. The Node MCU, which is responsible for actuating the system response, is connected to the buzzer, motorized sprinkler, and status LEDs. It also connects to two pushbuttons, the “Emergency pushbutton” and the “Reset pushbutton” for manual control. The setup is detailed in Figure 3.

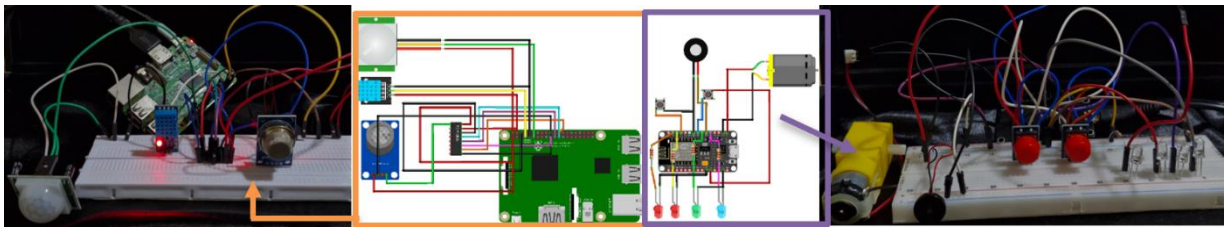

Fig. 3. Hardware integration of the system

Code Development, Dashboard Design, and Implementation

The system functionality is divided into two main parts, each of which contains its own functionality. The first part is basically the sensor module, which is running on the Raspberry Pi. It uses Python and the Paho MQTT library with a DHT11 temperature and humidity sensor. An MQ2 gas sensor and a PIR motion sensor are also connected to it to enhance its functionality. Python scripts on the Raspberry Pi publish sensor data to MQTT topics, while the Node MCU subscribes and triggers actuators based on thresholds. The incoming data is published to the ESP8266, where Arduino libraries are programmed to subscribe to the topics and interpret sensor information. This module automatically manages system states dynamically by triggering hardware alarms such as LEDs, a buzzer, and a DC motor with a

manual override facility by two pushbuttons for users to directly take control in the event of an emergency. The dashboard is built using Flask and web standard technologies in order to give real-time sensor data and emergency status, in order to enhance situational awareness.

Communication and Security Enhancements

The proposed system of this paper implements MQTT communication on port 8883, chosen for its reliable, efficient real-time communication between its nodes. This type of communication ensures the integrity and confidentiality of the system's data. To enhance the security in such a crucial system, the data transmitted will be encrypted using TLS and self-signed certificates to ensure that the central MQTT broker authenticates each other. This mitigates the risks of interception and unauthorized access. Comparing this communication to SMS-based communication suggests a reduced latency, secure, and swift emergency response.

Results and Discussion

Sensor Connectivity & Data Publishing

For the initial test, we aimed to verify that all three sensors were correctly connected to the Raspberry Pi and capable of successfully publishing their data to their designated topics. To evaluate this functionality, each sensor was manually activated.

For the DHT, we increased the ambient temperature using a heat source and observed that the updated temperature values were published to Fire Sensor/Temperature30. Next, the MQ2 sensor has been exposed to a small amount of gas, and the corresponding gas level readings were published to Fire Sensor/Gas31. Finally, we moved within the PIR sensors detection range, and a motion detection message was sent to Fire Sensor/Motion. Using MQTT X, we created an MQTT client that subscribes to the topic FireSensor/#33. As shown in Figure 4, the data from all three sensors were successfully transmitted, each through its respective topic, confirming the successful setup of MQTT communication.

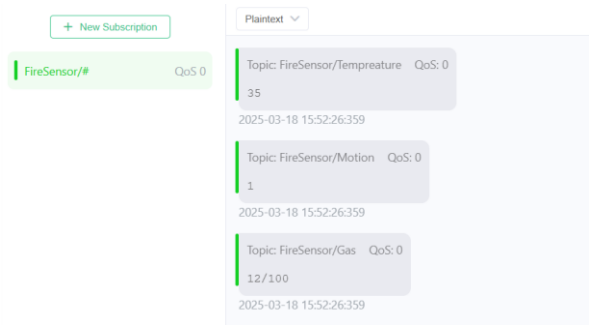

Fig. 4. Sensor Connectivity test

MQTT Subscription & Data Reception

Similarly, we proceeded with connecting the Node MCU as a client and subscribed to the topic Fire Sensor/*. As shown in the serial monitor of the Arduino IDE. The Node MCU continuously received data from the broker. Accordingly, when the temperature and smoke levels exceeded the predefined thresholds, the system correctly triggered the HEAT_DETECTED and SMOKE_DETECTED statuses as shown in Figure 5.

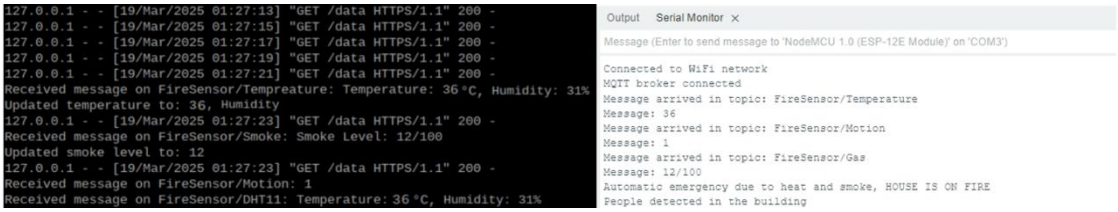

Fig. 5. MQTT Data publishing and reception

Fire Simulation Test

Next, we tested the system functionality by simulating fire conditions (temperature >35°C and smoke >10/100), triggering the HOUSE_ON_FIRE state, activating LEDs, buzzer, and sprinklers (Figure 6). This confirmed the successful logic of our system under fire events.

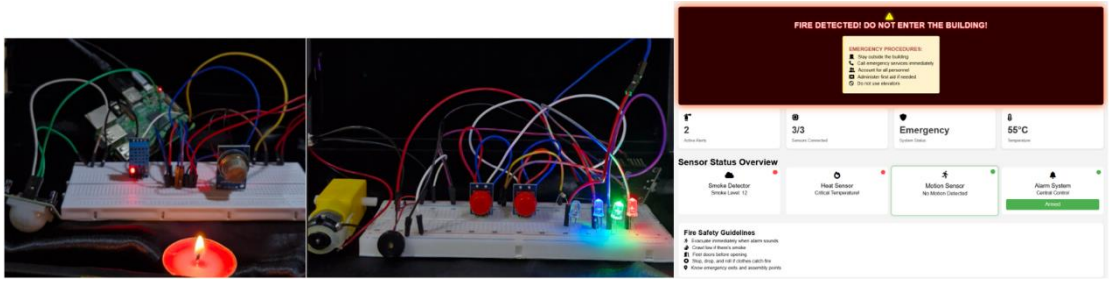

Fig. 6. "House on Fire" scenario test

People Detected Status

Moving on, in the same testing environment as the earlier test case, we simulated motion in front of the PIR sensor. Figure 4 reveals how the 4th LED was triggered. This has confirmed the detection of people within the fire premises. Additionally, the right side of Figure 7 shows the dashboard that will be run by the authorities and how the alert was received from their end.

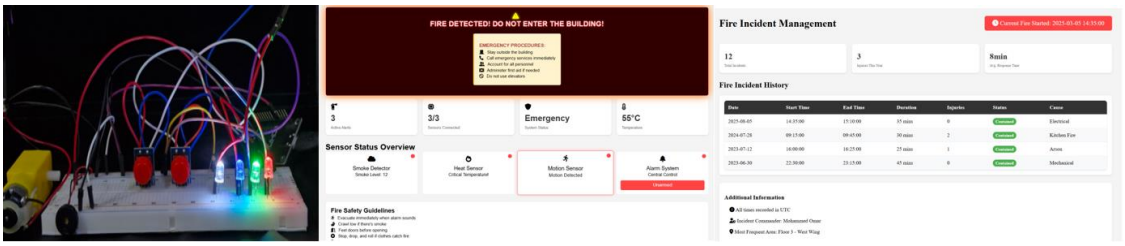

Fig. 7. Motion detected in a house fire scenario test

Emergency and False Alarm Buttons

In order to verify the functionality of the two switches mounted, we initially pressed the emergency button while the system was in SYSTEM_MONITORING mode, and it immediately enabled the MANUAL_OVERRIDE mode as displayed in the Node MCU serial monitor reading, Figure 8. Next, we simulated a fire condition and pressed the false alarm button within a 60-second window (Figure 5). Consequently, the Node MCU switched from HOUSE_ON_FIRE to SYSTEM_MONITORING. This clearly confirms that the system effectively handles manual overrides and can disable emergency alerts for false alarms.

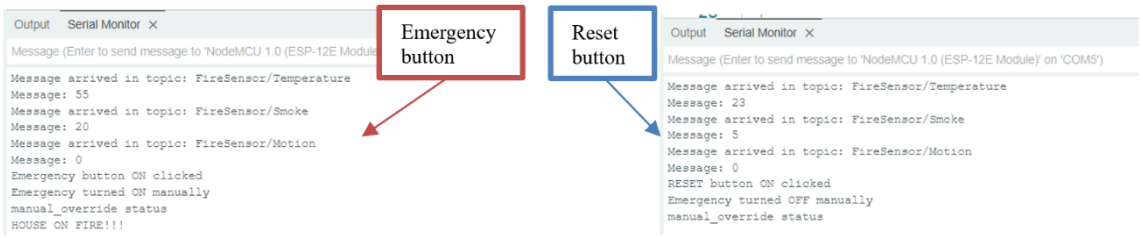

Fig. 8. Manual Reset and Emergency status control test

Discussion

The positive outcomes of these tests signify a strong and secure fire safety system. Proper connection of sensors and data processing ensures quick detection and response to fire accidents. Proper functionality of emergency and false alarm buttons provides the user with critical control, enhancing safety. Additionally, secure communication protocols protect against data breaches, establishing a dependable system for critical fire safety applications.

Conclusion

This multi-sensor IoT fire detection system effectively reduces false alarms via sensor fusion, safeguards data via TLS-encrypted MQTT, and allows for swift response via manual overrides and a real-time dashboard. Initial testing shows correct detection and reliable operation. Future work might involve the addition of a dedicated authority page on the dashboard to record incident information (e.g., start/containment time, injuries, and cause) for improved reporting.

Acknowledgements

The authors would like to express their sincere gratitude to Abu Dhabi University for their support, which made the publication of this research possible. Special thanks are extended to Eng. Gasm El Bary for his support and advice.

Conflict of interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

Lee, W., Kim, J., & Park, S. (2013). Development of building fire safety system with automatic security firm monitoring capability. *Fire Safety Journal*, *58*, 65–73. doi.org

Mahgoub, A., Tarrad, N., Elsherif, R., Al-Ali, A., & Ismail, L. (2019). IoT-based fire alarm system. In *Proceedings of the 2019 World Symposium on Software Engineering Research and Innovation (WorldS4)*. IEEE. doi.org

Mahzan, N. N., Abidin, Z., & Hassan, M. (2018). Design of an Arduino-based home fire alarm system with GSM module. *Journal of Physics: Conference Series*, *1019*(1), Article 012079. doi.org

Muheden, K., Ahmad, S., & Ali, M. (2016). *Design and implementation of the mobile fire alarm system using wireless sensor networks*. IEEE Xplore. ieee.org

Palamar, A., & Palamar, M. (2023). Fire safety monitoring system based on Internet of Things. *CEUR Workshop Proceedings*, *3468*, Article 9. rwth-aachen.de

Patange, P., & Yadav, S. (2015). Design and implementation of automatic fire alarm system based on wireless sensor. *Journal of Emerging Technologies and Innovative Research (JETIR)*, (JETIR1509008). academia.edu

Udurume, M., Benson, C., & Okafor, O. (2025). Developing a fire monitoring system based on MQTT, ESP-NOW, and a REM in industrial environments. *Applied Sciences*, *15*(2),

Bibliographic information of this paper for citing:

Seyam, Sara; Sabouni, Sima & Hassan, Rasha (2026). Secure Multi-Sensor IoT-Based Fire Detection System Using MQTT and TLS Encryption. *Journal of Information Technology Management*, 18 (2), 167-175. https://doi.org/10.22059/jitm.2026.107236