World Journal of Advanced Engineering Technology and Sciences

eISSN: 2582-8266

Cross Ref DOI: 10.30574/wjaets

Journal homepage: https://wjaets.com/

(Research Article)

Design and implementation of an IoT-based fire detection and alert system for real-time hazard monitoring

Omolayo Abegunde [1], Sunday B. Ogunjide [2], Akinola Ibikunle [1] and Oluropo F. Dairo [2,*]

[1] *Department of Computer Engineering, Redeemer's University, Ede 232102, Nigeria.*

[2] *Department of Electrical and Electronic Engineering, Redeemer's University, Ede 232102, Nigeria.*

World Journal of Advanced Engineering Technology and Sciences, 2026, 19(02), 014-027

Publication history: Received on 17 March 2026; revised on 02 May 2026; accepted on 05 May 2026

Article DOI: https://doi.org/10.30574/wjaets.2026.19.2.0251

Abstract

The rapid growth of Internet of Things (IoT) technology has led to the development of innovative solutions for real-time hazard monitoring, particularly in fire detection and alert systems. Traditional fire detection methods often fail to provide timely alerts, which put lives and property at risk. To address this issue, this study presents an IoT-based fire detection and alert system, leveraging wireless sensor networks, and cloud computing to detect fire and alert authorities in real-time. The system employs smoke detectors, temperature sensors, and gas sensors to monitor environmental conditions and detect potential fire hazards. Data is transmitted to a cloud platform, where it is been analyzed to identify patterns and anomalies indicative of a fire. Upon detection, alerts are sent to emergency responders and stakeholders through short message service, email, and mobile app. Experimental results demonstrate high accuracy rate and a response time of approximately three seconds, surpassing traditional methods and several past IoT solutions. The integration of IoT technology and cloud computing in this study enables real time monitoring, predictive analytics, and automated response, making this system an essential tool for fire safety management in residential, commercial and industrial settings. Ultimately, this study contributes to the realization of United Nations Sustainable Development Goal (SDG) 9, which focuses on Industry, Innovation, and Infrastructure, through innovative smart building safety measures. Simultaneously, the implementation of real-time hazard monitoring has a twofold effect on achieving Sustainable Development Goal 11, enhancing safety and sustainability within educational environments.

Keywords: IoT-Based Fire Detection; Real-Time Monitoring; Microcontroller; Temperature Sensor; Cloud Integration

1. Introduction

Fire poses a significant threat to safety and property, and traditional detection methods often fail to provide timely alerts. Conventional methods, such as standalone smoke alarms and manual bells, can only react once a fire is already underway [1][2][3]. Such systems often experience delays in providing reports and do not offer real-time status updates to safety personnel, resulting in limited early warning [4–7]. In practice, this means valuable minutes can be lost before authorities know where a blaze has begun. Additionally, traditional alarms are not effective in large environments and does not support centralize monitoring. Consequently, fires in remote buildings may go undetected until it is too late [8][9][10]. To enhance safety, it is essential to adopt new methodologies that facilitate continuous, networked monitoring and immediate alerts. This approach is preferable to relying solely on periodic checks or local alarms [11][12]. The recent progress in the Internet of Things (IoT) has created fresh opportunities for fire hazard monitoring. Data from environmental sensors can be read by modern microcontroller platforms, such as ESP32 and Node MCU, which can also communicate wirelessly [13–16]. In practice, these sensors are connected to an ESP32 and Node MCU board, and data is uploaded to the cloud through a network shield [17][18][19]. This data can be retrieved by a mobile app (Android or iOS), which can then automatically alert the building manager or emergency contacts when a sensor

* Corresponding author: Oluropo F. Dairo.

reading is critical. This IoT setup effectively turns passive alarms into an online monitoring system, allowing users to monitor their alarms remotely and keep their homes safe [20].

Several recent studies illustrate the power of IoT-based fire detection. A novel hybrid dynamic best model selection (HDBMS) machine learning-based algorithm for internet of things (IoT)-enabled fire detection in buildings was proposed by Mujeeb et al. [21]. Higher accuracy and adaptability across diverse fire scenarios was provided by the result, which outperforms traditional static models. Chuanshen et al. [22] developed a wireless intelligent thermoelectric response fire warning system with adjustable thresholds by integrating Internet of Things (IoT) technology. The study revealed that voltage signals can be monitored in real time through OLED displays and mobile devices. The adjustable fire warning threshold voltage demonstrates flexibility in adapting to diverse application scenarios and showcases the great potential for next-generation, intelligent, IoT-based fire warning applications. Khushwant et al. [23] analyzed a smart Internet of Things (IoT) system focusing on planning and evaluating a wireless sensor network that can detect house fires in their earliest stages. The study addressed the crucial issue of providing rapid fire alerts by incorporating a variety of sensors into the wireless sensor network. The study concluded that IoT-based fire detection systems have the potential to offer numerous benefits, including real-time monitoring, remote management, and data-driven insights. A fire detection and anti-fire security (FDAS) system for smart agriculture using the IoT and embedded system was proposed by Abdennabi et al. [24]. This system used an embedded system and sensors to measure the amount of smoke in the air and the size of the fire. The data obtained from the sensors was sent to the ThingSpeak platform via the machine-to-machine Message Queuing Telemetry Transport (MQTT) protocol for display and analysis. Performance results showed the proposed scheme's accurate fire detection and anti-fire system performance, which improves food security and sustainability in agriculture. A novel forest fire detection model utilizing audio classification and machine learning was proposed by Kemahyanto et al. [25]. The development of an audio-based pipeline was initiated using real-world environmental sound recordings. The findings indicate that a combination of acoustic sensing and machine learning provides an effective, cost-efficient, and real-time solution for forest fire monitoring in complex, ever-changing environments. Yuvaraj et al. [26] developed a compact, smart fire safety system that integrates real-time, deep-learning-based detection with automated extinguishing. Their results demonstrate the practicality and novelty of an AI-driven detection and suppression framework. This framework offers a scalable, modular solution for smart buildings, industrial plants, and other safety-critical environments. A novel airborne UAV-based IoT (UIoT) system for wildfire sensing, detection, and extinguishing was proposed Montaser et al. [27]. Their results suggest that the UAV-IoT-based fire classification system provides a fast, reliable solution for recognizing and extinguishing wildfires while minimizing power consumption and maximizing battery lifespan. Lien et al. [28] designed a hybrid kitchen safety guarding framework using embedded devices and onboard sensors to detect abnormal events and block gas sources in time through the Internet of Things. Results show that the developed framework can precisely recognize stove fire intensity and detect earthquake levels for kitchen safety guarding.

Various aspects of IoT-based fire detection systems have been explored in previous studies, including sensor technologies, communication protocols, and machine learning algorithms. Nonetheless, in light of the escalating frequency and severity of fire incidents on a global scale, it is imperative to further explore the potential of Internet of Things (IoT)-based fire detection and alert systems. The goal of this study is to design an IoT-enabled fire detection system that acquires and communicates data in real time, integrates cloud-based data analysis and notification systems, and uses tests to evaluate the systems reliability. This study contributes to the realization of United Nations Sustainable Development Goal (SDG) 9 through innovative smart building safety measures.

2. Methodology

2.1. Research approach

The adoption of an applied experimental design characterized this study. The implementation was meticulously structured in phases: system design, hardware configuration, software configuration, and testing. The architecture of the IoT-based fire detection system was designed to integrate environmental sensors, a microcontroller, alert actuators, and cloud connectivity. The ESP32 microcontroller, which has built-in Wi-Fi, served as the central processing unit and interfaced with multiple sensors. The faculty of engineering building at Redeemer's University was the site of the development and testing of the hardware and software components. The purpose of this was to evaluate their performance in real conditions. The system was developed as a dependable and efficient method for detecting potential fire hazards by utilizing the Internet of Things (IoT) technology to identify these dangers in real time. This innovative approach allows for continuous monitoring and immediate response to any signs of fire risk. Fig. 1 illustrates the key components of the framework, highlighting how each part contributes to the overall functionality and effectiveness of the system.

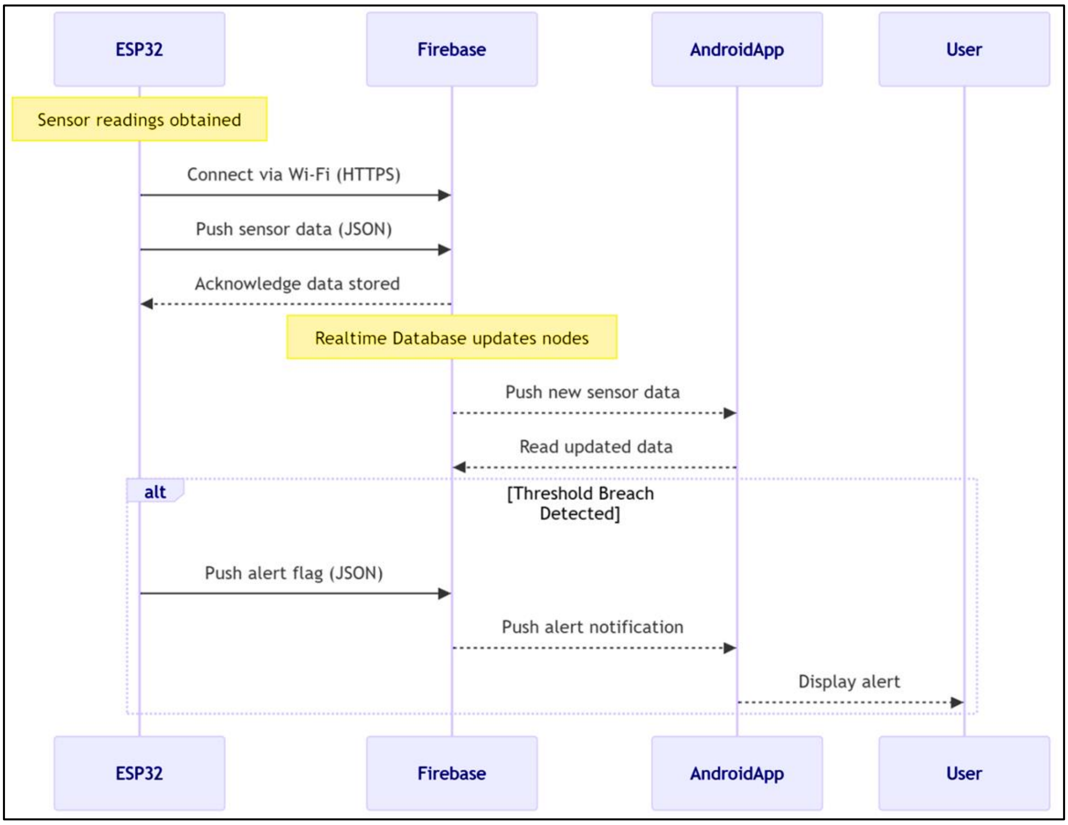

Figure 1 System architecture

2.2. Hardware design

The hardware setup of the project comprised four essential elements, with the fire detection and notification system being of paramount importance. The ESP32 microcontroller was utilized as the processing unit. It is a 32-bit MCU with built-in Wi-Fi and Bluetooth. The microcontroller was used to collect sensor information. It processed the information. Finally, it transmits the information to the cloud via firebase as shown in the supplementary material. The DHT22 sensor measured the humidity and temperature levels of the environment and provided real-time readings that could indicate potential early fire conditions. The MQ-2 gas/smoke sensor played a crucial role in detecting combustible gases and smoke particles in the air, which are common signs of potential fire hazards. A buzzer was also added to provide an immediate local alert whenever sensor readings exceeded predetermined safety thresholds, notifying individuals nearby of any danger. The hardware components worked together to create a functional, responsive, and scalable IoT-based fire alarm system. The sensors were connected to the ESP32 GPIO pins. The DHT22 provides calibrated digital temperature and humidity readings. The MQ-2, on the other hand, outputs a signal proportional to the concentration of combustible gases. These inputs were connected to the digital and analog input pins of the ESP32, respectively. The ESP32 was powered by a 5V USB supply and shared a common ground with the sensors. Pull-up resistors were used as needed (e.g., the DHT22 data line) to ensure reliable communication. A breadboard prototype was assembled to arrange the components and facilitate connections. The LED (visual) and buzzer (audio) provided local alerts when hazardous conditions were detected. Fig. 2 shows the block diagram of the hardware integration. This integration goes from sensors to ESP32 to indicators. The DHT22 was selected for its ±0.5°C temperature accuracy and extensive humidity range, while the MQ-2 gas sensor was employed for its high sensitivity to combustible gases (LPG, propane, smoke). Each sensor continuously monitors its parameter and relays data to the ESP32 for processing. The system was tested using known values to calibrate the readings, and the alert hardware (LED/buzzer) was verified by manually triggering threshold breaches.

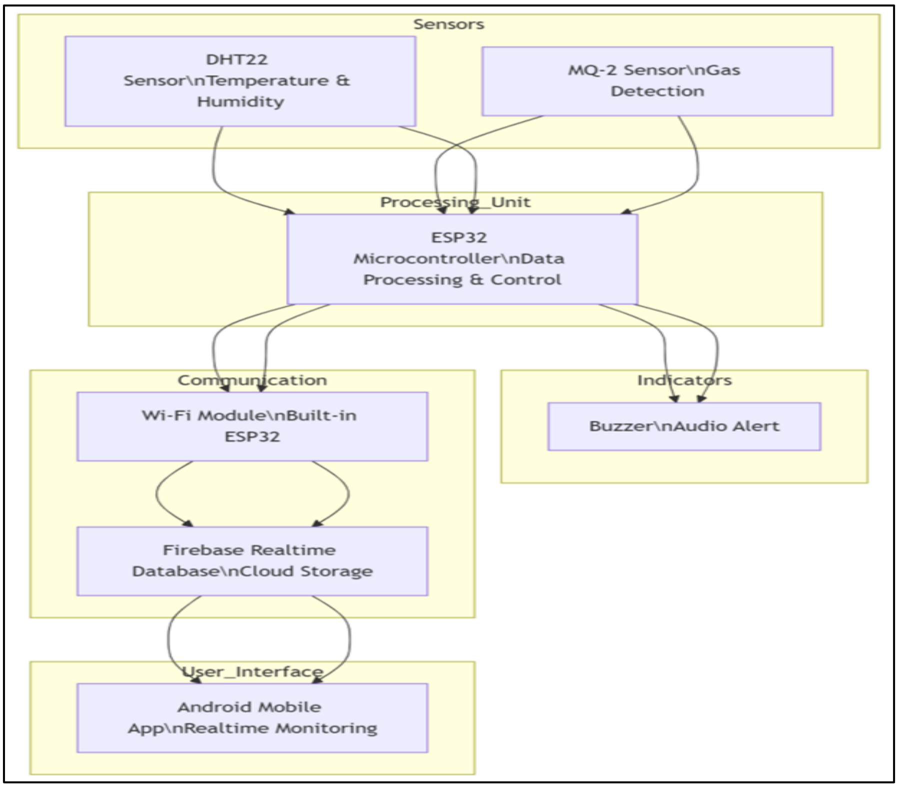

Figure 2 Hardware integration.

The ESP32 is configured to interface with the mobile application utilizing firebase. The communication architecture between the ESP32 microcontroller, the firebase real-time database, and the Android application follows a client-server model facilitated by cloud services as depicted in the supplementary material. The ESP32 acts as a device that acquires and publishes data, while Firebase serves as the cloud-based intermediary platform that is responsible for data storage, synchronization, and real-time updates. The Android app is the client interface that allows users to check the system's status and receive updates.

2.3. Communication flow

Environmental data, including temperature, humidity, and gas concentration, is continuously read from the DHT22 and MQ-2 sensors every two seconds by the ESP32. Additionally, the ESP32 establishes a Wi-Fi connection and authenticates with the firebase real-time database using firebase SDK libraries (FirebaseESP32 for the Arduino IDE). When new sensor data is collected, the ESP32 formats the readings as JSON objects and sends them to Firebase via HTTPS requests. Firebase stores the sensor readings under designated database nodes that are structured by parameters such as temperature, humidity, and gas level. The database supports real-time synchronization, ensuring that any updates are immediately reflected across all connected clients. The Android application is connected to the same firebase real-time database instance, which allows for real-time data sharing and synchronization across devices. Using the firebase SDK for android, the application listens for changes in database nodes related to sensor readings. Whenever the ESP32 uploads new data, firebase instantly pushes these updates to the application, enabling real-time monitoring. If the sensor readings exceed predefined thresholds, the ESP32 triggers a local alert using a buzzer and LED and writes a specific alert flag to firebase. Upon detecting this change, the android application displays notifications and alerts to the user. Fig. 3 shows the system circuit diagram.

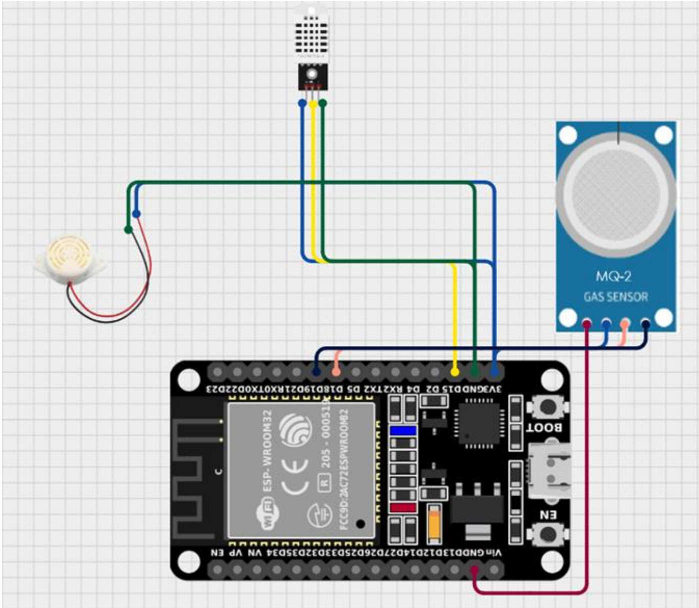

Figure 3 System circuit design.

2.4. Software design and implementation

This section focuses on the software components and processes that enable real-time fire detection and monitoring. It explains how the project uses flutter for developing the mobile application and firebase services for the backend, including the real-time database, cloud messaging and authentication. This architecture enables seamless real-time communication between the ESP32 microcontroller, the firebase cloud, and the flutter-based android application.

The ESP32 was programmed using the Arduino IDE, with C/C++ libraries being leveraged for sensor interfacing and network communication. The essential libraries encompassed the DHT library, which facilitates the reading of the DHT22, and an MQ-2 sensor library, which enables the interpretation of gas readings. The software routine runs as a continuous loop. Upon startup, it initializes the sensors and Wi-Fi, then reads sensor values every two seconds. Each reading is compared against predefined safety thresholds: temperature greater than 35 °C and gas reading above its safe limit. When a threshold is exceeded, the code immediately triggers LED and buzzer alerts. The network communication was configured so that the ESP32 would connect to the local Wi-Fi network and authenticate with the Firebase real-time database. A firebase project was created and the necessary API keys and rules were set. The program uses Wi-Fi libraries to establish the connection. Then, sensor data, temperature, humidity, and gas are transmitted to Firebase in JSON format. Firebase's data structure includes nodes for each parameter, enabling real-time synchronization and remote monitoring. The firebase real-time database automatically updates client devices when new data arrives, ensuring live updates. To simplify database writes, the Firebase ESP32 library was used for cloud integration. The data format in Firebase NoSQL DB is structured. As is common in Firebase text nodes, all sensor values were stored as strings. A simple database table was created to record the sensor readings. Tables 1 and 2 depict the Firebase table named Sensor Readings and the uses of the Firebase and Flutter libraries, respectively. Each sensor data entry contains three fields. Storing the values as strings in Firebase enables easy JSON handling. This design ensures that each parameter is recorded and accessible for historical analysis.

Table 1 Database schema

Field	Datatype	Description
Temperature	String	Temperature in °C (as text)
Humidity	String	Relative humidity in % (as text)
Mq2	String	MQ-2 gas sensor value (as text)

Table 2 Firebase, flutter libraries and their uses.

WiFi.h	**Enables ESP32 to connect to Wi-Fi networks**
FirebaseESP32.h	Facilitates communication between ESP32 and firebase real-time Database
DHT.h	Used for interfacing with the DHT22 temperature and humidity sensor
Adafruit_Sensor.h	Provides sensor abstraction for DHT integration
ArduinoJson.h	Helps in formatting data as JSON objects for Firebase
firebase_core	Initializes and connects flutter to firebase
firebase_database	Connects flutter to firebase real-time database
firebase_messaging	Enables firebase cloud messaging for push notifications
flutter_local_notifications	Displays local notifications on the device
provider	State management within the flutter app
intl	It is used for date and time formatting
http	Used for making HTTP requests.

A flowchart was used to depict the system's operation, as illustrated in Fig. 4. The sensor data acquisition process starts with the ESP32 initializing all components and entering a loop that reads each sensor. If any reading exceeds its limit, the ESP32 activates the local buzzer and LED. In parallel, the data are uploaded to Firebase, and any triggers are sent.

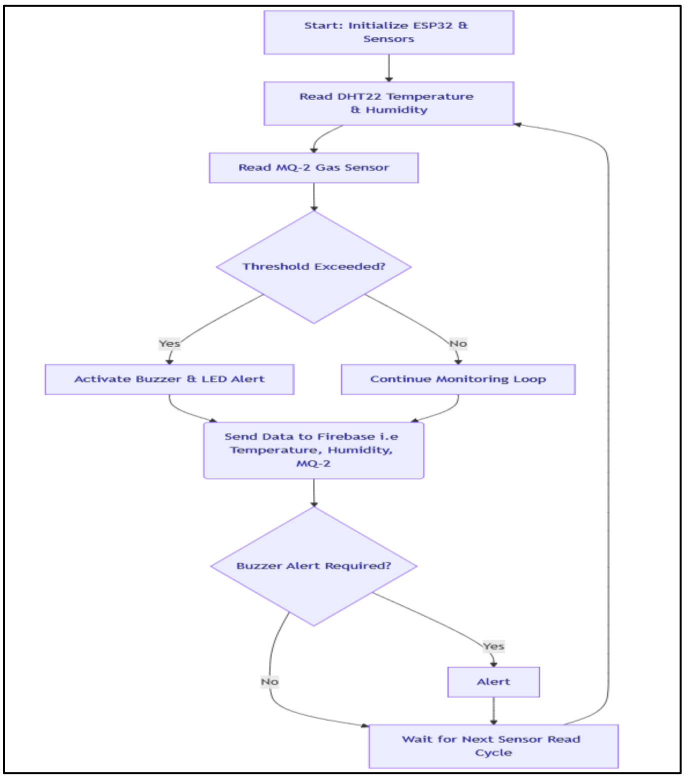

Figure 4 System flowchart

The implementation of these methodology steps was aimed at developing a reliable, real-time fire hazard monitoring system. The development of the hardware and software components was concurrent, with individual testing and subsequent integration. All design steps drew upon established IoT practices to ensure robust functionality, with each step contributing to the overall effectiveness of the system.

2.5. Mobile application

The mobile application is an essential part of the IoT-based fire detection and alert system, giving users real-time access to environmental sensor data and immediate alerts of possible fire hazards. It was developed using Flutter, a modern cross-platform framework based on the Dart programming language. To ensure seamless real-time communication between the hardware and the end-user, Firebase suites of services were integrated, including Firebase Real-Time Database for cloud storage and synchronization and Firebase Cloud Messaging (FCM) for push notifications. Clean architecture principles are adhered to by the mobile application, ensuring clear separation of concerns between the presentation, domain and data layers. This enhances maintainability, scalability and testability. The architecture of the application is depicted in Fig. 5.

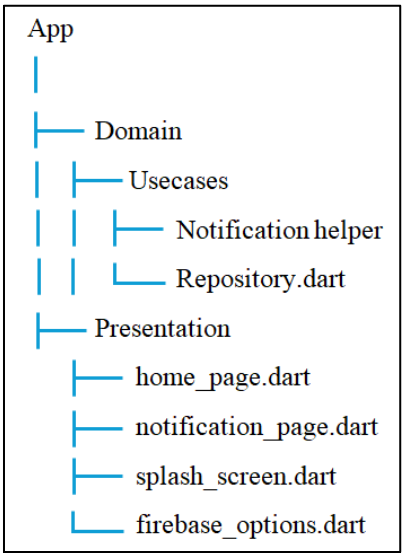

Figure 5 Architecture of the application.

2.6. Development tools

The real-time fire detection and alarm system, which facilitates hazardous area monitoring, relies on a variety of tools and platforms that enable rapid prototyping, efficient coding, real-time testing, and deployment. Each tool was carefully selected based on its compatibility with project goals, particularly cross-platform mobile development, cloud integration, and embedded system support. The Flutter SDK provides the framework for building cross-platform, responsive mobile applications. Its cross-platform capabilities allow a single codebase to be deployed on both Android and iOS, reducing development time while ensuring a consistent user experience. The Arduino IDE is a tool used to write, compile, and upload firmware code to the ESP32 microcontroller. It offers debugging tools, emulators, and code analysis to boost productivity. The cloud backend that stores and synchronizes sensor readings and system alerts in real time is firebase's real-time database. It enables the mobile app to access live data instantly. The firebase console is a web interface that allows developers to manage and interact with their Firebase projects. The console provides access to various firebase services and tools, enabling developers to build, run, and monitor their mobile and web applications. Mermaid.js is a JavaScript library that allows users to create diagrams and visualizations from text and code using a Markdown-inspired syntax. It was used to create flowcharts, sequence diagrams, and system architecture diagrams for the report. Git/GitHub provides version control for the flutter application source code and the Arduino firmware. Lastly, the Dart programming language was used to implement the application's business logic, UI components, and state management.

Boundary conditions for fire alert trigger

The conditions that trigger a fire alert are clearly outlined and illustrated in Equation 1.

$$(T > 30) \wedge (H < 60) \wedge (G > 70) \quad (1)$$

where, T is the temperature in °C (from DHT22), H denotes humidity in % (from DHT22) and G represents gas concentration (analog value from MQ2).

Piecewise-style conditional expression was also employed in this study (evaluates to 1 if true, 0 if false), as illustrated in Eq. 2.

$$F(T,\ H,\ G) = \{1,\ \text{if}\ \ T > 30;\ \ H < 60;\ G > 70)\ \text{otherwise},\ 0 \quad (2)$$

The boundary conditions and hazardous events are highlighted in Table 3. For the function to return 1 (trigger an alert), all three variables must be true simultaneously.

Table 3 Hazardous events.

Variables	Hazardous event
Temperature	Exceeding 30 °C
Humidity	Falling below 60%
Gas level	Exceeding 70 ppm

2.7. Testing and validation

The performance, accuracy, and efficiency of the IoT-based fire detection and alert system are evaluated using the data analysis techniques outlined in this section. The analysis uses quantitative metrics. A comparison was made between the readings from the DHT22 sensor and those from the calibrated thermometer and hygrometer, with the DHT22 sensor being placed in a controlled environment for this purpose. The MQ-2 gas detector was also subjected to regulated gas supplies, and the measurement was evaluated in comparison to the established ppm values for smoke presence that are known to be present. A rising temperature was simulated manually using a heat source. Humidity was reduced using heat, and gas concentration was increased using light smoke and a gas source until the thresholds were exceeded to test the sensors reactions. Breaches of threshold are detected accurately by the system, which also triggers local alerts (in the form of a buzzer) and marks the Alert Status as "Hazard Detected" in Firebase. The ESP32 was fed values that hovered around the threshold limits to ensure an accurate evaluation and to avoid false triggers. The system only activates alerts when the threshold is breached and not for borderline or fluctuating values. Additionally, the live data uploads from ESP32 to firebase were closely monitored as the mobile application was observed. Environmental conditions were gradually changed to ascertain how quickly the updated data appears on Firebase and reflects on the app. It was observed that firebase reflected updated sensor readings within one to three seconds, and the mobile app immediately mirrored these changes without requiring a manual refresh. We simulated a temporary Wi-Fi disconnection for the ESP32 to see if it would resume uploading data automatically upon reconnection without losing any data. The ESP32 was found to reconnect to Wi-Fi and continue uploading data to firebase seamlessly. Threshold breaches were triggered manually to observe the ESP32 physical response. Upon detection of hazardous conditions, the buzzer sounded, and the LED illuminate within 2 seconds. Lastly, hazard condition was triggered to confirm that AlertStatus = "Hazard Detected" is written to firebase. Firebase cloud messaging sends a push notification to the mobile device and the user receives clear and immediate notification of the hazard on the mobile device.

3. Results and discussion

This section presents the results of the Internet of Things (IoT)-based fire detection and alarm system. The system was evaluated under normal and hazardous conditions. During the evaluation period, the system sent both a local alert through the buzzer and a cloud alert through the mobile application notification system whenever the threshold values were breached to inform the relevant parties. The system receives sensor data, which the ESP32 uploads to firebase. The data is then sent to the mobile application, updating the previous record within two seconds.

3.1. System demonstration

The fire detection system is deployed in a custom-built enclosure that is designed for portability and environmental protection. The system hardware comprises an ESP32 integrated into the inner surface of the enclosure lid, with meticulously aligned apertures for the DHT22 and MQ2 sensors, respectively. The system packaging is shown in two views: the inner view in Fig. 6 and the outer view in Fig. 7.

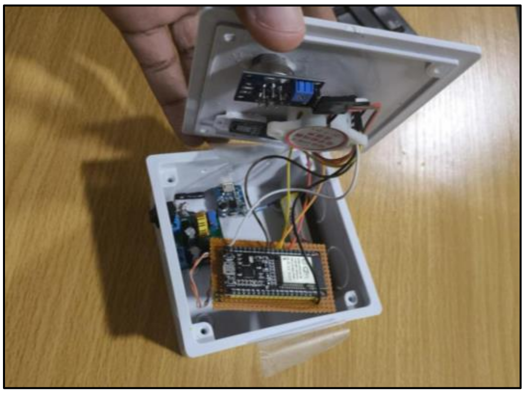

Figure 6 Inner view of the project**.**

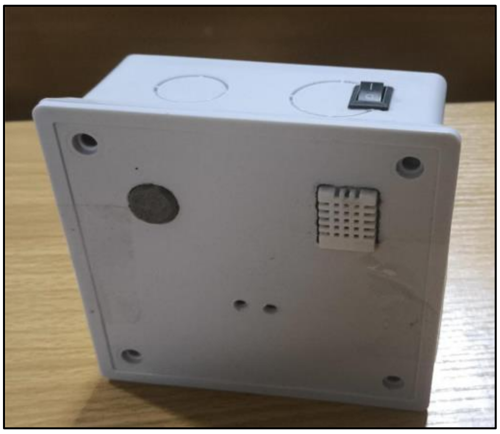

Figure 7 Outer view of the project.

The enclosure internally houses three key hardware components. One is a rechargeable Ultra Fire 3.7-volt lithium battery. It serves as the primary power source. Another is a DC-to-DC buck converter (LM2596). It regulates voltage. This converter supplies stable power to the ESP32-CAM. The enclosure also contains a 134N3P USB charging module. This module enables safe battery recharging through micro-USB. To ensure resilience against vibration and minor impact, all internal components were mounted securely with adhesive and screws. The cabling between the units was kept to a minimum and tidily arranged to maintain airflow and make future upkeep easier. During testing, the device operated wirelessly and was powered solely by its internal battery. Its compact size and flexible design enabled discreet mounting on walls, ceilings, or desktops as needed.

3.2. Mobile application

The mobile application is designed to interface directly with the ESP32, enabling continuous monitoring of the system's sensor readings. When the app is launched, the ESP32 connected to the mobile hotspot retrieves the sensor readings and sends them to the firebase library. The app then retrieves the data from firebase and updates in real time. The mobile application has two states. These states represent two different situations. The first state indicates when the system is safe. The second state indicates when the threshold values are exceeded. These situations are illustrated in Figs. 8 and 9.

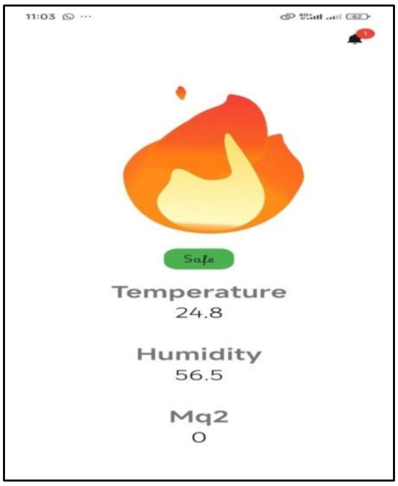

Figure 8 System safe state.

Figure 9 System hazard state.

3.3. Overview of test setups

A series of systematic tests were designed and implemented to evaluate the performance, accuracy, and reliability of the IoT-based fire detection and alert system for real-time hazard monitoring. These tests verified that the system adhered to its functional requirements regarding sensor accuracy, threshold testing, data synchronization, and alert mechanisms. The test scenarios and environments were selected to replicate normal conditions. They were also selected to replicate hazardous conditions. These conditions were selected for realistic operating environments. The DHT22 temperature, humidity sensor and the MQ2 were tested, and there was a match between the values of the thermometer and hygrometer and the values of the DHT22 and MQ2 with low amount of variance. The system sensors were tested using a lighter as a heat source to bring them to their threshold values. The fire alarm buzzer sounded and a notification was sent by the mobile application (Scorch). When the fire alarm sensor values approached the threshold values, nothing happened until the threshold was exceeded, ensuring that the system does not produce false alarms. A real-time synchronization test showed that the mobile application updated from the sensor to firebase and to the mobile application within two seconds, demonstrating that the system meets the requirements for real-time hazard monitoring. The Wi-Fi connection of the ESP32 was deliberately disconnected and then reconnected in order to assess its reaction to this alteration in connectivity. When the system reconnected, it updated its sensor data without any issues. The system local alert is a buzzer that is activated when threshold values are exceeded and remains on for 10 seconds before being deactivated. In the event that, subsequent to the passage of time, the sensor data continues to satisfy the threshold value condition, the buzzer will be triggered once more until such a time as it no longer meets the requisite condition prior to the termination of the loop. The mobile application using Flutter local notification is notified via Firebase notification when the sensor data meets the threshold value condition. This notification is shown to the user, and when the user taps on it, the application opens on the mobile phone. Table 4 presents the results obtained during testing of the system both in normal and hazardous conditions.

Table 4 Obtained results at normal and hazardous cases.

Time (Minutes)	Temperature (°C)	Humidity (%)	Gas (ppm)	Condition	Alert Status
0	27	65	45	Normal	No alert
5	28	63	50	Normal	No alert
10	29	61	55	Normal	No alert
15	31	58	72	Hazard	Local + Remote Alert
20	33	55	80	Hazard	Local + Remote Alert
25	35	52	90	Hazard	Local + Remote Alert
30	28	63	50	Normal	No alert

3.4. Comparative analysis with previous study

The performance, portability, and response time of the proposed Internet of Things (IoT)-based fire detection and alarm system were compared with those of existing studies. A comparison was made between the present study and the works of Kabir et al. [29], Irakomeye et al. [30] and Pan et al. [31]. The focus of Kabir et al [29] was on device interconnection and data sharing through Zigbee, Wi-Fi, and MQTT. They focused on centralized synchronization while neglecting to incorporate real-time external communication and local onsite alert systems. DHT22 and MQ-2 sensors were utilized by Irakomeye et al. [30] and IoT cloud systems and early target fire detection were leveraged. However, the study focused solely on Tanzanian industrial environments. The project is deficient in the implementation of backup procedures in the event of a network failure, and the notification system is exclusively cloud-based, utilizing a mobile application as the sole means of communication. Cloud platforms were leveraged by Pan et al [31] for centralized monitoring and multi-sensor data collection. The use of complex AI models (ANNs) for predicting fire risk necessitates substantial computational power and is particularly well-suited for large commercial buildings. The project prioritized enhancing accuracy through the use of AI, while reducing the need for immediate action. Table 5 provides a detailed comparison of past studies on fire detection systems.

Table 5 Comparative analysis of previous studies on fire detection systems.

Study	Sensor	Alerts	Comm. Tech	Cloud/AI	Limitations	Improvements
Kabir et al. [29]	Many	None	Zigbee, Wifi	MQTT	No external real time alerts	Adds fire-base real-time notification.
Irakomeye et al. [30]	MQ-2, DHT22	Mobile only	IoT/WSN	Cloud	Relies solely on mobile alerts	Adds redundant local (buzzer) alerts
Pan et al. [31]	Multi-sensor	Cloud only	IoT, AI-Cloud	AI/ANN	Too complex for small applications	Simple, scalable, suitable for universities

It is noteworthy that the present study achieves high overall accuracy, compared to the existing studies in terms of latency, offering instantaneous readings updates less than 1000 milliseconds and portability utilizing ESP32 hardware.

Abbreviations

IoT	Internet of Things	JSON	Javascript object notation
SMS	Short message service	HTTPS	Hypertext transfer protocol secure
MCU	Multicontroller unit	SDK	Software development kit
Wi-Fi	Wireless fidelity	API	Application programming interface

ESP	Espressif systems	NoSQL	Not only structured query language
HDBMS	Hybrid dynamic best model selection	DB	Database
iOS	iPhone operating system	FCM	Firebase cloud messaging
OLED	Organic light emitting diode	DC	Direct current
FDAS	Fire detection and anti-fire security	LM	Lunar module
MQTT	Message queuing telemetry transport	AI	Artificial intelligence
UAV	Unmanned aerial vehicle	ANNs	Artificial neural networks
SDG	Sustainable development goal	GSM	Global system for mobile
DHT	Digital humidity and temperature sensor	WSN	Wireless sensor network
MQ-2	Gas sensor	IDE	Integrated development environment
USB	Universal serial bus	LED	Light emitting diode
LPG	Liquefied petroleum gas		

4. Conclusions

The design and implementation of an IoT-based fire detection and alert system for real-time hazard monitoring is a significant advancement in fire safety technology. This system leverages interconnected devices and sensors to detect, analyze, and respond to fire incidents in real-time, providing enhanced accuracy, faster response times, and remote monitoring capabilities. The following conclusions are drawn based on the findings of this study:

- The IoT-based fire detection system in this study utilized wireless sensor networks and cloud computing for real-time hazard monitoring.
- The integration of multiple sensors, such as smoke, temperature, and gas detectors, enables the effective detection of potential fire hazards.
- The system facilitates real-time monitoring, predictive analytics, and automated responses, making it well-suited for fire safety management.
- Emergency responders and stakeholders receive alerts via multiple channels (SMS, email, and mobile app) to take swift action.
- The system boasts an impressive high accuracy rate and a swift response time of around three seconds, surpassing traditional methods.

Ultimately, this study contributes to the realization of United Nations Sustainable Development Goal (SDG) 9, which focuses on Industry, Innovation, and Infrastructure, through innovative smart building safety measures. Simultaneously, the implementation of real-time hazard monitoring has a twofold effect on achieving Sustainable Development Goal 11 (Sustainable Cities and Communities), enhancing safety and sustainability within educational environments. Our future work will consider incorporating additional sensors, including photoelectric smoke detectors, flame sensors, particulate matter monitors to improve detection accuracy across diverse fire-precursor signatures, integrate a GSM module as a fallback for mobile notifications through SMS when Wi-Fi is unavailable.

Compliance with ethical standards

Acknowledgments

This study was supported by the Faculty of Engineering, Redeemer's University, Ede, Osun State, Nigeria.

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Han R, Li J, Liu Y and Liu H.(2025). Forest fire object detection based on multi-task model and extreme weather simulation algorithm. Eng Appl Artif Intell 2026;163 : 113099. https://doi.org/10.1016/j.engappai.2025.113099.

[2] Wen Y, Qiao Y, Lam CC, Brilakis I, Lee S and Wong MO. (2026). Semantic BIM enrichment for firefighting assets: Fire-ART dataset and panoramic image-based 3D reconstruction. ISPRS J Photogramm Remote Sens 2026;231:679–703. https://doi.org//10.1016/j.isprsjprs.2025.11.015.

[3] Ali MM, Tahtali M and Ghodrat M. (2025). Real-time CCTV-based deep learning for early detection of lithium-ion battery fires. J Power Sources 2025;659:238452. https://doi.org/10.1016/j.jpowsour.2025.238452.

[4] Chen WD, Chen YC, Yu KM, Lee CL, Wu ML, Feng MZ, Chiu WF, Lei MY and Wang SC. (2026). Intelligent early-stage fire detection system for enhanced safety in large warehouse environments. Fire Saf J 2026;160:104818. https://doi.org/10.1016/j.firesaf.2025.104618.

[5] Yuan L, Zhou P, Wang Z and Liu J. (2025). A high-performance model for accurate fire detection in aerial imagery. Neural Networks 2026;197:108414. https://doi.org/10.1016/j.neunet.2025.108414.

[6] Zhou Z, Zhao H, Li X, Liu Y, Jiang T, Jin J and Guo Y. (2025). High-performance fire detection framework based on feature enhancement and multimodal fusion. J Saf Sci Resil 2026;7:100212. https://doi.org/10.1016/j.jnlssr.2025.03.004.

[7] Liu W, He R, Liu Y, Jiang S, Fang J, Wu L and Lu F. (2025). Research on fire assessment during robot-assisted downhole rescue based on the fusion of human and artificial intelligence. Technol Forecast Soc Change 2026;223124458. https://doi.org/10.1016/j.techfore.2025.124458.

[8] Zhang Y, Chen L, Shi Y and Wang H. (2025). Dynamic modeling and real-time risk assessment of tunnel fires using pressure-state-response (PSR)-enhanced bayesian network. Results Eng 2025;28:107717. https://doi.org/doi.org/10.1016/j.rineng.2025.107716.

[9] Ahn Y, Kim S, Han KW, Lee KB, Yang HS, Cho K and Ryu JH. (2025). Adaptive progressive learning for minimizing false alarms in fire detection with limited Re-training data. Eng Appl Artif Intell 2026;166:113611. https://doi.org/10.1016/j.engappai.2025.113611.

[10] Cao X, Wu K, Geng X and Guan Q. (2024). Field detection of indoor fire threat situation based on LSTM-Kriging network. J Build Eng 2024;84:108686. https://doi.org/10.1016/j.jobe.2024.108686.

[11] Wong MO and Lee S. (2025). Quantitative analysis of the effectiveness of real-time indoor navigation and information sharing for collaborative fire response. Dev Built Environ 2025;23:100690. https://doi.org/10.1016/j.dibe.2025.100690.

[12] Wesabi FNA, Almansour H, Iskandar HG and Yaseen I. (2025). Enhanced Fire Detection System for Blind and Visually Challenged People Using Artificial Intelligence with Deep Convolutional Neural Networks. Comput Mater Contin 2025;85:5765–87. https://doi.org/10.32604/cmc.2025.067571.

[13] Morchid A, Et-taibi B, Oughannou Z, Alami RE, Qjidaa H, Jamil MO, Boufounas EM and Abid MR. (2024). IoT-enabled smart agriculture for improving water management: A smart irrigation control using embedded systems and Server-Sent Events. Sci African 2025;27:e02527. https://doi.org/10.1016/j.sciaf.2024.e02527.

[14] Hu J, Cheng F, Liu M, Xu X and Li X. (2025). MicroFallNet: A lightweight model for real-time fall detection on smart wristbands. Pervasive Mob Comput 2025;109:102046. https://doi.org/10.1016/j.pmcj.2025.102046.

[15] Nkinyam CM, Ujah CO, Asadu CO, Anyaka B and Olubambi PA. (2025). Development of a low-cost monitoring device for solar electric (PV) system using internet of things (IoT). Results Eng 2025;28:107324. https://doi.org/10.1016/j.rineng.2025.107324.

[16] Narayana TL, Venkatesh C, Kiran A, Babu CJ, Khan SB, Almusharraf A and Quasim MT. (2024). Advances in real time smart monitoring of environmental parameters using IoT and sensors. Heliyon 2024;10:e28193. https://doi.org/10.1016/j.heliyon.2024.e28195.

[17] Fusco P, Rimoli GP, Guerriero A, Palmieri M. (2025). On-device training and pruning for energy saving and continuous learning in resource-constrained MCUs. Futur Gener Comput Syst 2026;176:108194. https://doi.org/10.1016/j.future.2025.108194.

[18] Rehman SU, Usman M, Toor MH and Hussaini QA. (2024). Advancing structural health monitoring: A vibration-based IoT approach for remote real-time systems. Sensors Actuators A Phys 2024;365:114863. https://doi.org/10.1016/j.sna.2023.114863.

[19] Anuradha P, Vasanth K, Renuka G and Rao AR. (2023). IoT based enabling home automation system for individuals with diverse disabilities. E-Prime - Adv Electr Eng Electron Energy 2023;6:100366. https://doi.org/10.1016/j.prime.2023.100366.

[20] Wijayakusuma P, Hakim GPN and Li B. (2025). TerraGrow: Integrated platform for real time plant monitoring and automated watering system with IoT and fuzzy Sugeno Algorithm. HardwareX 2025;24:e00724. https://doi.org/10.1016/j.ohx.2025.e00724.

[21] Khan MA, Song W, Khan A, Ali M, Karim R and Zhan J. (2025). Machine learning hybrid dynamic best model selection algorithm for real-time fire prediction using IoT-enabled multi-sensor data in buildings. J Saf Sci Resil 2026;7. https://doi.org/10.1016/j.jnlssr.2025.100236.

[22] Wang C, Dun H, Liu L, Ding H, He H, Lin B, Hu W, Sun N, Hu Y and Yu B. (2025). IoT-enabled hierarchical thermoelectric nanocomposites for adaptive fire safety systems: Self-powered sensing and threshold-tunable alerts. Chem Eng J 2025;520. https://doi.org/10.1016/j.cej.2025.165998.

[23] Singh K, Yadav M, Barak D and Moreira F. (2025). Reliability of IoT Systems for Fire Detection. Procedia Comput Sci 2025;256:63–70. https://doi.org/10.1016/j.procs.2025.02.096.

[24] Morchid A, Alblushi IGM, Khalid HM, Alami R, Said Z, Qjidaa H, Cuce E, Muyeen SM and Jamil MO. (2025). Fire detection and anti-fire system to enhance food security: A concept of smart agriculture systems-based IoT and embedded systems with machine-to-machine protocol. Sci African 2025;27:e02559 https://doi.org/10.1016/j.sciaf.2025.e02559.

[25] Exaudi K, Stiawan D, Suprapto BY, Fakhrurroja H, Idris MY, Alghamdi TA and Budiarto R. (2025). An Improved Forest Fire Detection Model Using Audio Classification and Machine Learning. Comput Mater Contin 2025;86:1–24. https://doi.org/10.32604/cmc.2025.069377.

[26] Yuvaraj R, Kumar DS, Bhalerao SA, Murugesan K, Vellaiyan S and Minh NV. (2025). Real-time fire detection and suppression system using YOLO11n and Raspberry Pi for thermal safety applications. Case Stud Therm Eng 2025;75:107159. https://doi.org/10.1016/j.csite.2025.107159.

[27] Ramadan MNA, Basmaji T, Gad A, Hamdan H, Akgün BT, Ali MA, Alkhedher M and Ghaza M. (2024). Towards early forest fire detection and prevention using AI-powered drones and the IoT. Internet of Things 2024;27:101248. https://doi.org/10.1016/j.iot.2024.101248.

[28] Chen LW, Tseng HF, Cho CY and Tsai MF. (2024). Hybrid kitchen safety guarding with stove fire recognition based on the Internet of Things. J Netw Comput Appl 2024;229:103921. https://doi.org/10.1016/j.jnca.2024.103921.

[29] Kabir MH. (2021). Smart Attendance and Leave Management System Using Fingerprint Recognition for Students and Employees in Academic Institute. Int J Sci Technol Res 2021. https://www.researchgate.net/profile/Md-Kabir-79/publication/353220662.

[30] Irakomeye J, Nshimiye A, Ngondya K, Sumari EJ, Leo J and Sam A. (2022). Development of IoT based Security System for Fire Monitoring and Prevention in Tanzanian Industries. Int J Adv Sci Res Eng 2022. https://api.semanticscholar.org/CorpusID:245810304.

[31] Pan G, Xie Y and Yang Q. (2024). IoT-based cloud monitoring system for building fires. Int J Metrol Qual Eng 2025;16. https://doi.org/10.1051/ijmqe/2024020.