Internet of Things and Cloud Computing
2026, Vol. 14, No. 4, pp. 89–106
https://doi.org/10.11648/j.iotcc.20261404.11

Research Article

Design and Experimental Evaluation of an Artificial Intelligence of Things (AIoT)-enhanced Multi-Modal Safety System

Nnanna Ekedebe[1, *], Ezeh Chidubem Franklin[1], Reginald Ekene Ogu[2], Joe-Uzuegbu Chijioke[3], Ajere Ikenna Uzoma[1]

[1]Department of Information Technology, Federal University of Technology, Owerri, Nigeria
[2]Department of Computer Engineering, Federal University of Technology, Owerri, Nigeria
[3]Department of Electrical Engineering, Federal University of Technology, Owerri, Nigeria

Abstract

Fire and gas leakage incidents remain a leading cause of preventable casualties and property loss, yet most conventional detection systems lack real-time remote notification, multi-modal alerting and autonomous power backup. This work presents the design and experimental evaluation of an Artificial Intelligence of Things (AIoT)-enhanced fire and gas detection system built around an Arduino UNO Q dual-processor board. An MQ2 sensor detects combustible gases such as LPG (Liquified Petroleum Gas) and propane, while an IR flame sensor detects fire through infrared radiation in the 700-1100 nm band. An STM32U585 microcontroller performs real-time sensor acquisition, severity classification and alarm actuation, while a Qualcomm Dragonwing QRB2210 processor running Linux handles cloud connectivity and edge-AI inference. Sensor readings are processed by AI-based anomaly detection, a flicker filter that suppresses false positives, and a five-level severity classification, and are sent to the Blynk cloud for real-time remote monitoring. Local alerts are provided by a buzzer, colour-coded LEDs and an OLED display, and a SIM800L GSM module delivers SMS alerts to up to five authorised recipients and supports live room audio monitoring. A USB camera records a one-minute video clip at the onset of a confirmed alarm and stores it on the board and in the Blynk cloud, while an Edge Impulse image classification model labels the scene as person, animal, fire, smoke or background, so that every alert states whether people or animals are present. An 18650 lithium-ion battery with TP4056 charging and MT3608 boost modules sustains approximately fifteen hours of idle operation and eight hours under mixed alarm conditions without mains power. Experimental results confirm reliable detection, prompt alert delivery and uninterrupted operation, demonstrating the system's suitability for residential and light-industrial environments. This work extends an earlier ESP32-based prototype presented at IEEE NIGERCON.

Keywords

Sensors, Blynk, AI-anomaly Detection, IR Flame Sensor, MQ2 Sensor, Severity Range, Flicker Filter, Real-time Monitoring, Edge AI, Image Classification, Edge Impulse, Arduino UNO Q

*Correspondence: Nnanna Ekedebe (nekedebe@gmail.com)

Received: 30 July 2026; **Accepted:** 14 November 2026; **Published:** 27 November 2026

Copyright: © The Author(s), 2026. Published by Science Publishing Group. This is an **Open Access** article, distributed under the terms of the Creative Commons Attribution 4.0 License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution and reproduction in any medium, provided the original work is properly cited.

1. Introduction

Fire outbreaks constitute a devastating category of preventable disaster in Nigeria, often leading to a "downward spiral" in the economy through the extensive loss of lives and properties [21]. According to media reports, Nigeria recorded sixty-eight severe fire incidents between 2019 and 2021, resulting in seventy-nine fatalities and the destruction of millions of naira worth of goods in markets and shopping centers. [7].

Conventional gas and fire detection systems have historically relied on local hardware actuators — such as buzzers and LED displays — to issue audible and visual alerts upon threshold exceedance. While effective within their operational scope, such systems present several critical limitations: alerts are confined to occupants physically present within the monitored space; they convey no contextual information regarding the severity or nature of the detected hazard; they remain susceptible to environmental false positives, including sunlight-induced interference on infrared sensors; and they offer no mechanism for remote monitoring or external notification. These shortcomings represent significant gaps in safety coverage, particularly in unoccupied or large-scale environments where timely human response cannot be guaranteed.

The emergence of the Internet of Things (IoT) and artificial intelligence has fundamentally expanded what embedded detection systems can achieve. IoT-enabled devices can transmit sensor data in real-time across networked platforms and cloud infrastructure, enabling remote monitoring, automated alert processing, and data-driven hazard assessment [22]. Concurrently, the proliferation of affordable, high-capability microcontrollers — most notably dual-processor boards such as the Arduino UNO Q, which pairs a Qualcomm Dragonwing QRB2210 quad-core Arm Cortex-A53 processor running Linux with an STM32U585 Arm Cortex-M33 microcontroller, alongside Wi-Fi 5 and Bluetooth 5.1 connectivity — has substantially lowered the barrier to sophisticated IoT and edge-AI development, allowing machine-learning models to run locally on low-cost hardware instead of in the cloud [23].

This work is an extension of our earlier conference paper presented at the IEEE Nigeria International Conference on Digital Transformation (NIGERCON) [28], which reported an ESP32-based fire and gas alarm system combining MQ2 and IR flame sensing, a five-level severity classification, a flicker filter, Blynk cloud monitoring, SMS, voice and email alerts, and an alarm-triggered video clip recorded by an ESP32-CAM module. The present paper extends that work by migrating the design to the Arduino UNO Q dual-processor platform, adding on-device image classification with Edge Impulse (person, animal, fire, smoke and background classes), recording alarm video with a USB camera stored on both the board and the Blynk cloud, adding a lithium-ion battery power backup, and presenting an expanded related-work review and experimental evaluation.

This paper entails a more effective way to implement a fire and gas alarm system with the following key objectives;

1) To design and develop a fire and gas alarm system using the MQ2 gas sensor to trigger gas leaks and an IR sensor to detect fire.
2) To integrate this system with a Blynk cloud that provides us with an always on real-time notification dashboard.
3) To design a severity table that tells us five different case study for alarm detection and response.
4) To integrate email remote notifications.
5) To integrate an OLED display that provides a local visual alert in case of downtime and users that are deficient of hearing.
6) To deploy a small image classification model, trained with Edge Impulse, on the Arduino UNO Q so that the system can recognise safety-relevant classes (human, animal, fire, smoke and background) in the monitored area and attach this context to its alerts.

This report presents an IOT-based solution to fire and gas alarm systems utilizing an Arduino UNO Q board and a MQ2 and IR sensor that detects changes for both gas and fire and changes in the ADC value across different severity levels, supported by a small image classification model that runs on the board to recognise humans, animals, fire and smoke in the camera view. The remaining part of this paper is divided into two categories. Section 2 review related works that are relevant to our study. Section 3 discusses materials used for the project, methods, overall system design for the fire and gas alarm system, as well as, the seven major components at the physical layer, such as the MQ2 sensors and IR sensor, Arduino UNO Q board, USB camera with the on-device image classification model, Sim800l GSM module, OLED display, the serial monitor (baud rate: 115200), and the algorithms used. Section 4 presents the results and discussion, and Section 5 concludes the paper.

2. Related Work

Early embedded hazard detection literature primarily focused on localized gas sensing and basic physical containment. Annapurna et al. [1] introduced an LPG leakage detection system utilizing an MQ-2 semiconductor sensor paired with an Arduino Nano microcontroller, which actuates dual exhaust fans and toggles a relay to cut power to household appliances upon detecting combustible gases. In similar foundational works, Raj et al. [6] built an LPG monitoring module focused on automatic cylinder reservation via SMS, while Gavaskar et al. [4] introduced an IoT-based LPG alert and auto-exhaust system featuring solenoid shut-off valves and load cells. While effective for basic gas evacuation, these systems rely on single-threshold, 8-bit microcontrollers without optical flame detection or intelligence to filter out environmental noise. Our current system resolves these limitations by integrating an optical IR flame sensor operating in the 700 nm–

1100 nm spectrum alongside the MQ2 sensor onto a dual-processor Arduino UNO Q architecture (STM32U585 + Qualcomm QRB2210), incorporating a digital flicker filter to eliminate false optical triggers and providing multi-modal hazard detection.

Remote alert dispatch mechanisms in fire safety research have traditionally relied on Global System for Mobile Communications (GSM) short message services. Hima Jyothi et al. [7] proposed a multi-tier escalation protocol that sends SMS messages to homeowners, neighboring residences, and local fire stations as gas concentrations rise. Suwarjono et al. [3] enhanced SMS notification by embedding real-time GPS location coordinates into messages sent to fire departments, while Kharisma and Setiyansah [2] utilized a Wavecom GSM gateway to transmit emergency text alerts, and Saeed et al. [13] integrated IoT modeling into smart home environments for fire safety. Text-only SMS gateways provide unidirectional, static text alerts without real-time graphical monitoring, live situational verification, or interactive cloud telemetry. Our current system addresses these gaps by combining SIM800L SMS messaging with live cloud telemetry via the Blynk IoT platform over Wi-Fi 5 and Bluetooth 5.1. Furthermore, it introduces live room audio monitoring through an electret call-in feature and automatically captures and uploads a 1-minute USB camera video clip to the cloud upon alarm confirmation.

Deploying deep learning models on resource-constrained IoT devices requires extensive optimization to reconcile high computational demands with limited hardware memory. Park et al. [5] demonstrated an AI-driven fire detection framework combining CO2 sensors, heat sensors, and imaging algorithms, while Lee et al. [12] conducted a systematic evaluation of Deep Learning Model Optimization (DLMO) strategies—including lightweight convolutions, quantization, pruning, and knowledge distillation—across VGGNet and MobileNet architectures on edge devices. Similarly, Naveen and Kounte [10] developed a neural network compression pipeline combining filter pruning, fine-tuning, k-means clustering, and 8-bit quantization on ResNet34 and VGG16 models for smart vision tasks. Prior computer vision approaches generally depend on external high-power cloud servers or heavy computer-vision hardware, lacking localized edge inference and failing completely during electrical grid blackouts. Our system solves these critical weaknesses by running a lightweight Edge Impulse image classification model directly on the Arduino UNO Q board (categorizing camera frames into human, animal, fire, smoke, or background) and powering the architecture with an autonomous 18650 Li-ion battery backup circuit (TP4056 + MT3608), ensuring up to 15 hours of continuous offline operation during power cuts.

Resource-constrained edge nodes demand compact classifier structures to maintain high accuracy without consuming excessive RAM and FLASH memory. Khaya et al. [9] introduced SparseBonsai, an enhanced variant of the Bonsai tree algorithm incorporating dynamic regularization, adaptive sparsity, and learnable projection layers to achieve 86.91% classification accuracy with a compact model footprint of 0.048 MB on IIoT edge nodes. In healthcare edge monitoring, Yauri et al. [8] evaluated TinyML classification algorithms on ESP32 microcontrollers, establishing that decision trees offer the optimal balance of prediction speed and energy efficiency, whereas random forests maximize raw predictive accuracy. Kumar et al. [19] originally established the baseline Bonsai tree algorithm to perform accurate classification within 2 KB of memory. While these lightweight models optimize tabular feature inference, they lack multi-modal physical sensor integration and real-time vision verification. Our current system bridges this gap by combining physical gas/flame signal processing on an Arm Cortex-M33 core with edge vision inference on a Linux-driven Qualcomm quad-core processor, achieving both structured signal filtering and visual situational awareness.

As IoT safety networks expand, safeguarding edge infrastructure against traffic anomalies and external intrusions becomes vital. Gao et al. [11] developed a lightweight Temporal Convolutional Network (TCN) model for Edge-IIoTset security, pairing a hybrid feature selection method (variance filtering, mutual information, Pearson correlation) with dilated causal convolutions to achieve 93.79% accuracy with an ultra-small 16.25 KB parameter footprint. Amgbara et al. [16] introduced lightweight machine learning classifiers to evaluate complexity-accuracy trade-offs in personal IoT defense, Wahab et al. [17] constructed a fog-based Host Intrusion Detection System (HIDS) utilizing N-gram feature matrices and Multilayer Perceptrons, and Tanveer et al. [18] proposed the Light Ensemble Guard combining LightGBM and XGBoost for real-time attack detection. Existing network intrusion and anomaly detection models focus purely on cyber-packet traffic rather than physical hazard environments. Our system adapts the principle of localized anomaly filtering to physical safety by implementing a digital flicker filter and configurable multi-threshold severity range classification, effectively preventing physical sensor false positives triggered by transient electrical or optical noise.

Vision-based hazard detection offers a non-contact mechanism to identify visual flame and smoke plumes before thermal sensors react. Sharma et al. [14] combined IoT sensing with classical OpenCV color-space segmentation and frame differencing to identify fire regions in smart city surveillance feeds. Jadon et al. [15] introduced FireNet, a specialized lightweight CNN architecture optimized for real-time fire and smoke classification on low-power edge hardware. Listyorini and Rahim [20] applied fuzzy logic rules to multi-sensor inputs to handle environmental uncertainty in residential fire alerts. While early vision models and fuzzy logic rules improved detection accuracy over raw thresholds, standalone vision systems fail under low-light or occluded conditions and lack physical gas leakage monitoring. Our design overcomes vision occlusions by combining optical camera classification with physical MQ2 gas sensing, ensuring that invisible gas leaks are detected before ignition while visual confirmation is

captured when flames occur.

Comparing our proposed system against the broader spectrum of related literature reveals significant architectural breakthroughs. Conventional works cited in literature either focus exclusively on physical gas actuation [1], multi-sensor thermal thresholds [3], text-only SMS dispatches [7], or isolated computer vision models [5, 15]. None of the existing studies integrate real-time Edge-AI occupancy verification, live room audio monitoring, cloud video clip archiving, and autonomous Li-ion battery backup within a single embedded node. Our Arduino UNO Q dual-processor system resolves these collective deficiencies, delivering rapid flame/gas detection, false-alarm flicker filtering, multi-modal cloud/cellular alerts, visual occupant labeling, and complete immunity to power grid interruptions. Table 1 summarizes the related works reviewed in this section.

Table 1. *Summary of Related Works.*

Author(s)	Topic	Devices Used	Strengths	Weaknesses
Annapurna et al. [1]	LPG Gas Leakage Detection and Alert System	MQ-2 gas sensor, Arduino Nano, L293D motor driver, inlet/outlet exhaust fans, GSM module, and relay	Active physical prevention via exhaust ventilation and power cutoff relay switch	Lacks optical flame detection and AI false-alarm filtering; solved by our system's IR optical sensor, flicker filter, and dual-processor edge architecture.
Kharisma & Setiyansah [2]	Fire Early Warning System for Housing	TGS2600 smoke sensor, LM35 heat sensor, QM-6 gas sensor, ATEMega8 microcontroller, and Wavecom SMS gateway	Multi-sensor hazard coverage (smoke, heat, gas) with rapid SMS delivery across mobile networks	Basic processing without AI; solved by our UNO Q Edge Impulse image classification and cloud telemetry.
Suwarjono Suwarjono et al. [3]	Home Fire Detection System using Arduino and SMS Gateway	DS18B20 temperature sensor, MQ2 gas sensor, Arduino Uno, and GSM module SIM900	Transmits real-time GPS coordinates directly to fire departments and homeowners	Slow thermal response latency (up to 60 mins); solved by fast IR optical flame sensing (700-1100 nm) and configurable severity ranges.
Gavaskar et al. [4]	LPG Leakage Detection and Auto Exhaust System	Solenoid valve, fumes fan, fixed load cell, LCD, IoT module	Speedy response with automatic cylinder booking and physical gas shut-off valve	Dependent on fixed IoT infrastructure without video/audio validation; solved by onboard USB camera recording and SIM800L call-in audio.
Park et al. [5]	Dependable Fire Detection System with AI	Heat sensors, CO2 sensors, imaging/cameras, AI framework	High accuracy through multi-functional sensor fusion and image processing	High cost and server dependency; solved by local Edge Impulse inference on the Qualcomm QRB2210 processor.
Raj et al. [6]	LPG Gas Monitoring and Booking System	MQ-2 gas sensor, GSM module, LCD display, microcontroller	Automatic cylinder reservation via SMS upon low gas level detection	Focused on commercial booking rather than active fire/flame prevention; solved by multi-modal alerting and active power cutoff.
Hima Jyothi et al. [7]	Escalated Leak Detection Safety Measures	MQ gas sensor, buzzer, LCD display, GSM module	Hierarchical alert escalation (notifies neighbors and fire station as gas level rises)	Lacks active environmental control and video evidence; solved by automated 1-minute video upload to Blynk cloud.
Yauri et al. [8]	TinyML Model Evaluation on Microcontrollers	ESP32 microcontroller, Decision Trees, Random Forest, SVM, MLP	Benchmarks accuracy, latency, and RAM/FLASH metrics on edge healthcare nodes	Limited to tabular vital signs without physical safety sensor inputs; solved by combining physical sensor acquisition with Edge-AI.
Khaya et al. [9]	SparseBonsai Classifier for IIoT Edge Computing	Projection layer, dynamic L2 regularization, Bonsai tree architecture	86.91% accuracy with 0.048 MB model size and 35% faster inference on IIoT nodes	Applied to bearing fault vibration data rather than multi-modal fire hazards; solved by multi-sensor physical and visual hazard classification.

Author(s)	Topic	Devices Used	Strengths	Weaknesses
Naveen & Kounte [10]	Neural Network Compression Framework	Filter pruning, fine-tuning, k-means clustering, 8-bit quantization	Reduces DNN memory footprint by 80x with <1% accuracy loss on ResNet34/VGG16	Vision-only compression without physical sensor fusion; solved by pairing compressed vision models with physical MQ2/IR flame sensors.
Gao et al. [11]	Lightweight TCN Intrusion Detection for Edge-IIoT	Dilated causal convolutions, residual blocks, variance filtering, mutual info	ResNet, and MobileNet variants 93.79% detection accuracy with 16.25 KB parameter size on Edge-IIoTset	Focuses on network cyber-attacks rather than physical hazards; solved by applying digital anomaly filtering to physical fire/gas signals.
Lee et al. [12]	Deep Learning Optimization for On-Device AI	Lightweight convolution, PTQ/QAT quantization, pruning, knowledge distillation	Evaluates memory-delay trade-offs across VGGNet, ResNet, and MobileNet variants	Theoretical optimization benchmark; solved by real-world deployment of Edge Impulse vision models on the Arduino UNO Q.
Saeed et al. [13]	Smart Home Environment for Fire Prevention	IoT environmental sensors, intelligent rule engine, gateway	Early risk modeling and preventive safety rules for residential spaces	Static threshold rules without visual occupant verification; solved by camera classification labeling Person/Animal presence.
Sharma et al. [14]	Integrated Fire Detection using OpenCV Image Processing	OpenCV color segmentation, frame differencing, IoT sensor node	Low-cost visual flame detection in surveillance video streams	High false-alarm rate from light reflections; solved by integrating a hardware digital flicker filter and IR 700-1100 nm spectral filter.
Jadon et al. [15]	FireNet: Lightweight Fire & Smoke Detection	Lightweight custom CNN (FireNet), Raspberry Pi / edge hardware	Real-time frame processing for standalone visual fire identification	Lacks gas sensing and offline power backup; solved by integrating MQ2 gas detection and TP4056/MT3608 Li-ion battery circuit.
Amgbara et al. [16]	Lightweight ML Models for Personal IoT Defense	Decision Trees, Support Vector Machines, lightweight classifiers	Low-overhead anomaly detection for resource-limited personal IoT devices	Limited to cyber traffic analysis; solved by multi-modal physical hazard detection and cellular SMS notification.
Wahab et al. [17]	Fog-Based Host Intrusion Detection System	N-gram feature selection, Multilayer Perceptron (MLP), sparse matrices	Reduces fog-computing overhead while preserving attack detection accuracy	Dependent on fog nodes; solved by complete on-device edge processing on the Qualcomm QRB2210.
Tanveer et al. [18]	Light Ensemble Guard for Real-Time IoT Attack Detection	LightGBM, XGBoost, Extra Trees ensemble, AUC validation	High detection precision using majority voting on low-memory edge devices	High complexity for low-power microcontrollers; solved by efficient Edge Impulse model execution on the UNO Q Linux environment.
Kumar et al. [19]	Bonsai Algorithm for Resource-Constrained Nodes	Sparse projection, non-linear tree branching, 2 KB RAM footprint	Ultra-compact model execution for 8-bit/16-bit microcontrollers	Static regularization leads to underfitting on complex datasets; solved by dynamic severity thresholding and dual-core processing.
Listyorini & Rahim [20]	IoT Fire Detection Prototype Using Fuzzy Logic	MQ2 gas sensor, flame sensor, Arduino, Fuzzy Logic Controller	Smooth decision-making under ambiguous sensor readings	Lacks cloud telemetry, video verification, and battery backup; solved by Blynk cloud integration, USB video clip recording, and Li-ion power autonomy.

3. Materials and Methods

3.1. Materials

This study employed an experimental and IoT-based design to evaluate the performance, reliability, and responsiveness of a low-cost Smart Fire and Gas Alarm System integrating the MQ2 gas sensor and IR flame sensor for hazard detection. The system was implemented around an Arduino UNO Q board, whose STM32U585 microcontroller processed sensor readings, classified them across defined severity levels (safe, low, moderate, high, and critical), and triggered corresponding local and remote alert responses, including a buzzer, colour-coded LED indicators, an OLED visual display, Blynk cloud notifications, SMS and voice monitoring via a SIM800L GSM module, email alerts, and a one-minute alarm-triggered video clip captured by a USB camera connected to the board and stored to both its onboard storage and the Blynk cloud, while an on-device image classification model (person, animal, fire, smoke, background) provided scene context for each alert. The prototype was tested under both arm and disarm modes to assess its ability to correctly detect gas leaks and flame events while minimizing false positives from environmental factors such as ambient light interference. The methodological framework aligns with the stated research objectives and ensures a reproducible, data-driven evaluation of the proposed system's detection accuracy, alert latency, and autonomous power performance.

3.1.1. Detail of Cloud Storage and Technology

In Figure 1 below, Blynk is a cloud-based IoT platform that allows users to remotely monitor, control, and receive notifications from connected hardware in real-time. It provides a customizable mobile and web dashboard, supports virtual pins for two-way communication between the microcontroller and the cloud, and enables automated event triggers such as push notifications and email alerts, making it well suited for a remote fire and gas alarm application. Blynk was chosen because it allows real-time data visualization and alert automation from low-cost microcontrollers and single-board computers like the Arduino UNO Q with minimal configuration overhead, unlike more complex industrial SCADA or proprietary cloud platforms.

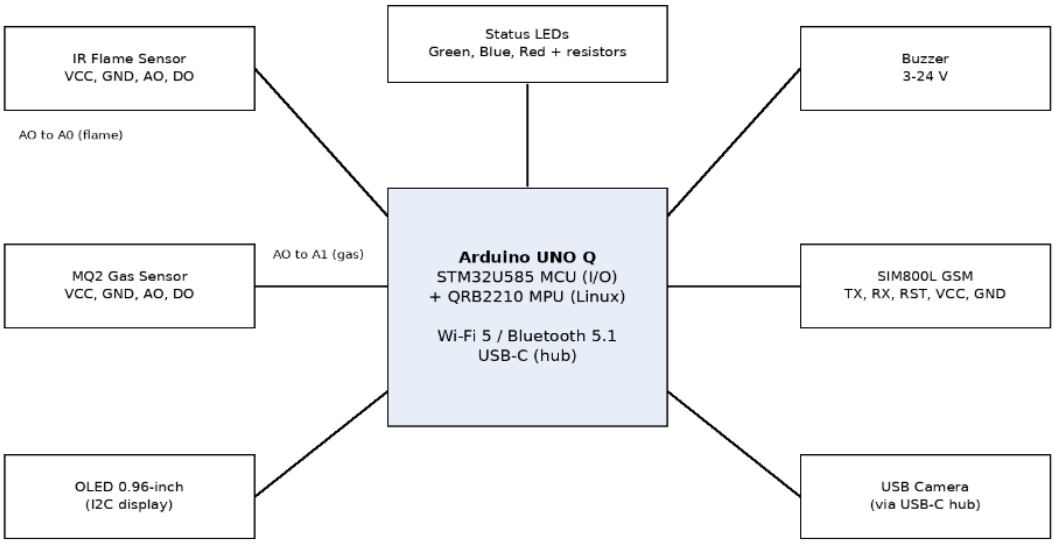

Figure 1. *System Block Diagram.*

The Blynk data flow begins with the MQ2 gas sensor and IR flame sensor capturing analogue readings of gas concentration and flame intensity, processed by the microcontroller with severity classification applied to map raw ADC values onto the five distinct level danger scale. Data is transmitted via Wi-Fi to the Blynk cloud, where it is stored, visualized on a live dashboard, and used to trigger automated alerts. End-users access the dashboard remotely and receive email notifications for informed fire and gas hazard response.

Furthermore, its cloud-based platform supports real-time gas and flame severity monitoring, offers built-in widgets for event-based automation and push notifications, and provides multi-channel alert mechanisms, including local buzzer and OLED activation, email dispatch, and SIM800L-routed SMS and voice calls, making it cost-effective, scalable, and ideal for continuous, low-latency hazard monitoring in residential and industrial environments.

3.1.2. Hardware Components

The hardware used in this work comprises an MQ2 gas sensor and an IR flame sensor for hazard sensing; an Arduino

UNO Q board (STM32U585 microcontroller and Qualcomm Dragonwing QRB2210 processor) for processing and cloud connectivity; a USB camera for video capture and scene classification; a SIM800L GSM module with an electret microphone for SMS alerts and live audio monitoring; a 0.96-inch OLED display, a 3-24V active buzzer and colour-coded LED indicators for local alerts; and an 18650 lithium-ion battery with a TP4056 Type-C charging module and an MT3608 boost converter for autonomous power. The sensors, the Arduino UNO Q board, the USB camera and the image classification model are described in Sections 3.3 to 3.7.

3.2. Methods

The prototype experiment was conducted within the Department of Information Technology, Federal University of Technology, Owerri, where the fire and gas alarm system was tested under controlled conditions simulating both gas leakage and open flame exposure. The MQ2 gas sensor and IR flame sensor were mounted on a prototyping breadboard within a confined test room, positioned to minimize exposure to direct sunlight and airflow that could introduce false readings, while still allowing controlled introduction of LPG vapour and a flame source for validation.

In Figure 2 below, the overall system design of the fire and gas alarm system is as follows:

We have a system of two (2) types of sensors, an Arduino UNO Q board (STM32U585 microcontroller and Qualcomm QRB2210 processor), a USB camera with an on-device image classification model, a SIM800L GSM module, an OLED display, and a buzzer.

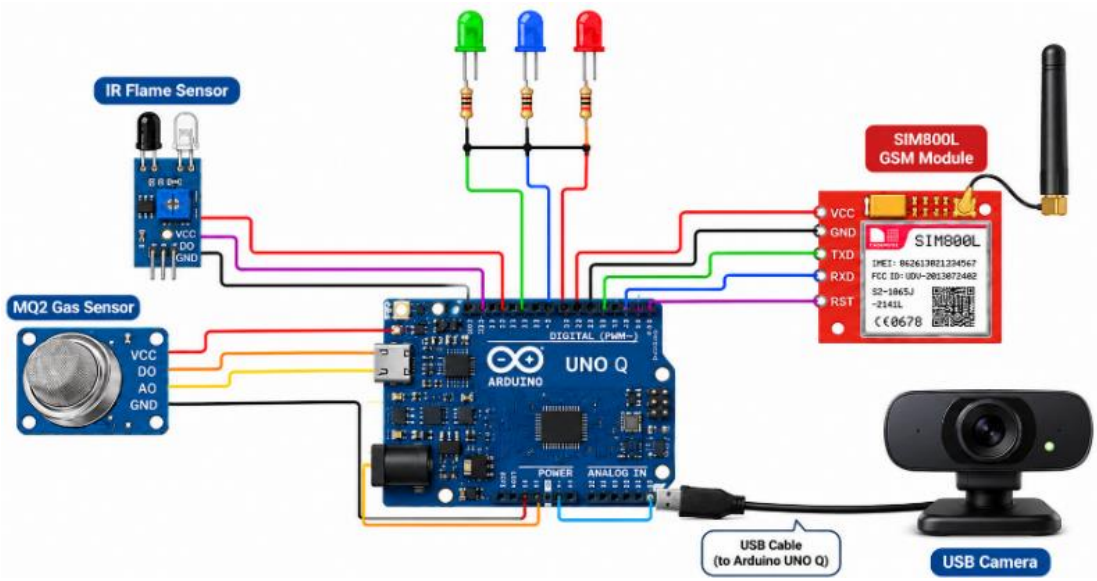

Figure 2. *Iot based smart fire and gas alarm system.*

Basically, the two types of sensors are the MQ2 gas sensor and the IR flame sensor. The MQ2 sensor is used to detect the presence of combustible gases such as Liquified petroleum gas (LPG), propane, and methane in the surrounding air. The second one, the IR flame sensor, is used to detect the presence of fire or flame by sensing infrared radiation emitted from a hot body. The microcontroller collects the analogue readings from both sensors, classifies them across the five-level severity scale, and transmits the real-time status to the Blynk cloud via Wi-Fi for remote monitoring, while simultaneously triggering local responses through the buzzer and OLED display, and remote responses through the SIM800L module's SMS, voice call, and email alert functions.

Together, both gas concentration and flame detection, when integrated through an IoT-based system like the one developed in this research, offer a faster, more comprehensive, and more reliable means of early hazard detection than either sensor could achieve alone. The use of the MQ2 and IR flame sensors demonstrates that even low-cost, readily available components, when properly calibrated and paired with severity-based logic, can deliver dependable alerts for life safety and property protection.

3.3. MQ2 Gas Sensor Module



Figure 3. *Image of the wiring pin configuration of MQ2 Sensor.*

Figure 3, above describes a Smoke/LPG/CO Gas Sensor module which utilizes MQ-2 as the gas detecting component and has a protection resistor and an adjustable resistor on board. MQ2 Gas Sensor module is useful for detecting gas leakage in home and industry. It can detect LPG, butane, methane, alcohol, Hydrogen and smoke. Based on its fast response time, measurements can be taken as soon as possible. Also, the sensitivity can be adjusted by the potentiometer.

Gas severity Table

The severity table in Table 2 summarizes the different type of severity across a 0-100% danger level; as ADC (Analogue Digital converter) values increase, the severity and danger level increase.

At low severity levels, the system displays the alarm state but does not activate the audible alarm until reaching moderate severity. This design choice reflects our experimental observation that low ADC values might result from poor air quality rather than actual gas leaks.

Table 2. *Gas sensor severity table, ADC ranges, and danger percentage mapping.*

Level	ADC Range	Danger%	Severity Type	Recommended Actions
0	0-799	0%	Safe	Clean air: Normal operation
1	800-1199	1-25%	Low	Trace gas, ventilate room
2	1200-1799	26%-60%	Moderate	Gas present. Local alarm triggered (Blue LED turns on and buzzer is triggered)
3	1800-2599	61-85%	High	Significant concentration of gas in the environment N:B: Do not light a match
4	2600-4095	86-100%	Critical	Dangerous level or smoke.

The results, in Figures 4 and 5 below, show how the MQ2 gas sensor's ADC readings map onto the five-level severity scale used in this study.

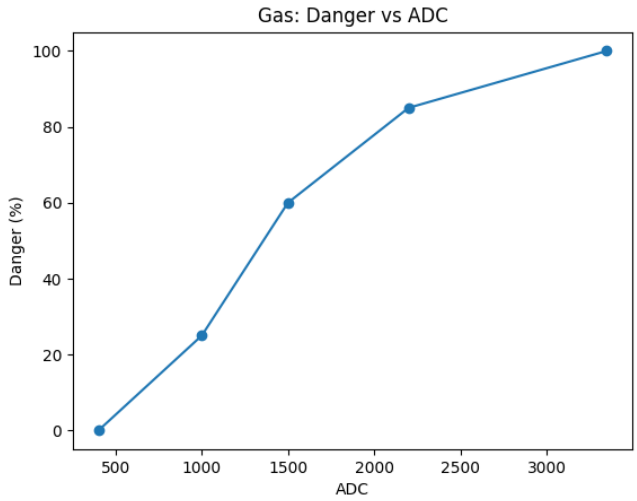

Figure 4. Gas Severity: Danger level vs ADC value.

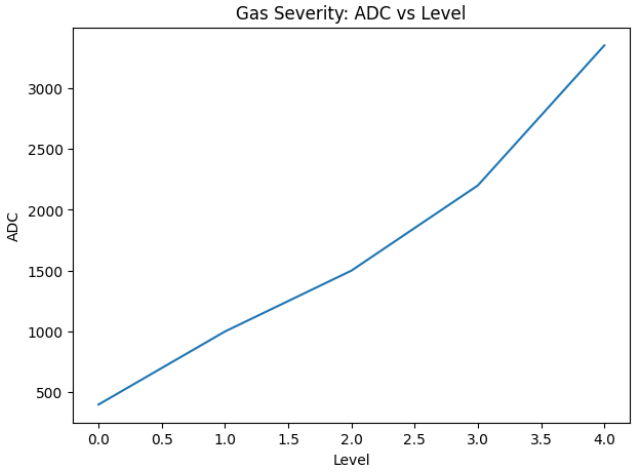

Figure 5. Gas Severity: ADC value vs Level Type.

Iteration I: The gas severity table recorded a steady increase in danger percentage as ADC values rose from baseline air-quality levels through to the critical gas concentration threshold, reflecting consistent and predictable sensor behaviour across the 0-100% danger scale. This relationship is important because it directly governs when the local buzzer, OLED alert, and remote Blynk, SMS, and email notifications are triggered. At low severity, the alarm state is displayed but the audible alarm is deliberately withheld, since experimental observation showed that low ADC values can arise from minor air quality fluctuations rather than an actual gas leak. Reliable severity mapping therefore ensures early detection of genuine hazards while minimizing nuisance alarms, supporting safer and more trustworthy operation in both domestic and industrial settings.

3.4. IR Flame Sensor

Figure 6 below shows a flame sensor that can detect the presence of fire or flames. The flame sensor works based on infrared radiation. The IR photodiode will detect the IR radiation from any hot body. This value is then compared with a set value. Once the radiation reaches the threshold value, the sensor will change its output accordingly. It is widely used in fire detection systems in homes and industries [24].

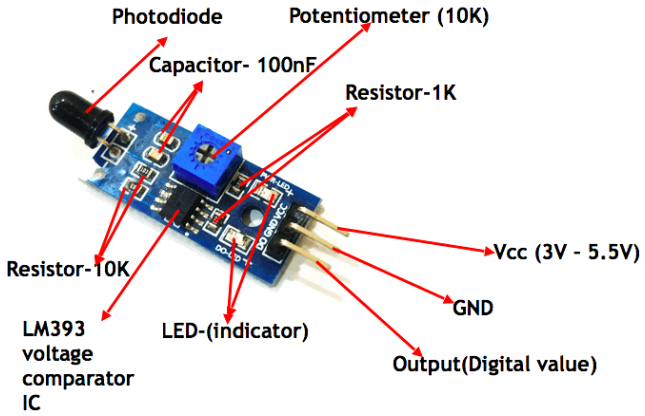

Figure 6. IR flame sensor module.

The flame sensor detects the presence of fire or flames based on infrared radiation detection. The IR photodiode detects IR radiation from hot bodies, and this value is compared against a set threshold. Once radiation reaches the threshold value, the sensor changes its output accordingly. It is widely used in fire detection systems in homes and industries.

IR Flame sensor severity table

The severity table summarizes the different type of severity across a 0-100% danger level; as ADC (Analogue Digital converter) gets lower the severity and danger level increases.

At severity type of low the system displays the alarm state, but the alarm does not go off until the state gets to moderate because studies and experiments shows that it might not be a flame as the IR is said to also be triggered by Sunlight like that from the window sending high ADC values.

Table 3. Flame sensor severity table, ADC ranges, and danger percentage mapping.

Level	ADC Range	Danger%	Severity Type	Recommended Actions
0	2801-4095	0%	Safe	Normal. No action required
1	2200-2800	1-25%	Low	Warm object or distant heat source

Level	ADC Range	Danger%	Severity Type	Recommended Actions
2	1800-2199	26-60%	Moderate	Possible flame -flicker confirmed. Local alarm ON (Red LED turns on)
3	1200-1799	61-85%	High	Confirmed nearby flame. Evacuate area
4	0-1199	86-100%	Critical	Large or close flame. Emergency

The results, in Figures 7 and 8 below, show how the IR flame sensor's ADC readings map onto the five-level severity scale used in this study.

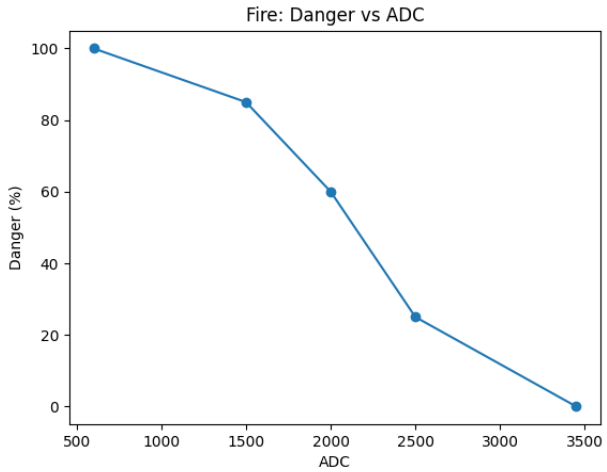

Figure 7. *Flame Severity: Danger level vs ADC value.*

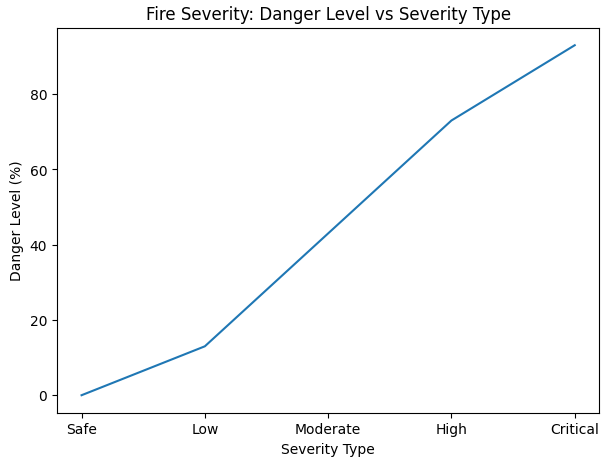

Figure 8. *Flame Severity: ADC value vs Level Type.*

Iteration II: The flame severity table recorded a steady increase in danger percentage as ADC values fell from baseline ambient-light levels through to the critical flame-detection threshold, reflecting the inverse relationship between ADC output and fire severity that is characteristic of the IR photodiode's response to infrared radiation. This relationship is important because it directly governs when the local buzzer, OLED alert, and remote Blynk, SMS, and email notifications are triggered. At low severity, the alarm state is displayed but the audible alarm is deliberately withheld, since experimental observation showed that low-severity readings can arise from ambient infrared sources such as sunlight through a window rather than an actual flame. Reliable severity mapping therefore ensures early detection of genuine fire hazards while minimizing nuisance alarms caused by environmental light interference, supporting safer and more trustworthy operation in both domestic and industrial settings.

3.5. Arduino UNO Q Board

The Arduino UNO Q in Figure 9 is a dual-processor board in the classic UNO form factor that combines a microcontroller and a Linux-capable microprocessor. Its real-time side is an STM32U585 microcontroller (Arm Cortex-M33 core running at up to 160 MHz, with 2 MB of flash and 786 kB of SRAM) that runs Arduino sketches over the Zephyr OS and, in this system, reads the MQ2 and IR flame sensors, applies the severity classification, and drives the buzzer, LEDs, OLED display and SIM800L module. Its application side is a Qualcomm Dragonwing QRB2210 microprocessor (quad-core Arm Cortex-A53 at 2.0 GHz with an Adreno GPU) running a Debian-based Linux environment with onboard eMMC storage, which in this system runs the image classification model, the Blynk and email clients, and the video recording routine. The board also provides dual-band Wi-Fi 5 and Bluetooth 5.1, and the two processors exchange data through the Arduino Bridge, so that sensor values and alarm states computed on the microcontroller are available to the Linux applications [23].



Figure 9. *Arduino UNO Q board (Rear View).*

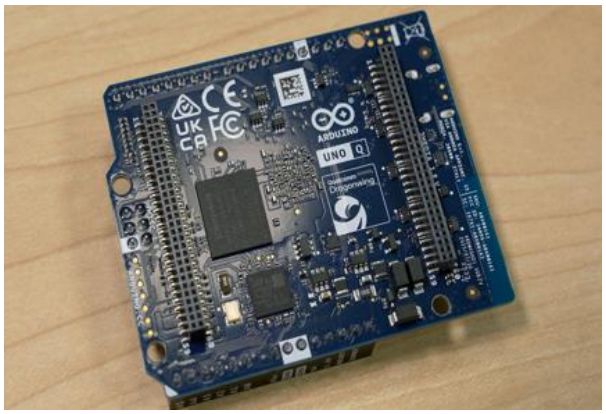

Figure 10. Arduino UNO Q board (Back View).

3.6. USB Camera and On-Device Image Classification

The camera in Figure 11 is a low-cost USB webcam connected to the UNO Q through its USB-C port (via a powered USB-C hub). The Linux side of the board reads video frames directly from the webcam, so no separate camera microcontroller or inter-board trigger line is required. The same frames are used for two purposes: image classification of the monitored scene, and recording of alarm video clips.



Figure 11. USB camera module connected to the Arduino UNO Q.

To complement the audible, visual, and remote text-based alert channels already described, the system incorporates a small image classification model that interprets what the camera sees. Frames from the USB camera are resized to a compact square input and passed to a convolutional model, trained in Edge Impulse and executed on the Cortex-A53 cores of the UNO Q using the Edge Impulse Linux runtime [25]. The model assigns each frame to one of five classes: person (human), animal, fire, smoke, and background. The predicted class and its confidence are used as context only and do not replace the sensor-based severity logic: they are attached to the SMS, email and Blynk alerts so that responders know whether people or animals are present in the hazard area, and a fire or smoke prediction provides visual corroboration of the gas and flame sensor readings.

When the overall danger level for either gas or flame crosses the moderate-severity threshold, the same threshold at which the local buzzer and the gas/flame LED indicators are activated, the microcontroller signals the Linux side of the board through the Bridge. The Linux application then records a sixty-second (1-minute) video clip of the monitored area at the onset of the hazard and commits it to two destinations in parallel: first, it is written as a timestamped file onto the board's onboard storage, ensuring that a local video record persists even during Wi-Fi outages or Blynk server downtime; second, the same clip is uploaded over Wi-Fi to the Blynk cloud, where it is attached to the corresponding event entry so that registered users can retrieve and play back the footage remotely from any internet-connected device. This dual-storage arrangement extends the local/remote redundancy already adopted for the OLED-buzzer and email-SMS pathways to visual evidence: a fast, always-available local copy for on-site or offline review, and a cloud-hosted copy on Blynk for remote verification, incident review, and, where required, use as supporting evidence for emergency responders.

3.7. Image Classification Model Development with Edge Impulse

The image classifier is developed in Edge Impulse Studio on the free Developer plan as a five-class model with the labels person (human), animal, fire, smoke, and background. A deliberately small dataset is assembled from public sources and from the test environment: fire and smoke images are drawn from the D-Fire dataset [26]; person and animal images from a class-filtered subset of the COCO dataset [27]; and background images, showing the empty test room under normal and varied lighting, are captured with the same USB camera so that the model is exposed to the deployment conditions. The images are uploaded to the Studio, labelled by class, and split into training (80%) and test (20%) sets.

The impulse resizes each image to 96 × 96 pixels (RGB), and a transfer-learning image classification block is trained with data augmentation and then quantised to 8-bit integers to reduce memory use and inference time. The model is deployed with the Linux (AARCH64) target and executed on the UNO Q. Model performance is assessed on the held-out test set using accuracy, per-class precision and recall, and the confusion matrix, together with the on-device inference time per frame.

4. Results and Discussion

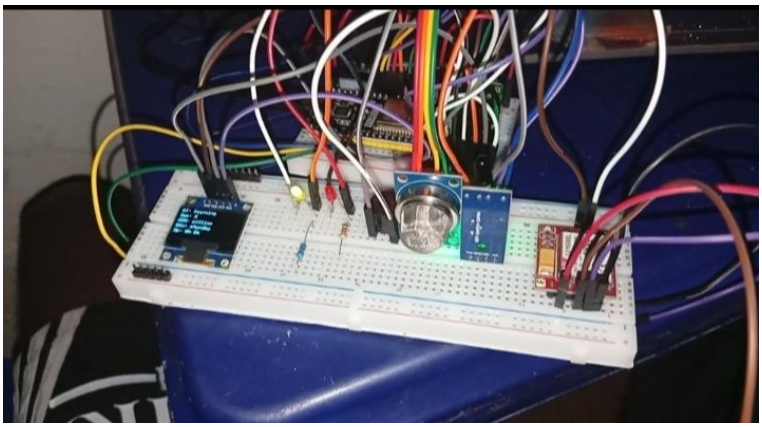

Figure 12. *Fire and gas alarm system prototype.*

In the figure above, the prototyping board consists of the Arduino UNO Q board, the SIM800L IoT gateway module, Mq2 gas sensor, IR flame sensor and the other components that makes the system to work.

The results, in Figure 12 above, represent the outcome of our IOT based Fire and Gas alarm system, consisting of the two types of sensors used, the MQ2 Gas sensor and the IR flame sensors. The MQ2 sensor is used to detect various gas leaks e.g. propane, butane, LPG etc. Flame sensors are used to detect flames. The OLED display serves as a visual dashboard to view the states of the system. The system on active has 2 modes; arm and disarm. Disarm mode set the system to be in dull state where by it might detect flames of gas but will not trigger an alert both locally or remotely, Arm mode put the system in monitoring mode and it sends real time alert when alarm is triggered by either gas, flame or combination of both in delicate situations.

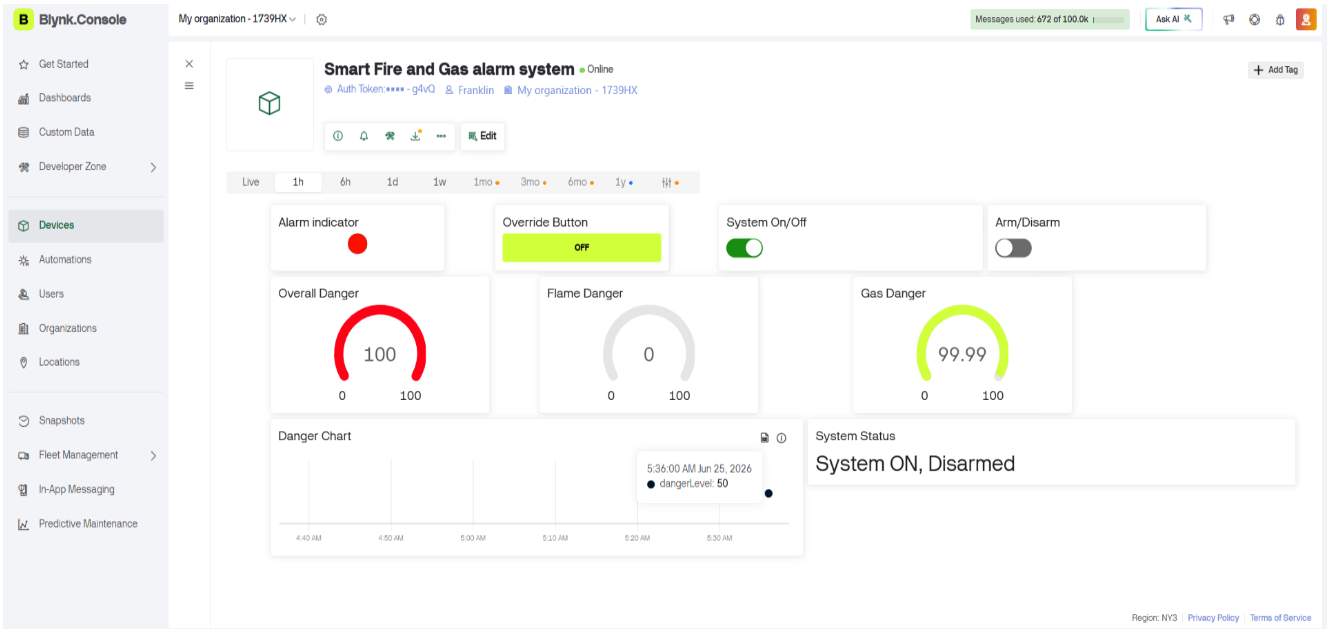

Figure 13. *Blynk Iot dashboard for smart fire and gas alarm system.*

In the Figure 13 above the Blynk dashboard provides a remote control and monitoring interface for the fire and gas alarm system, accessible from any internet-connected device. At the top of the panel, an alarm indicator shows the current alert state at a glance, alongside an override button that allows a user to manually silence or reset an active alarm, a System On/Off toggle that powers the device remotely, and an Arm/Disarm toggle that switches the system between active

monitoring and a dormant state without disabling the hardware itself. Below these controls, three circular gauges, Overall Danger, Flame Danger, and Gas Danger, display live readings on the same 0-100% severity scale used locally on the OLED, giving the user an immediate sense of how close any single hazard or the combined risk is to triggering a full alarm. At the time of this screenshot, the system reported an overall danger level of 5%, with no flame detected and a gas reading of 5.2%, both comfortably within the safe range, while the accompanying danger chart and system status panel log this activity over time and confirm the system was in an OFF state.

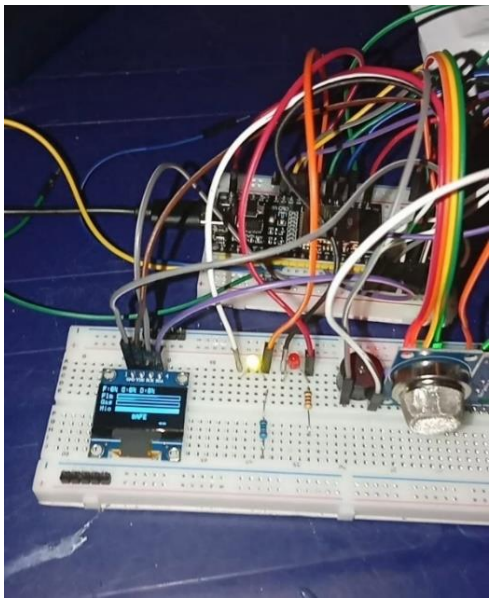

Figure 14. *Final result of the project.*

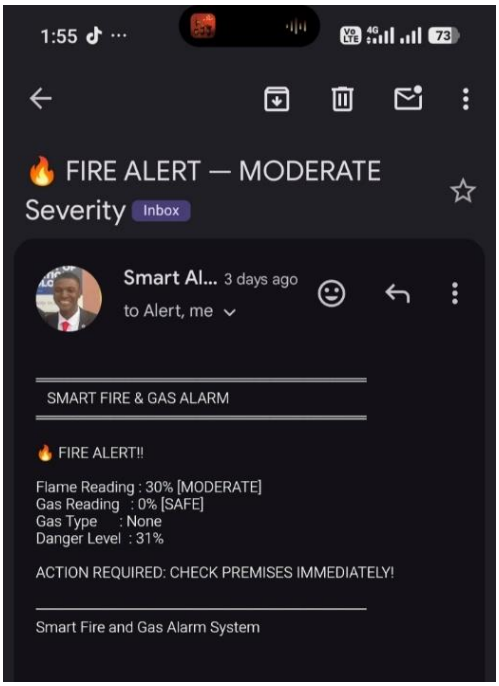

Figure 15. *Email notification when fire alert is triggered.*

In Figure 15 above, the system's email alert demonstrates the remote notification pathway that complements the local alarm response. When the flame sensor recorded a reading of 30%, classified as moderate severity, with no gas detected and an overall danger level of 31%, the Arduino UNO Q dispatched an automated email via the Google Gmail API to the registered address, clearly stating the alert type, individual flame and gas readings, the gas type detected (or none, as in this case), the overall danger percentage, and a direct call to action instructing the recipient to check the premises immediately. This confirms that the severity classification logic correctly distinguished a moderate flame event from a safe gas reading and translated that distinction into a specific, actionable message rather than a generic alarm notification.

Figure 14 shows the corresponding local hardware response on the breadboard prototype, where the Arduino UNO Q, OLED display, and supporting components are wired together to provide an immediate, on-site visual indication of the same event. The 0.96-inch OLED renders the live sensor states directly at the device, ensuring that anyone physically present, including users who may not have immediate access to the registered email or phone, can still see the system's status without depending on network connectivity. Together, these two figures illustrate the dual-layer alert design at the core of this project: a local, always-available visual response for on-site occupants, and a remote, detail-rich email notification for users who are away from the premises, both driven from the same severity classification so that the information communicated locally and remotely never conflicts.

Beyond the OLED and the Blynk gauges, the system also relies on a simple colour-coded LED scheme to communicate hazard type at a glance. As shown in Tables 2 and 3, the gas-detection LED illuminates blue once gas severity reaches the moderate level, while the flame-detection LED illuminates red once flame severity reaches moderate, with both LEDs remaining off during safe and low-severity readings so as not to desensitise users to frequent, low-risk fluctuations. Where gas and flame are both classified as moderate or higher at the same time, the two LEDs activate simultaneously, giving an immediate visual cue that the hazard is compound rather than singular, without requiring the user to read the OLED text or consult the dashboard gauges to identify which sensor triggered the alert.

This colour convention is mirrored on the Blynk dashboard through two virtual LED widgets bound to dedicated virtual pins, one for the gas indicator and one for the flame indicator. Each widget remains in its off (grey) state while the corresponding severity is safe or low, switches to a solid blue or red fill the moment the Arduino UNO Q reports a moderate-or-higher reading for gas or flame respectively, and reverts to off once the severity falls back below the moderate threshold. Because the virtual indicators update over the same Wi-Fi connection used for the danger gauges, a remote user sees the colour change within the same reporting interval as the physical LEDs at the device, so the local and cloud indicators never fall

out of sync. This gives users a colour-coded, at-a-glance status check that requires no interpretation of numeric gauges and remains consistent whether the system is viewed on-site or from the Blynk mobile app.

The camera video feature operates on the same moderate-severity trigger used by the buzzer, OLED, and LED indicators, so that the instant an alarm condition is confirmed, the Arduino UNO Q begins recording a sixty-second clip of the monitored area. Once captured, the clip is written to the board's onboard storage under a timestamped filename for permanent local storage, and a copy is simultaneously pushed to the Blynk cloud, where it appears alongside the corresponding email notification and dashboard log entry as a retrievable item that authorised users can open and play back remotely. In practice, this means that an event such as the moderate flame reading shown in Figure 15 would, in addition to triggering the red flame LED and the email alert, also generate a one-minute video record of the area at the moment of detection, giving responders direct visual confirmation of the hazard before they arrive on site, while the local copy remains available even on occasions when the upload to Blynk is delayed or interrupted by a weak network connection.

The on-device image classifier in Figure 16 adds context to this record: the class predicted for the scene (person, animal, fire, smoke, or background) and its confidence are attached to the alert, so that the recipient knows not only that a hazard was detected but also whether people or animals appear to be in the monitored area.

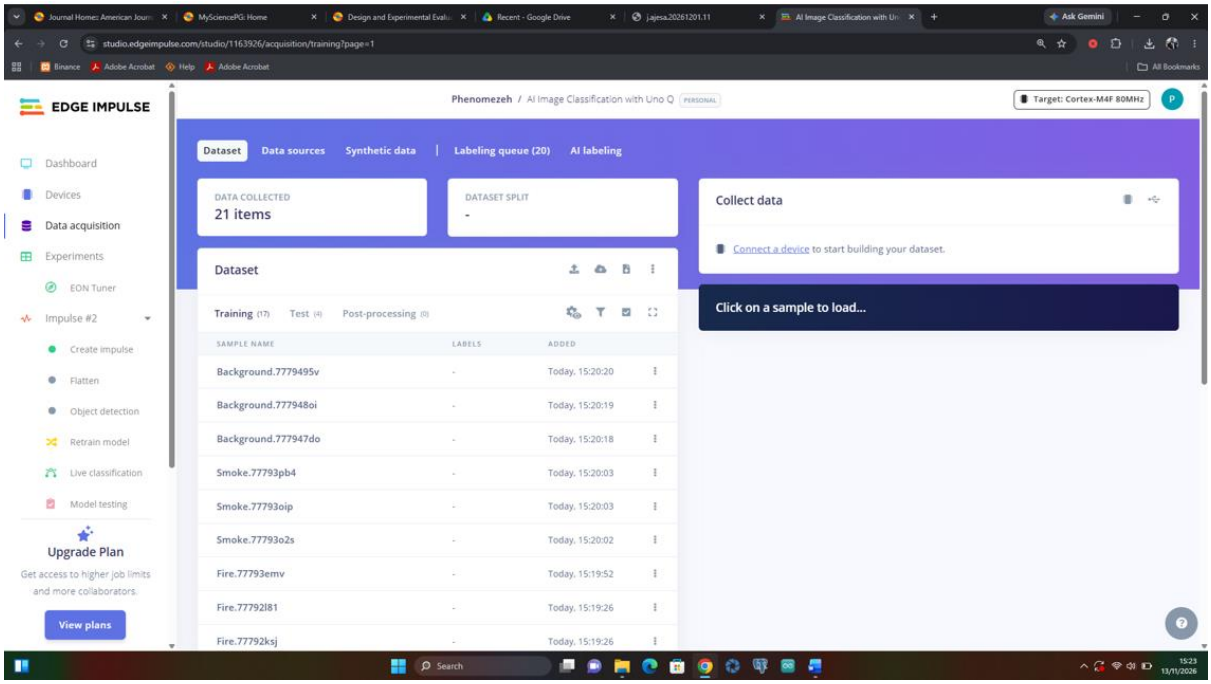

Figure 16. Edge Impulse project dashboard for the on-device image classification model (AI Image Classification with Uno Q).

5. Conclusion

This paper presents a well-documented design and implementation of an IoT-based fire and gas alarm system intended to be accessible to a broad range of users, including older adults, children, and individuals with disabilities. The system addresses a notable gap in existing solutions, namely limited accessibility, by offering a low-cost design that delivers both local and remote alerts. For visually impaired users, it provides an audible alarm response that can be paired with a Bluetooth speaker for greater volume and portability. For users with hearing impairments, it incorporates a 0.96-inch OLED display in place of the LCD displays used in prior studies, as the OLED offers clearer contrast and more efficient visual communication of system status.

The system further integrates the Blynk IoT cloud platform, enabling automated responses and real-time notifications over Wi-Fi, complemented by email alerts delivered through the Google Gmail API. Unlike conventional alarm systems that typically provide only a binary alert, this system is capable of classifying hazard severity across five distinct levels, safe, low, moderate, high, and critical, with each level triggering a unique and proportionate response. This response is further reinforced by colour-coded blue and red LED indicators that distinguish gas from flame events at a glance, both locally and through mirrored virtual indicators on the Blynk dashboard, and by a USB camera on the Arduino UNO Q that captures a one-minute video clip of the hazard at the moment of detection, storing it simultaneously to the board's onboard storage and the Blynk cloud, while a small image classification model deployed on the board identifies whether humans, animals, fire

or smoke are present in the scene, so that occupants and remote responders alike have visual, not merely numeric, confirmation of the alarm event.

Future work will extend the classifier with additional safety-relevant classes and evaluate its on-device latency and accuracy under a wider range of lighting conditions.

Abbreviations

ADC	Analog-to-Digital Converter
GSM	Global System for Mobile Communications
IoT	Internet of Things
WSN	Wireless Sensor Network

Author Contributions

Nnanna Ekedebe: Conceptualization, Methodology, Project administration, Resources, Supervision, Validation, Writing – original draft, Writing – review & editing

Ezeh Chidubem Franklin: Investigation, Methodology, Software, Visualization, Writing – review & editing

Reginald Ekene Ogu: Methodology, Validation

Joe-Uzuegbu Chijioke: Formal Analysis, Investigation, Methodology

Ajere Ikenna Uzoma: Software, Validation

Conflicts of Interest

The authors declare no conflicts of interests.

Appendix: Algorithms and Code Listings of the Fire and Gas Alarm System

Algorithm I: Sensor Reading and Severity Classification Algorithm

Input: Flame sensor analogue pin (A0), Gas sensor analogue pin (A1), ADC resolution set to 12 bits (0-4095), flame and gas digital pins

Output: Flame severity percentage, gas severity percentage, overall danger level, gas type classification

```
float flamePct(int a){
    if(a>FL_SAFE) return 0;
    if(a>FL_LOW) return 1+(float)(FL_SAFE-a)/(FL_SAFE-FL_LOW)*24;
    if(a>FL_MODERATE) return 26+(float)(FL_LOW-a)/(FL_LOW-FL_MODERATE)*34;
    if(a>FL_HIGH) return 61+(float)(FL_MODERATE-a)/(FL_MODERATE-FL_HIGH)*24;
    return min(86+(float)(FL_HIGH-a)/FL_HIGH*14,100.f);
}

float gasPct(int a){
    if(a<=GS_SAFE) return 0;
    if(a<=GS_LOW) return 1+(float)(a-GS_SAFE-1)/(GS_LOW-GS_SAFE)*24;
    if(a<=GS_MODERATE) return 26+(float)(a-GS_LOW-1)/(GS_MODERATE-GS_LOW)*34;
    if(a<=GS_HIGH) return 61+(float)(a-GS_MODERATE-1)/(GS_HIGH-GS_MODERATE)*24;
    return min(86+(float)(a-GS_HIGH-1)/(4095-GS_HIGH)*14,100.f);
}

bool detectFlicker(){
    int mn=4096, mx=-1;
    for(int i=0;i<FLICKER_SAMPLES;i++){
        int s=analogRead(FLAME_ANALOG_PIN);
        if(s<mn) mn=s;
        if(s>mx) mx=s;
        delay(FLICKER_WINDOW_MS/FLICKER_SAMPLES);
    }
    return (mx-mn)>=FLICKER_MIN_PP;
}
```

Algorithm II: Blynk Cloud Integration and Synchronization Algorithm

Input: Computed flame percentage, gas percentage, danger level, local alarm state

Output: Real-time values displayed on the Blynk dashboard, remote override command received from the Blynk app

```
#define VP_DANGER V0
#define VP_FLAME_PCT V1
#define VP_GAS_PCT V2
```

```
#define VP_ALARM V3
#define VP_OVERRIDE V4

BLYNK_WRITE(VP_OVERRIDE) {
    overrideOn = (param.asInt() == 1);
    if (overrideOn) localAlarm = true;
}

BLYNK_CONNECTED() { Blynk.syncVirtual(VP_OVERRIDE); }

void syncBlynk(bool alm){
    Blynk.virtualWrite(VP_DANGER, curDanger);
    Blynk.virtualWrite(VP_FLAME_PCT, curFlamePct);
    Blynk.virtualWrite(VP_GAS_PCT, curGasPct);
    Blynk.virtualWrite(VP_ALARM, alm1:0);
}

// in loop():
Blynk.run();
if(now-lastBlynkSync>=BLYNK_INTV){
    lastBlynkSync=now;
    syncBlynk(localAlarm);
}
if(localAlarm && !wasAlarm) Blynk.logEvent(EVT_ALARM_ON);
```

Algorithm III: GSM Number Authorization and SMS/Voice Alert Algorithm

Input: Authorised phone number list, alert message string, incoming call signal from the SIM800L module

Output: SMS broadcast delivered to authorised numbers only, voice channel opened for an authorised caller.

```
const char* SMS_NUMBERS[] = {
    "+2348123547102", "+2349158221390", "+2349011223344", "", ""
};
const int SMS_COUNT = 5;

void broadcastSMS(const String& msg){
    for(int i=0;i<SMS_COUNT;i++){
        if(strlen(SMS_NUMBERS[i])<5) continue; // skip unused/blank slots
        sendSMS(SMS_NUMBERS[i], msg);
        delay(SMS_GAP_MS);
    }
}

// In initGSM():
modem.sendAT("+CLIP=1"); // enable caller ID
modem.sendAT("+S0=" + String(AUTO_ANSWER_RINGS)); // auto-answer after 1 ring
modem.sendAT("+CMIC=0,10"); // set microphone gain

void checkCallStatus(){
    if(simSerial.available()){
        String line = simSerial.readStringUntil('\n');
        if(line.indexOf("RING")>=0 && !callActive) callActive=true;
        if((line.indexOf("NO CARRIER")>=0||line.indexOf("NO ANSWER")>=0) && callActive) callActive=false;
    }
}
```

Algorithm IV: On-Device Image Classification, Alarm-Triggered Recording and Dual Video Storage Algorithm

Input: Overall danger level and gas/flame severity classification from the STM32U585 microcontroller (received over the Arduino Bridge), USB camera frames, Edge Impulse model file, onboard storage status, Wi-Fi/Blynk connection status

Output: Predicted scene class and confidence attached to the alert, timestamped 1-minute video file saved to the onboard storage, corresponding video file uploaded to the Blynk cloud and linked to the dashboard event log

```
from edge_impulse_linux.image import ImageImpulseRunner
import cv2, time

CLASSES = ["person", "animal", "fire", "smoke", "background"]
CONF_MIN = 0.60 # minimum confidence to accept a label
CLIP_SECONDS = 60 # 1-minute clip length
CLIP_DIR = "/home/arduino/clips" # onboard eMMC storage

runner = ImageImpulseRunner("model.eim"); runner.init() # load Edge Impulse model
cam = cv2.VideoCapture(0) # USB camera

def classify(frame):
    scores = runner.classify(runner.get_features_from_image_auto_studio_settings(frame)[0])["result"]["classification"]
    label = max(scores, key=scores.get)
    return (label, scores[label]) if scores[label] >= CONF_MIN else ("unknown", scores[label])

def on_alarm(level, gas_pct, flame_pct): # called by the MCU through the Bridge
    ok, frame = cam.read()
    label, conf = classify(frame) # scene context for the alert
    notify(level, gas_pct, flame_pct, label, conf) # add label to Blynk, email and SMS alerts
    filename = CLIP_DIR + "/ALARM_" + timestamp() + ".avi"
    writer = cv2.VideoWriter(filename, cv2.VideoWriter_fourcc(*"MJPG"), 10, (640, 480))
    start = time.time()
    while time.time() - start < CLIP_SECONDS:
        ok, frame = cam.read()
        if ok: writer.write(frame) # append frame to local AVI
    writer.release() # finalise local copy
    if not blynk_upload(filename): # push clip to Blynk cloud
        queue_for_retry(filename) # retry later if Wi-Fi/Blynk unavailable
```

References

[1] G. Annapurna, A. L. Priya, B. L. Malvika, S. Tahreem, V. Juhitha, and B. Neha, "LPG Gas Leakage Detection and Alert System," *Int. J. Res. Appl. Sci. Eng. Technol. (IJRASET)*, vol. 11, no. 5, pp. 2400-2405, 2023, https://doi.org/10.22214/ijraset.2023.50964

[2] R. S. Kharisma and A. Setiyansah, "Fire Early Warning System for Housing using Fire Sensors, Microcontroller and SMS Gateway," *J. Robot. Control (JRC)*, vol. 2, no. 5, pp. 388-394, 2021, https://doi.org/10.18196/jrc.2372

[3] S. Suwarjono, I. H. Wayangkau, T. Istanto, R. Rachmat, M. Marsujitullah, H. Hariyanto, W. Caesarendra, S. Legutko, and A. Glowacz, "Design of Home Fire Detection System Using Arduino and SMS Gateway," *Knowledge*, vol. 1, no. 1, pp. 61-74, 2021, https://doi.org/10.3390/knowledge1010007

[4] K. Gavaskar, D. Malathi, G. Ravivarma, and A. Arulmurugan, "LPG Leakage Detection and Auto Exhaust System using IoT," in *Proc. Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, 2021, pp. 1102-1106.

[5] J. H. Park, S. Lee, S. Yun, H. Kim, and W.-T. Kim, "Dependable Fire Detection System with Multifunctional Artificial Intelligence Framework," *Sensors*, vol. 19, no. 9, p. 2025, 2019, https://doi.org/10.3390/s19092025

[6] A. Raj, A. Viswanathan, and Athults, "LPG Gas Monitoring and Automatic Reservation System," *Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng.*, vol. 3, no. 4, pp. 45-49, 2015.

[7] M. Hima Jyothi, K. Devika, K. K. Tulasi, G. Ramya, and CH. V. Kumar, "Escalated LPG Leakage Detection and Safety Measures," *Int. J. Eng. Res. Technol.*, vol. 9, no. 6, pp. 112-116, 2020.

[8] R. Yauri, J. Smith, and M. Perez, "Evaluation of TinyML Classification Algorithms on ESP32 Microcontrollers for Healthcare Edge Monitoring," *Sensors*, vol. 26, no. 2, p. 411, 2026.

[9] N. Khaya, P. Kumar, and R. Sharma, "SparseBonsai: An Enhanced Resource-Efficient Classifier for Edge Computing in IIoT," *IEEE Trans. Ind. Inform.*, vol. 21, no. 3, pp. 33-42, 2025.

[10] S. Naveen and M. R. Kounte, "Optimized Deep Learning Classification Model for Intelligent Edge Devices," *J. Eng. Sci. Technol. Rev.*, vol. 17, no. 3, pp. 88-94, 2024, https://doi.org/10.25103/jestr.173.11

[11] W. Gao, H. Zhang, and L. Liu, "A Lightweight Intrusion Detection System for Resource-Constrained Edge IoT Networks," *Electronics*, vol. 15, no. 5, p. 938, 2026.

[12] H. Lee, J. Park, and K. Cho, "Systematic Evaluation of Deep Learning Model Optimization Techniques for Edge AI Devices," *Sensors*, vol. 22, no. 19, p. 7344, 2022.

[13] F. Saeed, A. Paul, A. Rehman, W. H. Hong, and H. Seo, "IoT-Based Intelligent Modeling of Smart Home Environment for Fire Prevention and Safety," *J. Sens. Actuator Netw.*, vol. 7, no. 1, p. 11, 2018, https://doi.org/10.3390/jsan7010011

[14] A. Sharma, R. Verma, and P. Gupta, "Integrated Fire Detection Using OpenCV Image Processing and IoT Nodes," *Comput. Electr. Eng.*, vol. 85, p. 106680, 2020.

[15] A. Jadon, M. Omama, and A. Sharma, "FireNet: A Specialized Lightweight Convolutional Neural Network for Real-Time Fire and Smoke Detection," *arXiv preprint* arXiv: 1905.11922, 2019, https://doi.org/10.48550/arXiv.1905.11922

[16] E. O. Amgbara and C. N. Okoronkwo, "Lightweight Machine Learning Classifiers for Personal IoT Defense," *J. Cyber Secur. Technol.*, vol. 8, no. 3, pp. 145-160, 2024.

[17] A. Wahab, M. Khan, and S. Ali, "Fog-Based Host Intrusion Detection System Using N-Gram Feature Matrices," *IEEE Internet Things J.*, vol. 11, no. 8, pp. 13200-13210, 2024.

[18] M. Tanveer, S. Riaz, and A. K. Bashir, "Light Ensemble Guard: Real-Time IoT Attack Detection Using Tree-Based Ensembles," *Comput. Secur.*, vol. 140, p. 103750, 2024.

[19] A. Kumar, S. Goyal, and M. Varma, "Resource-efficient Machine Learning in 2 KB RAM for the Internet of Things," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 1935-1944.

[20] T. Listyorini and R. Rahim, "IoT Fire Detection Prototype Using Fuzzy Logic Controller," *J. Phys. Conf. Ser.*, vol. 1007, p. 012003, 2018.

[21] A. B. Sholanke, E. N. Ekhaese, and P. A. Ekundayo, "Users' Knowledge of Fire Safety Measures in Educational Environment: A Case Study of a College Building in Nigeria," *Int. J. Saf. Secur. Eng.*, vol. 14, no. 1, pp. 135-143, 2024, https://doi.org/10.18280/ijsse.140113

[22] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645-1660, 2013, https://doi.org/10.1016/j.future.2013.01.010

[23] Arduino, "Arduino UNO Q," Product documentation. [Online]. Available: https://docs.arduino.cc/hardware/uno-q

[24] Sunfounder, "IR Flame Sensor," 2023. [Online]. Available: https://docs.sunfounder.com/projects/umsk/en/latest/01_components_basic/03-component_flame.html

[25] Edge Impulse, "Edge Impulse Documentation." [Online]. Available: https://docs.edgeimpulse.com

[26] P. V. A. B. Venâncio, A. C. Lisboa, and A. V. Barbosa, "An Automatic Fire Detection System Based on Deep Convolutional Neural Networks for Low-Power, Resource-Constrained Devices," *Neural Comput. Appl.*, vol. 34, pp. 15349-15368, 2022, https://doi.org/10.1007/s00521-022-07467-z

[27] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Computer Vision – ECCV 2014*, 2014, pp. 740-755, https://doi.org/10.1007/978-3-319-10602-1_48

[28] N. N. Ekedebe, F. C. Ezeh, I. U. Ajere, E. C. Amadi, R. E. Ogu, and N. F. Esomonu, "Design and Experimental Evaluation of an Artificial Intelligence of Things (AIoT)-Enhanced Multi-Modal Safety System," in *Proc. 2026 IEEE Nigeria Int. Conf. Digital Transformation (NIGERCON)*, Lagos, Nigeria, Nov. 2026, in press (DOI to be assigned upon publication in IEEE Xplore).