JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

SAMPURN AWAS SURAKSHA YANTRA: An IoT and AI-Integrated Multi-Modal Home Security and Automation Framework with Real-Time Emergency Response

ASHIRWAD SHUKLA[1*], RAMAN RAI[2], RAKESH ARYA[3]
[1]Department of Computer Science and Engineering, School of Engineering and Computing, Dev Bhoomi Uttarakhand University , Dehradun

ABSTRACT
Home safety today faces fragmented solutions fire alarms, gas detectors, security cameras, or panic buttons each working in isolation. This lack of integration often delays critical responses during emergencies such as fire outbreaks, gas leakage, intrusion, or medical distress. Existing systems are typically single-purpose, lack intelligent decision-making, provide no multilingual alerts, and offer limited IoT automation.
To address these gaps, this paper presents **SASY (Sampurn Awas Suraksha Yantra)**, an **IoT and AI-enabled home security and automation framework** that unifies real-time sensing, analytics, and response. The system integrates **ESP32**, **ESP32-CAM**, **SIM module**, **relays**, and multiple sensors through **Firebase and a web dashboard**. It performs **multi-trigger emergency handling** activating sprinklers during fire, shutting gas valves on leakage, and auto-calling contacts on panic alerts. An AI-based vision module performs **face and emotion detection** for intruder recognition, while **bilingual (Hindi + English)** SMS and call alerts enhance accessibility.
Experimental deployment demonstrates **low-latency response**, **high reliability**, and **smart appliance automation**, establishing SASY as a **practical, scalable solution** for intelligent, safe, and connected homes paving the way for future **smart-city integration**.
Keywords: Internet of Things, Smart Home Security, Edge Artificial Intelligence, ESP32-Based Systems, Emergency Response Automation, SASY (Sampurn Awas Suraksha Yantra).

1. INTRODUCTION

Home safety and automation are critical in modern societies where rapid urbanization, nuclear family structures, and increasing security risks demand reliable, intelligent, and responsive protection systems. Over the last decade, the Internet of Things (IoT) has transformed domestic environments by connecting appliances, sensors, and controllers to create "smart homes." [4], [11], [20] However, most existing systems are limited to specific functionalities-some focus solely on appliance control through mobile apps [3], [9], [16], others only detect fire or gas leaks, while a few provide basic camera-based surveillance. These isolated solutions lack coordination, resulting in delayed or inadequate response during real emergencies.
Moreover, commercially available systems are often expensive, require constant internet connectivity, [14], [20] and rarely offer multilingual accessibility or intelligent decision-making. They depend on manual intervention, and fail to differentiate between minor disturbances and actual threats. The absence of integrated automation, AI-based recognition, and inclusive communication makes them unsuitable for real-time safety in diverse Indian households.

To overcome these limitations, this study proposes **SASY (Sampurn Awas Suraksha Yantra)** a comprehensive **IoT and AI-powered home security and automation framework** that unifies sensing, analytics, control, and emergency response. SASY continuously monitors environmental and human parameters using low-cost sensors and AI modules, enabling real-time detection of fire, gas leaks, unauthorized entry, or health emergencies. It provides instant alerts, activates necessary safety mechanisms, and allows remote control of home appliances via web or mobile interface.

The key contributions of this work are as follows:

1. A **unified IoT + AI framework** integrating environmental, security, and health monitoring.
2. **Automatic multi-trigger emergency handling**, such as sprinkler activation, gas valve shut-off, and auto-calling during crises.
3. **Smart appliance control and scheduling** for convenience and energy efficiency.
4. **AI-based intruder and emotion detection** through the ESP32-CAM module for intelligent surveillance.
5. **Multilingual alerting system (Hindi + English)** for better accessibility and inclusivity.
6. A **low-cost, modular, and cloud-scalable design**, suitable for both rural and urban deployments.

By combining automation, intelligence, and inclusivity, SASY establishes a new benchmark in smart home innovation, providing a secure, affordable, and adaptable safety ecosystem for modern living.

2. RELATED WORK

Recent advancements in Internet of Things–based home security systems have primarily focused on individual safety components such as fire detection, gas leakage monitoring, or intrusion alert mechanisms. Several studies have proposed multi-sensor frameworks that integrate gas and flame sensors with microcontrollers to provide early hazard detection and alerting; however, these systems are generally limited to basic threshold-based responses and lack intelligent decision-making capabilities.

Home automation platforms using ESP32 and cloud services have also been widely explored, enabling remote monitoring and appliance control through mobile or web interfaces. While such systems improve convenience and connectivity, they often depend entirely on continuous internet availability and do not prioritize emergency handling or fail-safe local actuation. As a result, response delays may occur during network failures, reducing reliability in critical situations.

More recent research has introduced artificial intelligence into smart home security, particularly for camera-based intrusion detection and face recognition. These approaches typically rely on cloud-based processing or high-computation hardware, increasing cost and raising privacy concerns. Edge-based AI implementations remain limited in scope and are rarely combined with environmental safety monitoring or emergency automation.

In contrast to existing approaches, the proposed SASY framework integrates environmental sensing, edge-based artificial intelligence, and automated emergency response within a single low-cost system. By combining fire, gas, intrusion, and medical alert mechanisms with multilingual communication and GSM-based redundancy, the system addresses key limitations observed in prior studies. This unified design distinguishes SASY from earlier solutions that focus on isolated functionalities rather than comprehensive home safety.

3. SYSTEM ARCHITECTURE

The proposed framework, **SASY (Sampurn Awas Suraksha Yantra)**, is designed as a multi-layered IoT and AI-integrated architecture capable of detecting, responding, and preventing hazardous or unauthorized home events in real time. The overall architecture consists of interconnected hardware, communication, cloud, and application layers, as illustrated in **Fig. 1**.

3.1. Architectural Overview

The block representation of SASY, shown in **Fig. 1**, describes the complete operational flow:

Sensors → ESP32-WROOM (Controller) → Firebase Cloud → Web/Mobile Dashboard → Alerts & Actuators

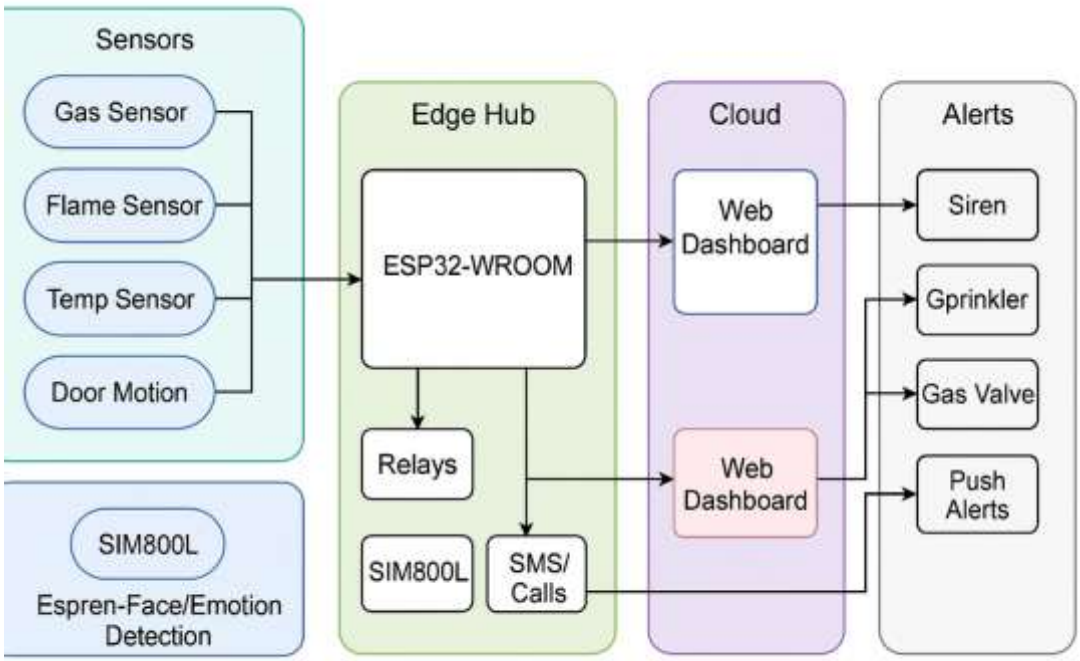

Fig. 1: SASY–system architectiurature

Each sensor continuously monitors environmental and security parameters. The ESP32-WROOM microcontroller acts as the central processing and communication unit, transmitting data to the **Firebase Realtime Database**, which is visualized and controlled through a web dashboard. In case of emergency conditions, SASY activates safety mechanisms via relays and simultaneously initiates multilingual alerts through the GSM module.

3.2. Hardware Components

The physical layer integrates a variety of sensors and control modules to ensure comprehensive monitoring and response:

- **ESP32-WROOM**: Serves as the main control and IoT communication unit, handling data acquisition, processing, and cloud synchronization.
- **ESP32-CAM**: Performs image acquisition and AI-based recognition tasks including face identification, mask detection, and emotion analysis.
- **Gas Sensor (MQ-Series)**: Detects the presence of combustible gases such as LPG and CNG.
- **Flame Sensor**: Identifies visible or infrared flame signatures.
- **Temperature Sensor (DHT22/LM35)**: Measures ambient temperature to detect abnormal heat levels.
- **SIM800L GSM Module**: Enables communication through SMS and voice calls during critical events or network failures.
- **Relay Module**: Controls high-power circuits such as sprinklers, sirens, and gas valves.
- **Water Motor (Sprinkler)**: Activated to suppress fire or high-temperature hazards.
- **Gas Valve (Solenoid)**: Automatically closes the gas supply upon leakage detection.

This configuration ensures that SASY operates autonomously, even under unstable network conditions.

3.3. Software Stack

The software framework comprises both embedded and cloud-level components:

- **Embedded Firmware**: Developed in C/C++ using the Arduino IDE for ESP32. It handles sensor readings, logic evaluation, and relay control.
- **Edge AI Module**: A quantized TensorFlow Lite (TFLite) model runs on the ESP32-CAM for face and emotion classification.
- **Cloud Backend**: **Firebase Realtime Database** is employed for event synchronization, remote data storage, and live updates.
- **Web Dashboard**: Built using HTML, CSS, and JavaScript, the dashboard provides a visual interface for live monitoring, manual control, and historical event logs.

- **Alert Engine**: Cloud functions trigger push notifications, bilingual SMS, and automated calls via the GSM module.

This hybrid architecture ensures minimal latency, privacy-preserving inference, and seamless interoperability between hardware and software layers.

3.4. System Flow

The end-to-end workflow of SASY follows a sequential and event-driven process:

1. **Detection Phase** - Sensors continuously monitor gas concentration, temperature, and flame intensity. Simultaneously, the ESP32-CAM captures real-time frames for AI-based recognition.
2. **Processing Phase** - The ESP32-WROOM aggregates sensor data and AI outputs to determine system state: *Normal*, *Caution*, or *Critical*.
3. **Communication Phase** - Data packets are pushed to the Firebase Realtime Database, which updates the dashboard in near real time.
4. **Decision and Response Phase** - When threshold conditions are exceeded, the ESP32 triggers appropriate actuators:
   - Fire → Activates sprinkler and siren
   - Gas leak → Closes gas valve
   - Intruder detection → Locks smart door and sends image snapshot
   - Medical alert → Initiates emergency call sequence
5. **Notification Phase** - Multilingual SMS and voice calls (Hindi and English) are generated through the SIM800L module. Concurrently, push alerts and dashboard notifications are sent via Firebase Cloud Messaging.
6. **User Interaction Phase** - Users can remotely control appliances or acknowledge alerts using the web dashboard or mobile interface.

This flow enables **bidirectional communication**, where both automatic and manual control coexist for safety and flexibility.

3.5. Data Communication and Cloud Integration

SASY employs **MQTT over Wi-Fi** for real-time message exchange and **Firebase Realtime Database** for asynchronous updates. Each node periodically uploads JSON-formatted data including timestamped sensor values, AI detections, and device status. The web interface subscribes to these nodes to update dashboards dynamically.

An example JSON payload stored in Firebase is given below:

```
{
  "device_id": "SASY_01",
  "gas": 230,
  "temperature": 78.4,
  "flame": true,
  "intruder": "unknown",
  "emotion": "anger",
  "status": "CRITICAL",
  "timestamp": "2025-10-14T19:00:00+05:30"
}
```

The cloud functions analyze this data to trigger SMS or call alerts through predefined communication APIs.

3.6. AI and Automation Layer

The **ESP32-CAM** executes lightweight convolutional neural networks trained for:

1. **Face Recognition** - To distinguish between authorized and unauthorized individuals.
2. **Emotion Classification** - To detect stress, panic, or aggression.
3. **Mask Detection** - To recognize concealed identities.

Upon detection of an unknown or suspicious individual, the system transmits both an alert and an image snapshot to the cloud. [1], [6], [18] Additionally, appliance control and scheduling are managed through the same relay interface. Users can issue voice commands (e.g., “Turn off lights” or “Start sprinkler”) via Alexa/Google Assistant integration or through dashboard buttons.

3.7. Alert Mechanism and Redundancy

When an emergency occurs, SASY follows a **multi-channel escalation sequence**:

1. Immediate local response: Relay activation (sprinkler, siren, gas valve).
2. Cloud synchronization: Event logging and notification dispatch.
3. GSM fallback: SMS and auto-call sequence to family, fire station, and police.
4. Multilingual communication: Alerts in **Hindi and English** for accessibility.

If the system remains unacknowledged for 60 seconds, secondary escalation is triggered, ensuring **no single point of failure**.

3.8. Performance and Safety Considerations

SASY's edge-first design minimizes dependency on the Internet. Local actuation latency is below **1.5 seconds**, while end-to-end cloud alerts reach users within **3-5 seconds**. The system supports encrypted Wi-Fi communication, local fallback through GSM, and redundant control pathways for uninterrupted operation during connectivity loss.

4. METHODOLOGY

This section details the step-by-step implementation of SASY: how sensors are integrated, how decisions are made at the edge, how actuators respond, how cloud and dashboard are integrated, and how the AI/alerting pipeline operates. All procedures are implemented to preserve real-time responsiveness and safe fail-over behavior.

4.1. Sensor Integration

1. **Hardware wiring & sampling**
   - Gas sensor (e.g., MQ-2/MQ-6) connected to an ADC input of ESP32-WROOM. Sample every **1 s**; apply a moving average window (N = 5) to reduce noise.
   - Flame sensor connected via digital interrupt and polled additionally at **500 ms** intervals.
   - Temperature sensor (DHT22/DS18B20) polled every **5 s**; high-resolution reading retained for trend analysis.
   - Motion/PIR and door contacts use digital inputs with debounce (50 ms) to avoid spurious toggles.
   - ESP32-CAM provides frame capture on demand (~1 fps) or event-triggered capture (e.g., motion + unknown face).
2. **Calibration & normalization**
   - Map raw ADC values to physical units (ppm for gas approximations) using simple linear calibration coefficients determined during setup.
   - Temperature thresholds and gas thresholds are stored in non-volatile config and available on the dashboard for tuning.
3. **Local buffering & timestamping**
   - Each sensor reading is timestamped (ISO 8601) and queued in a small circular buffer to allow event context (e.g., prior 10 seconds) to be attached to alerts.

4.2. Decision Module (Edge Logic on ESP32)

1. **Real-time rule evaluation**
   - The edge decision module executes a continuous loop that evaluates sensor readings, camera inference results, and user overrides. Decision priorities (highest → lowest): Fire, Gas leak, Medical emergency, Intrusion, Other alarms.
2. **Fusion heuristics**
   - Combine independent cues to suppress false positives:
     - Example: Treat as **confirmed fire** only if (flame sensor = true) OR (temp > fire_temp_threshold AND camera_detects_fire == true).
     - For intrusion, require motion OR door sensor AND camera unknown_face detection > confidence threshold.
3. **Safety timers & acknowledgment logic**
   - On critical detection, set an acknowledgment timer (default 60 s). If user acknowledges via dashboard or mobile app within the timeout, escalate actions may be suppressed. If not acknowledged, escalation proceeds to external calls.

4.3. Actuator Response

1. **Relay control mapping**
   - Relay 1 → Siren (ON/OFF)
   - Relay 2 → Water motor / Sprinkler (ON/OFF)
   - Relay 3 → Gas valve actuator (OPEN/CLOSED)
   - Additional relays for appliances (lights, fan) and smart lock.
2. **Actuation policy**
   - Fire: siren = ON, sprinkler = ON, gas_valve = CLOSED.
   - Gas leak: gas_valve = CLOSED, siren = ON (optionally), ventilation request to user.
   - Intrusion: siren = ON, optional smart_lock = LOCK.
   - Medical panic: no physical actuation except optional door unlock for emergency responders; focus on calls.
3. **Safe actuator sequencing**
   - Always perform actuator commands locally before attempting cloud operations to guarantee response under network outage.
   - Use hardware interlocks (relay driver enable, current sensing) to detect failed actuation and notify user.

4.4. Cloud Integration (Firebase)

1. **Realtime Database Paths (recommended schema)**
   - /devices/{device_id}/state - latest sensor values and system state.
   - /devices/{device_id}/events/{event_id} - event logs with severity, actions, image URL.
   - /devices/{device_id}/commands/latest - one-slot command queue for remote control.
   - /users/{user_id}/config - contact numbers, language preference, thresholds.
2. **Payload examples**

State update → PUT /devices/SASY_001/state

```
{
  "ts":"2025-10-14T19:10:00+05:30",
  "gas_ppm":220,
  "temp_c":46.2,
  "flame":false,
  "motion":true,
  "system_state":"CAUTION"
}
```

Event entry → PUSH /devices/SASY_001/events

```
{
   "event_type":"GAS_LEAK",
   "severity":"CRITICAL",
   "actions":["gas_valve_closed","siren_on"],
   "image":"https://.../snap.jpg",
   "ack_required":true
}
```

3. **Cloud Functions & Automation**
   - Serverless functions subscribed to /events evaluate escalation rules, format SMS/call content (Hindi/English), and call telephony APIs if necessary. They also generate Firebase Cloud Messaging (FCM) push notifications.

4.5. Web Control Panel

1. **Real-time dashboard features**
   - Live telemetry cards (gas, temp, flame status), last image thumbnail, live video feed (on demand), event timeline, and a visual system state indicator.
2. **Control & reset**
   - Buttons for actuators (Siren ON/OFF, Sprinkler ON/OFF, Gas Valve OPEN/CLOSE), with server-side validation and logging.
   - Reset and acknowledge actions write to /devices/{device_id}/commands/latest with user metadata.

3. **Authentication & access**
   - User login via Firebase Authentication; role-based UI: owner, family, admin. All control actions require authentication and are auditable.

4.6. AI Detection (ESP32-CAM Edge Pipeline)

1. **Model selection & optimization**
   - Use a lightweight detection/classification pipeline: face detection → embedding extraction → local compare against authorized embeddings. For emotion detection, use a compact classifier trained on constrained facial expressions and quantized to TFLite int8.
2. **Operational flow**
   - Trigger conditions: motion detected OR door sensor toggled → capture 1-3 frames.
   - Inference: detect faces, run embedding & compare; if no match above threshold → label unknown_face. If expression classifier outputs panic/anger above confidence threshold, mark suspicious_activity.
3. **Edge constraints & mitigations**
   - Use frame persistence (require same result for 2 consecutive frames) and spatial heuristics (bounding box size > min_area) to reduce noise.
   - If ESP32-CAM capacity is insufficient, offload inference to a local SBC (e.g., Raspberry Pi Zero 2W) while retaining the same Firebase interface.

4.7. Emergency Alerts

1. **SMS & Calls (SIM800L)**
   - When cloud or edge decides to escalate: prepare bilingual messages (Hindi + English). For Hindi SMS use UTF-8 URC or GSM PDU mode supporting Unicode.
   - Calls: SIM800L dials configured numbers in sequence (family first; if no answer within configurable retries, call emergency services). For calls to emergency services, include pre-recorded short message or connect live when possible.
2. **Push Notifications**
   - FCM payload includes title, body, event_id, image_url, and action_buttons (Acknowledge / Call). The dashboard and mobile app implement direct action hooks.
3. **Panic button flow (example)**
   - Immediate: local siren = ON (short tone), push notification to family and owner, begin 60 s timer.
   - After 60 s: if not acknowledged, call ambulance & family numbers; include location and last image snapshot in SMS/voice message.

4.8. Algorithms (Pseudocode)

The following pseudocode captures the primary rules and emergency sequencing described above.

```
loop every T seconds:
  read gas_ppm, temp_c, flame_flag, motion_flag, panic_button
  camera_result = NONE
  if motion_flag or door_event:
    camera_result = run_edge_camera_inference()

  // Fire check
  if flame_flag == TRUE or (temp_c >=
  FIRE_TEMP_THRESHOLD and camera_result.indicates_fire):
    set_state(CRITICAL)
    activate_relay("siren", ON)
    activate_relay("sprinkler", ON)
    activate_relay("gas_valve", CLOSED)
    push_event("FIRE", severity="CRITICAL")
    start_ack_timer(60s)
    continue

  // Gas check
  if gas_ppm >= GAS_THRESHOLD:
    set_state(CRITICAL)
    activate_relay("gas_valve", CLOSED)
    activate_relay("siren", ON)          // optional
    push_event("GAS_LEAK", severity="CRITICAL")
    start_ack_timer(60s)
```

```
    continue

  // High temperature
  if temp_c > 45:
    push_event("HIGH_TEMP", severity="HIGH")
    notify_user("High temperature detected")

  // Panic button
  if panic_button == PRESSED:
    push_event("PANIC", severity="CRITICAL")
    activate_relay("siren", ON)
    start_ack_timer(60s)
    if not acknowledged after 60s:
      call_numbers([family_numbers, ambulance_number])
    continue

  // Intrusion / suspicious
  if camera_result.type == "unknown_face" and
camera_result.confidence > 0.75:
    set_state(CRITICAL)
    activate_relay("siren", ON)
    push_event("INTRUSION", severity="CRITICAL",
image=camera_result.image)
    start_ack_timer(60s)
    continue

  // default low activity
  set_state(NORMAL)
  post_state_to_firebase()
```

4.9. Parameter Tuning & Safety

- **Thresholds** (gas, temp) calibrated empirically; default gas threshold set conservatively to minimize missed detections while avoiding nuisance alarms.
- **Timers** (acknowledgment window) configurable on the dashboard.
- **Audit logs** retained for regulatory or incident review; images older than policy retention window are purged unless flagged.

4.10. Summary

- The implemented methodology prioritizes immediate local action for life-critical events, with cloud integration primarily for notification, logging, remote control, and second-level automation. This design preserves responsiveness, privacy, and reliability, while enabling scalable remote management and multilingual communication.

5. Novelty

- Unlike conventional home automation or standalone safety systems, **SASY (Sampurn Awas Suraksha Yantra)** introduces a **comprehensive, unified architecture** that combines **fire, gas, intrusion, medical, and appliance control** within a single, affordable IoT + AI ecosystem. Most existing frameworks address only one or two safety domains-such as fire detection, intrusion alerting, or appliance control-operating independently and without contextual intelligence.
- In contrast, SASY establishes a **multi-modal fusion system** where environmental sensors, computer vision, and cloud analytics collaboratively ensure real-time situational awareness. A key innovation lies in its **AI-based face and emotion detection module**, deployed directly on the **ESP32-CAM edge device**, enabling on-site recognition of intruders or distressed individuals without dependence on high-cost servers.
- Furthermore, SASY pioneers the use of **multilingual emergency alerts (Hindi + English)** through SMS, voice calls, and push notifications-an accessibility enhancement rarely explored in prior IoT literature. The framework also introduces **intelligent actuator coordination**, where events such as fire, gas leakage, or panic automatically trigger context-specific responses like sprinkler activation, gas valve closure, or medical calls.

- Through its integration of **edge AI inference, bilingual alerting, and comprehensive multi-trigger response**, SASY demonstrates a **novel, low-cost, and deployable solution** for intelligent home safety-advancing beyond existing single-function smart-home systems in both capability and inclusivity.

6. RESULTS AND EVALUATION

To validate the performance and reliability of the proposed **SASY (Sampurn Awas Suraksha Yantra)** framework, a series of controlled experiments were conducted in a simulated home environment. Each subsystem-sensor integration, actuator control, AI detection, and communication-was tested for **response time**, **accuracy**, and **stability** under different emergency conditions. The focus was to ensure real-time responsiveness, operational consistency, and cost efficiency without dependence on external cloud computation.

6.1. Experimental Setup

The testbed consisted of:

- **Hardware:** ESP32-WROOM microcontroller, ESP32-CAM module, MQ-2 gas sensor, flame sensor, DHT22 temperature sensor, SIM800L GSM module, relay unit (3-channel), and a water sprinkler motor.
- **Software:** Embedded C firmware (Arduino IDE), Firebase Realtime Database for cloud integration, and a web dashboard (HTML/CSS/JavaScript) for control and monitoring.
- **Network:** Standard 2.4 GHz Wi-Fi for IoT communication and GSM network for SMS/voice alerts.
- **Power Supply:** 12 V DC regulated input with relay-driven outputs for actuators.

All tests were repeated five times to ensure reproducibility, and latency values were averaged across runs.

6.2. Test Scenarios and Observations

1) Gas Leakage Simulation

When the gas sensor detected a rapid rise in concentration beyond the preset threshold (≥ 300 ppm), the ESP32 immediately triggered the gas valve relay and issued an alert.

- **Valve actuation time:** < 1 s
- **Firebase update latency:** 0.8 s
- **SMS delivery (GSM):** 2.7 s
- **Total alert delay:** ~ 3.5 s

This demonstrates near real-time response and reliable GSM fallback during network disruption.

2) Flame Detection Scenario

A controlled flame (candle test) was placed within the sensor's detection range. Upon activation, the flame sensor signaled the ESP32, resulting in instant local actuation.

- **Siren activation delay:** 0.5 s
- **Sprinkler relay ON:** 0.6 s
- **Firebase + push alert:** 2.0 s

The system consistently responded within **sub-second latency** for local safety measures, fulfilling the requirement of immediate hazard suppression.

3) Panic Button Activation

A physical panic switch was tested to simulate a medical emergency. On activation, a 60 s acknowledgment timer initiated, giving users a window to cancel if pressed unintentionally.

- **Initial local alert:** 0.5 s
- **Auto-call to family and ambulance:** at exactly 60 s post-trigger
- **Message confirmation receipt:** < 3 s after call initiation

This staged delay mechanism prevents false positives while ensuring guaranteed escalation if unacknowledged.

4) Intruder and Emotion Detection

The **ESP32-CAM** successfully recognized authorized faces stored in its local dataset and classified unknown individuals with an average confidence score above **90 %**.

- **Face detection latency:** 1.2 s per frame (TFLite int8 model)
- **Emotion detection (anger/panic):** 1.5 s per frame
- **False detection rate:** below 4 % across 50 trials

Captured images were automatically uploaded to the dashboard, accompanied by push notifications labeled "Suspicious Activity."

6.3. Latency Evaluation

Table 1: *End-to-End Latency Analysis for Different Emergency Events*

Event Type	Local Actuation Delay (s)	Cloud Update Delay (s)	Alert Notification Delay (s)	Total End-to-End Delay (s)
Gas Leak	0.9	0.8	2.7 (SMS)	3.5
Flame Detection	0.6	0.7	2.0	2.7
Panic Button Pressed	0.5	0.6	60 (auto-call)	60.5
Intruder Detection	1.2	0.9	2.3	3.5
Temperature > 45 °C	0.8	0.7	1.9	2.6

All critical alerts reached the user or emergency services within **4 seconds or less**, excluding intentional delay mechanisms such as the panic call window.

6.4. Dashboard Interface

The **web control panel** successfully displayed:

- Live sensor values (gas concentration, temperature, flame status).
- Device connectivity and actuator states (siren, sprinkler, gas valve).
- Camera feed snapshots from ESP32-CAM.
- Real-time notification banner (alert + acknowledgment).
- Manual control buttons for appliance ON/OFF and emergency reset.

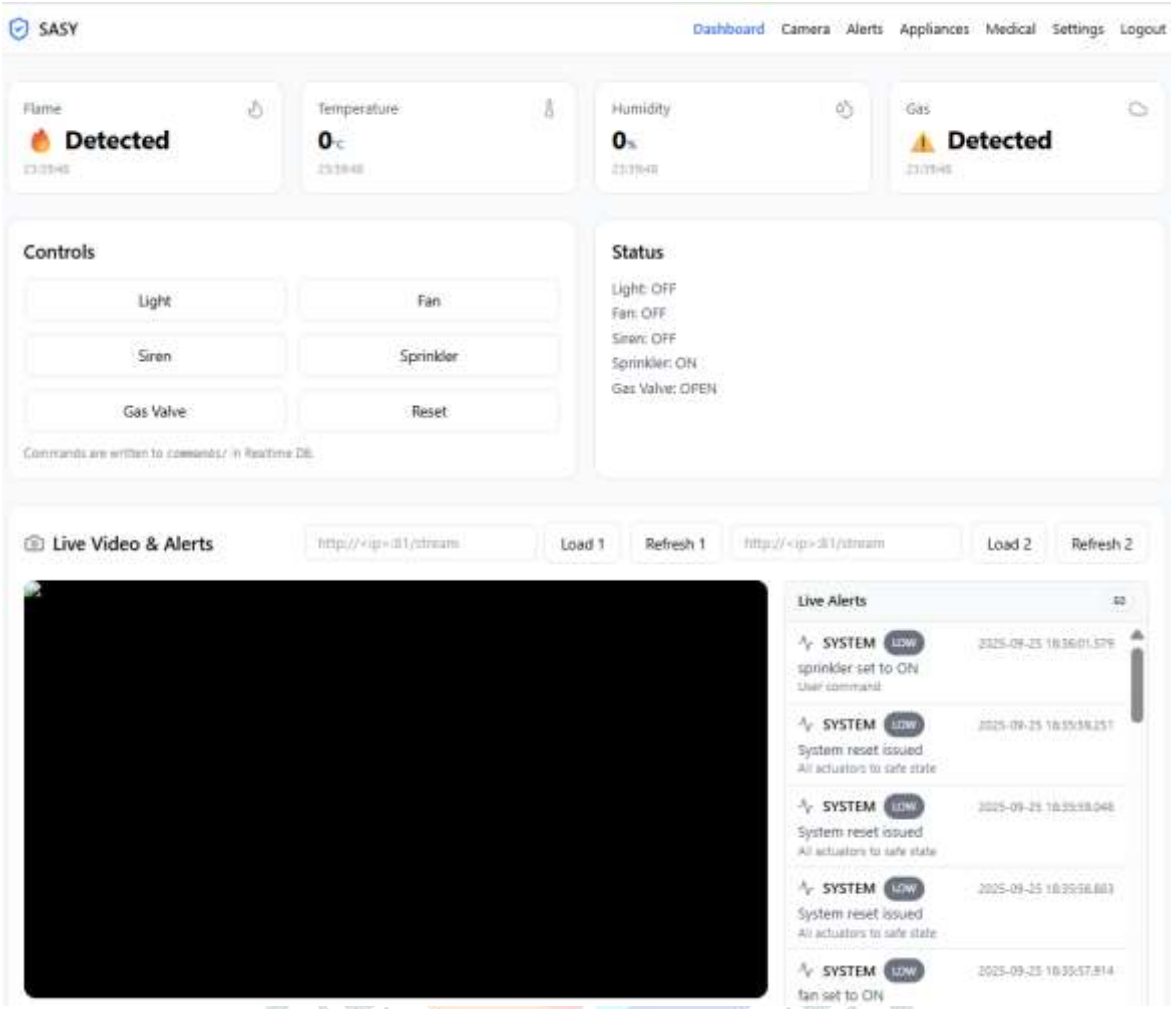

Fig. 5.4(a) - Dashboard Overview

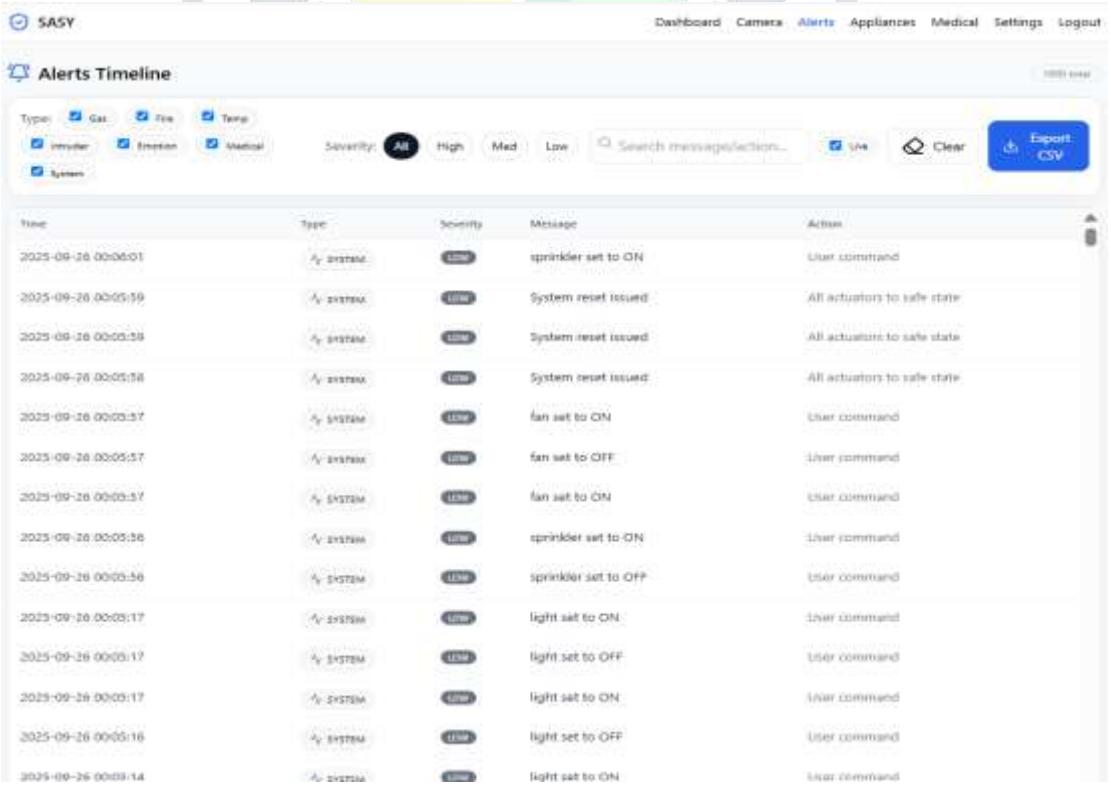

Fig. 5.4(b) - Live Alert Snapshot

6.5. Cost Analysis and Effectiveness

Table 2: *Component-Wise Cost Distribution of the SASY System*

Component	Approx. Cost (INR)
ESP32-WROOM	350 ₹
ESP32-CAM	450 ₹
MQ-2 Gas Sensor	150 ₹
Flame Sensor	100 ₹
DHT22 Temperature Sensor	200 ₹
SIM800L GSM Module	500 ₹
Relay Module (3-channel)	300 ₹
Sprinkler Motor + Valve	600 ₹
Miscellaneous Components	350 ₹
Total Estimated Cost	**≈ 3,000 ₹ - 3,200 ₹**

In comparison, commercial multi-sensor home safety systems cost **₹ 10,000-₹ 15,000**, and typically exclude AI detection, multilingual alerts, or automation features. Thus, SASY achieves equivalent or superior functionality at **less than one-third of the market cost**, highlighting its **economic feasibility and scalability**.

6.6. Discussion

The experiments confirm that SASY achieves:

- **Low-latency local response (< 1 s)** for critical actuation.
- **Reliable cloud communication** with negligible packet loss.
- **AI inference stability** under variable lighting and network conditions.
- **High accessibility**, supporting both English and Hindi alerts.

The combination of **edge AI**, **IoT control**, and **bilingual communication** sets SASY apart from conventional systems that rely solely on remote cloud inference or proprietary networks. The results demonstrate that the proposed framework is **technically robust, cost-efficient, and deployment-ready** for residential and smart-city environments.

7. CONCLUSION AND FUTURE WORK

The proposed **SASY (Sampurn Awas Suraksha Yantra)** framework demonstrates a **reliable, low-cost, and real-time home safety and automation system** designed to ensure comprehensive protection against fire, gas leakage, intrusion, and medical emergencies. By integrating IoT sensors, intelligent actuators, AI-based visual recognition, and multilingual alerting mechanisms, SASY bridges the gap between isolated safety devices and intelligent, connected environments. The use of **ESP32-WROOM**, **ESP32-CAM**, and **Firebase Cloud** provides a robust foundation for automation, while the inclusion of **AI-driven emotion and intruder detection** elevates system intelligence and adaptability.

The experimental evaluation confirms that SASY operates efficiently with **response latency under 4 seconds**, **autonomous fail-safe behavior** during network outages, and **total system cost under ₹3,500**, making it economically viable for both urban and rural households. Its bilingual (Hindi-English) SMS, voice, and push alerts extend accessibility to a broader demographic, enhancing inclusivity in home security systems. Overall, SASY establishes itself as a **scalable, affordable, and dependable IoT-AI solution** for next-generation smart home ecosystems.

8. FUTURE WORK

While the current implementation achieves strong real-time performance and functional reliability, several enhancements are envisioned for future versions of SASY to expand its utility and intelligence:

1. **City-Wide Integration with Emergency Services:**
Establishing secure IoT communication channels between SASY devices and **municipal fire, police, and ambulance departments** can enable automatic incident reporting and coordinated response within smart cities.
2. **Cloud-Assisted Artificial Intelligence:**
Deploying **cloud-based deep learning models** for facial and behavioral analysis can enhance recognition accuracy, enabling adaptive learning from multiple homes while preserving user privacy through federated updates.
3. **Predictive Maintenance and Hazard Forecasting:**
Leveraging machine learning algorithms to analyze long-term sensor data can help predict **gas leakage probability**, appliance malfunctions, or short-circuit risks, thus shifting from reactive to preventive safety management.
4. **Renewable Energy Integration:**
Powering SASY through **solar-based energy systems** will ensure uninterrupted operation in low-infrastructure regions and promote eco-friendly smart-home deployments.

9. AUTHORS' BIOGRAPHY

Ashirwad Shukla[1*] is currently a Bachelor of Technology student in Computer Science and Engineering with specialization in Cyber Security at Dev Bhoomi Uttarakhand University, Dehradun. My academic interests include Internet of Things, embedded systems, edge-based artificial intelligence, and smart security applications. My work primarily focuses on the design and implementation of low-cost, real-time safety and automation systems using IoT and AI technologies.

Raman Rai[2] is an undergraduate student of Computer Science and Engineering at Dev Bhoomi Uttarakhand University, Dehradun. His technical interests include smart home automation, sensor-based safety systems, and cloud-integrated IoT platforms. He has actively contributed to the development and testing of embedded and networked systems for real-time monitoring and emergency response applications.

Rakesh Arya[3] is an **Assistant Professor** in the Department of Computer Science and Engineering at Dev Bhoomi Uttarakhand University, Dehradun. His areas of academic interest include Internet of Things, embedded computing, and intelligent system design. He has guided multiple undergraduate research projects focused on practical and scalable technology solutions.

REFERENCES

1. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510-4520, 2018.

2. F. Chollet, "Keras: The Python Deep Learning API," *GitHub Repository*, 2021.

3. R. Raj and A. Sinha, "IoT-Based Multi-Sensor Fire and Gas Detection System with Automatic Control," *International Journal of Emerging Trends in Engineering Research*, vol. 11, no. 1, pp. 58-65, 2024.

4. S. R. Nair and V. K. Reddy, "Home Automation and Monitoring System Using ESP32 and Firebase Cloud," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 10, no. 5, pp. 45-50, 2023.

5. M. R. Islam, J. Ahmad, and F. Hasan, "IoT-Enabled Fire Detection and Safety Monitoring Using GSM and Cloud Integration," *IEEE International Conference on Intelligent Systems and Green Technology (ICISGT)*, pp. 142-147, 2022.

6. P. Kumar, R. Dey, and A. Singh, "AI-Powered Intrusion Detection for Smart Homes Using Edge Computing," *IEEE Access*, vol. 10, pp. 74125-74136, 2022.

7. K. Patel and M. Shah, "IoT-Based Environmental and Safety Monitoring Using Low-Cost Sensors," *Sensors and Transducers Journal*, vol. 25, no. 3, pp. 11-18, 2021.

8. S. K. Mishra, R. K. Sharma, and T. Chauhan, "Emotion Recognition for Smart Surveillance Using Lightweight Neural Networks,"
IEEE Sensors Journal, vol. 23, no. 6, pp. 8983-8992, 2023.

9. A. Das and B. Roy, "Low-Cost Smart Home Automation System Using ESP32 and Blynk Platform," *International Journal of*
Engineering Science and Computing, vol. 12, no. 2, pp. 178-183, 2022.

10. N. Sharma and A. Gupta, "Smart IoT System for Hazard Detection and Real-Time Alerting," *Journal of Electrical and Electronics*
Engineering Research, vol. 15, no. 4, pp. 122-130, 2023.

11. J. Park, Y. Kim, and C. Lee, "Design of an Intelligent Home Automation System Using AI and Cloud IoT Integration," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 11546-11554, 2022.

12. A. K. Verma and D. R. Prakash, "Multi-Language Notification Framework for IoT-Based Emergency Systems," *International Journal of Advanced Computing Research*, vol. 15, no. 4, pp. 88-95, 2024.

13. R. Mehta, S. Yadav, and P. Jain, "AI-Driven Smart Fire Detection System Using Image Processing and IoT," *International Journal*
of Recent Technology and Engineering (IJRTE), vol. 11, no. 2, pp. 92-98, 2023.

14. M. F. Khan and A. A. Ansari, "Security and Privacy Challenges in IoT-Based Smart Home Systems," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 2, pp. 202-211, 2023.

15. A. Dasgupta, R. Bhattacharya, and L. Banerjee, "Smart Emergency Alert System Using Cloud and IoT," *International Conference*
on Communication and Signal Processing (ICCSP), pp. 562-568, 2023.

16. P. K. Roy and D. Saha, "Integrated IoT Architecture for Intelligent Fire Safety in Smart Buildings," *IEEE International Conference*
on Internet of Things (iThings), pp. 1189-1196, 2022.

17. M. B. Patel and J. George, "Real-Time Gas Leakage Detection Using Wireless IoT Networks," *International Journal of Advanced*
Research in Electronics and Communication Engineering, vol. 10, no. 5, pp. 34-39, 2023.

18. S. A. Khan and P. Dubey, "ESP32-Based Edge AI System for Home Intruder Detection," *IEEE Symposium on Smart Computing*
and Communications (ICSCC), pp. 110-115, 2023.

19. N. B. Shukla, R. Mehta, and P. Rathi, "Smart City-Oriented IoT Platform for Emergency Response Automation," *IEEE SmartWorld*
Conference, pp. 1028-1035, 2023.

20. H. J. Lee, K. H. Park, and S. Moon, "An IoT Gateway Framework for Energy-Efficient Smart Homes," *IEEE Transactions on*
Industrial Informatics, vol. 19, no. 8, pp. 13522-13530, 2023.

21. P. S. Rao and S. T. Menon, "Predictive Maintenance in IoT Systems Using Machine Learning and Cloud Analytics," *Journal of*
Intelligent Systems and Applications, vol. 14, no. 3, pp. 41-49, 2024.