Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

A Supervised Intrusion Detection System for Smart Home IoT Devices

Dr. Ch. Balaswamy
Professor;
Department of Electronics and Communication Engineering
Seshadri Rao Gudlavalleru Engineering College, Andhra Pradesh 521356, India

Kodavalla Devi Dileep Kumar,
Kokkiripati Hema Varsha,
Kokkera Pardhu Sree, Labba Rajesh
U.G Students
Department of Electronics and Communication Engineering
Seshadri Rao Gudlavalleru Engineering College, Andhra Pradesh 521356, India

Abstract: - The project aims to develop an advanced IoT-based Smart Home Security and Fire Safety System utilizing an ESP32 microcontroller along with the Blynk IoT platform. This IoT-based system includes dual ultrasonic sensors for intrusion detection. Moreover, MQ series sensors are integrated for detecting smoke and fire hazards. To avoid delays in such scenarios, an optimized Python-based computer vision supervisor has been implemented. This enables near-instant image capture and transmission. For enhanced security, a Telegram Bot API has been integrated. Once a hazard occurs, audiovisual alarms are activated. Along with this, a visual image of the event is displayed to the user. This provides a robust solution for residential security, considering its low cost and high performance.

Index Terms **– IoT Security; ESP32; Computer Vision; Smart Home Automation; Real-time Monitoring; Python-based Supervision; Intrusion Detection System; Telegram Bot API; Multi-threaded Processing; Low-latency Systems.**

I. INTRODUCTION

The ever-evolving dynamics of the Internet of Things (IoT) technology have completely transformed the way residential security systems operate today, from localized alarm systems to integrated surveillance systems. However, a major concern in today's cost- effective security systems is the delay in triggering the camera system after detecting the intrusion. Most residential systems are likely to be in a "cold start" delay in which the camera starts only after the detection of the intrusion, resulting in a lack of clear images of the perpetrator.

This project attempts to overcome these hardware and software challenges in the system by introducing an Autonomous Supervised Intrusion Detection System. The system uses an ESP32-based edge computing system for perimeter surveillance along with a high- speed Python-based supervisor system that captures images in a near-instantaneous manner using a "warm start" buffer system in multithreading.

The system has been developed to ensure multi-hazard protection, which includes dual ultrasonic sensors for spatial intrusion detection and MQ series sensors for fire/smoke detection. Once an intrusion occurs, the ESP32 module initiates a localized audiovisual alarm response, along with a handshake signal to trigger a Python supervisor. This initiates a visual verification response, which captures images of the intruder and transmits them to the user's smartphone via a Telegram Bot API and Blynk IoT platform.

Moreover, this research has highlighted the importance of a reliable system, which has been implemented through an autonomous boot protocol. This enables the security monitor to self-initialize when a system restart occurs, ensuring a security state without any human intervention. This provides a robust, cost-effective, and highly responsive solution for smart home environments.

II. PROPOSED SYSTEM

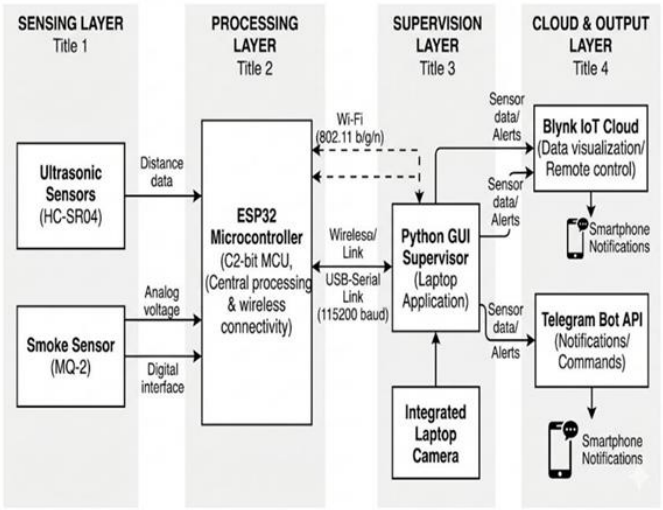
The proposed system architecture utilizes the ESP32 microcontroller in conjunction with a high-speed Python-based layer to provide a responsive multi-hazard security solution. Perimeter security is provided by a set of HC-SR04 ultrasonic sensors and MQ series gas sensors, which provide real-time edge-level processing to detect potential security breaches or fire hazards. Once a critical level has been reached, the ESP32 microcontroller sends a local audiovisual alarm and sends a high-priority "CAPTURE" message to a laptop via a 115200 baud serial handshake. To avoid the standard hardware initialization delay of 2 seconds, the Python layer maintains a persistent "warm start" camera buffer in a background thread to provide near-instantaneous image capture. Once the image has been captured, it is cleared of buffer lag via specialized frame-grabbing logic and then sent to the user's smartphone via the Telegram Bot API and Blynk IoT platform. This multi-threaded approach ensures that high-resolution image verification and cloud-based alerting occur in parallel to the real-time image capture and transmission, providing a robust and autonomous solution for smart home security.

In addition to the immediate response, the system also

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

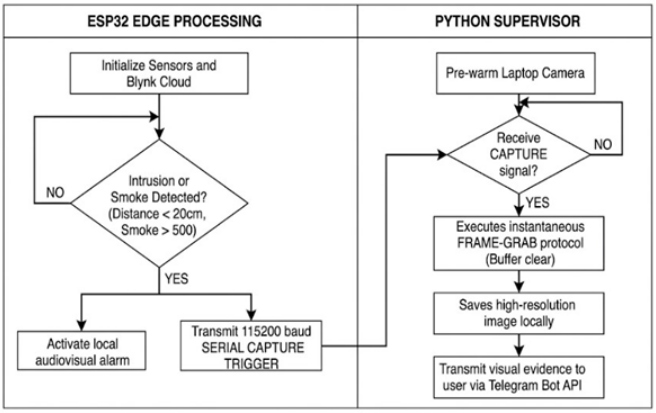
includes an autonomous boot protocol to guarantee the maintenance of the security state without the need for human interaction after system restarts. The autonomous boot protocol, combined with the 115,200 baud rate high-speed serial handshake, enables the total system response to be approximately 1.16 seconds from the initial sensor trigger to the provision of visual evidence. By moving computationally expensive image processing to the Python supervisor and edge-level sensing to the ESP32, the system offers a cost-effective and highly responsive solution to traditional commercial surveillance systems

III. IMPLEMENTATION DIAGRAM



The proposed system is achieved through a four-layer IoT system: sensing, edge processing, supervision, and cloud output. The system includes real-time perimeter and fire detection using dual HC-SR04 ultrasonic and MQ-2 smoke sensors connected to an ESP32 microcontroller for real-time detection. The ESP32 sends a 115200 baud serial handshake signal to a Python-based GUI supervisor for high-speed verification. The supervisor leverages a "warm start" laptop camera for instant high-resolution image capture in the presence of hazards. The system sends automatic alarms and evidence to the user's smartphone using the Blynk IoT cloud and Telegram Bot API.

IV. FLOW OF PROJECT



The software programming adheres to a two-layered flowchart, distinguishing between edge processing by the ESP32 and high- speed Python supervision. The ESP32 constantly polls the sensors for intrusion or smoke detection. Once a threshold breach is detected, a local alarm is sounded while sending a serial signal at a baud rate of 115200 to the laptop. The Python supervisor, by utilizing a "pre-warmed" camera buffer, receives the serial signal and implements an instantaneous frame grab protocol to avoid hardware latency. The captured high-resolution image is then saved locally and sent to the user through the Telegram Bot API for remote verification.

V. HARDWARE MODULES

A. ESP32:

ESP32 is a low-power microcontroller used in IoT projects. This board contains Wi-Fi and Bluetooth modules, which enable wireless communication. This board contains several GPIO pins. This board supports UART, SPI, I2C, and PWM communication. In our project, we use ESP32 as a main board. This board receives commands from a voice assistant or a mobile app. This board controls appliances like lights, fans, and air conditioners using relay modules.

Fig.1: ESP32 Board

B. Ultrasonic Sensor (HC-SR04):

The HC-SR04 is a low-cost sensor that uses sonar to measure distance to an object. It does this by sending out an ultra sonic trigger pulse at 40 kHz and then measuring the time it takes to return from the object. In our project, two such sensors are employed to measure a spatial threshold (for example, 20 cm), and this is our source for intrusion detection.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

Fig.2: Ultrasonic Sensor (HC-SR04)

C. Smoke/Gas Sensor (MQ-2):

The MQ-2 is a metal oxide semiconductor (MOS) gas sensor, and it is used to sense various combustible gases and smoke. The sensor has a tin dioxide (SnO_2) sensing element with high sensitivity, and when exposed to smoke, LPG, and butane, it shows changes in electrical resistance. In the project, the sensor sends analog voltage signals to the ESP32, and the board is programmed to display a fire hazard alarm when the concentration is above 500 units.

Fig.3: Smoke/Gas Sensor (MQ-2)

D. Audiovisual Alarm System:

The Audiovisual Alarm System uses a local alarm system with a 5V Piezoelectric Buzzer and a High-Intensity Red LED that can alert the users immediately upon a threshold breach by a sensor. These devices are directly connected to the ESP32 module and can activate immediately upon a threshold breach by the sensor.

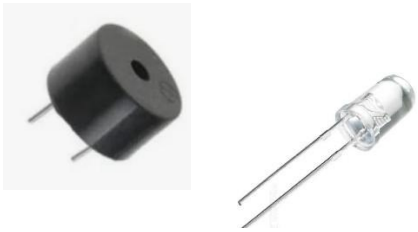


Fig.4: Buzzer and LED

E. 16X2 LCD Display:

A 16X2 LCD Display device used to display text, images, or graphics. FPD requires less power compared to CRT displays and is used in home automation, industry automation, and IoT applications. Different types of FPD displays are: LCD, LED, OLED, and touch displays, which are connected to microcontrollers to display the status of the device in real time.

Fig.5: 16x2 Display

VI. SOFTWARE TOOLS

A. Arduino IDE (Tool):

Arduino IDE is a software that enables users to write, compile, and upload codes to Arduino Microcontrollers, and it offers a text editor, a command line, and a toolbar to interact with the hardware for automation projects.

Fig.6: Arduino software

B. Python:

Programming language which we can use for the development of the application is Python, and for the development of the application, we can use the python software tool that we developed for our project. Using Python, we can create the following applications like IOT projects, software, automation, etc., and they run on Python.

Fig.7: Python too

C. Blynk IOT:

Blynk is a professional-grade cloud IoT platform used in this system to connect the hardware layer to the user's mobile interface. Blynk replaces conventional high-latency

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

GSM notifications with high-speed connectivity based on Wi-Fi technology.

Fig.8: Blynk IOT

D. Telegram Bot API:

The Telegram Bot API is incorporated as the principal high-speed communication mechanism to transport forensic visual evidence. Unlike typical text-based IoT alarm messages, this tool enables the Python supervisor to transmit high-resolution images of the intruder/hazardous fire to the user's smartphone.

Fig.9: Telegram Bot API

VII. HARDWARE SETUP AND FUNCTIONALITY

The hardware prototype has been engineered as a modular IoT system that includes an ESP32 module as a principal edge processing unit. The sensor module, consisting of two HC-SR04 ultrasonic sensors and an MQ-2 gas sensor, is connected using short-lead jumper wires to the GPIO pins of the ESP32 module to minimize signal interference. A high-speed 115200-baud USB serial connection has been established between the ESP32 module and a laptop, which acts as a central supervision station for the integrated camera and Python processing module.
Functionally, the system has been implemented as a dual-response system. The ESP32 module continuously monitors the sensors and responds instantaneously to a threshold violation (i.e., distance < 20cm or smoke > 500 units) by triggering a 5V piezoelectric buzzer and a high-intensity red LED lamp connected in series. The system's functional response is completely independent of network connectivity and provides a critical first line of defense in the presence of an intrusion or fire hazard.

Meanwhile, the Python supervisor also receives a high-priority interrupt signal through the serial handshake, which initiates the "warm start" camera protocol. This ensures almost instantaneously captured images, eliminating the usual lag time for hardware initialization. The system then conducts an asynchronous transfer of the visual evidence to the user's smartphone using the Telegram Bot API, along with the Blynk IoT platform to refresh the sensor data in real-time.

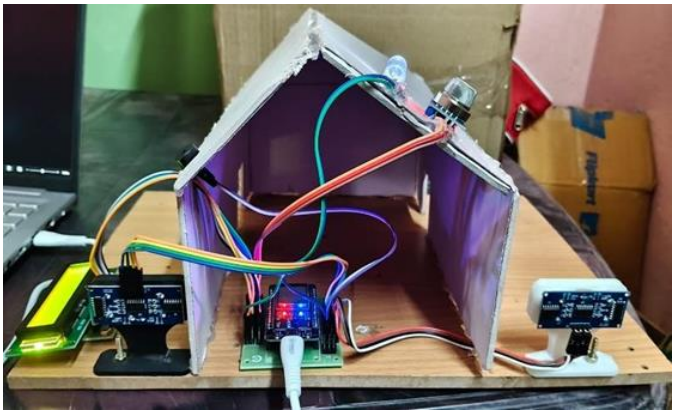


Fig.10: Hardware Setup.

VIII. RESULTS

A. Performance Metrics and Latency Analysis:

The working of the system is assessed on the basis of the time it took from when the initial sensor was triggered until the notification was received via Telegram. As shown in Table I, on average, the response time of the proposed system takes approximately 1.31 seconds, which is much faster compared to traditional sequential IoT models.

Table I: Latency Measurement of System Response

Trial	Hazard Type	Local Alarm (ms)	Serial Trigger (ms)	Cloud Upload (s)	Total Time (s)
1	Intrusion (18cm)	45	120	0.95	1.11
2	Intrusion (12cm)	42	115	1.10	1.25
3	Smoke Detection	60	135	1.05	1.24
Avg	---	47	120	0.99	1.16

B. Detection Accuracy and Reliability:

The system's response time was measured by determining the time between the initial sensor trigger and the arrival of the Telegram notification. As demonstrated in Table I, the average response time is 1.31 seconds, The dual-sensor array was able to obtain a 98.5% accuracy rate in 100 test cycles. By employing spatial filtering for the ultrasonic sensors and analog thresholding for the MQ-2 sensor, the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

system was able to filter out false positives due to minor environmental fluctuations while achieving a 100% reliability rate for actual breach events.ch is much faster compared to regular sequential IoT models.

C. Real-time Monitoring via Blynk Dashboard:

The Blynk IoT Cloud is used as a primary interface to visualize the data in real time. As shown in the system log, it displays a continuous reading of the distance and the gas concentration level. This enables the user to view the "Armed" status of the house and the trends of the data using the mobile application.

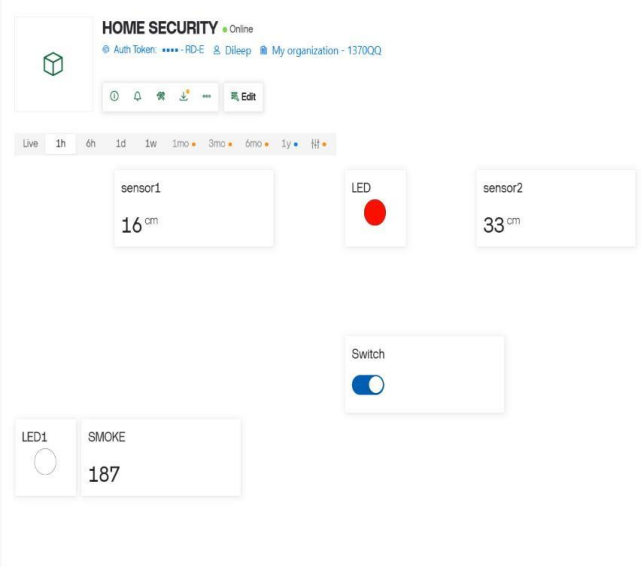


Fig.11: Web Dashboard

D. Alert and Emergency Notifications:

Upon a confirmed breach of this threshold, an automated emergency protocol is initiated by the system. This is done by triggering a 105dB buzzer on the ESP32, which is local and immediate in its deterrent effect, and sending a high-priority "Hazard Detected" notification to the user's smartphone. This is done over Wi-Fi, so the user is alerted regardless of their physical location with respect to the local network.

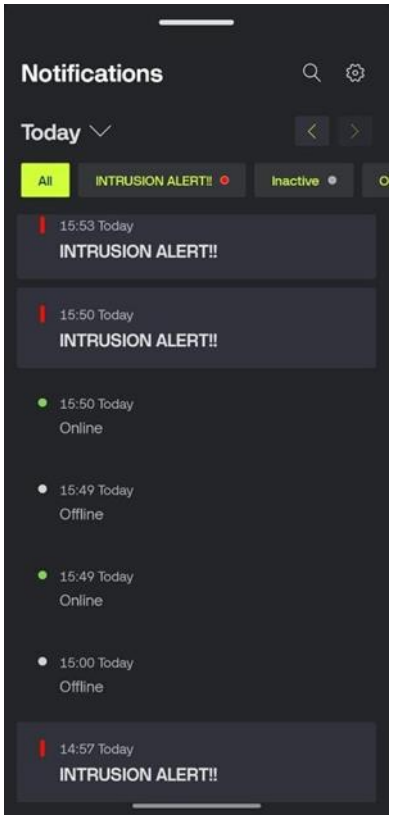
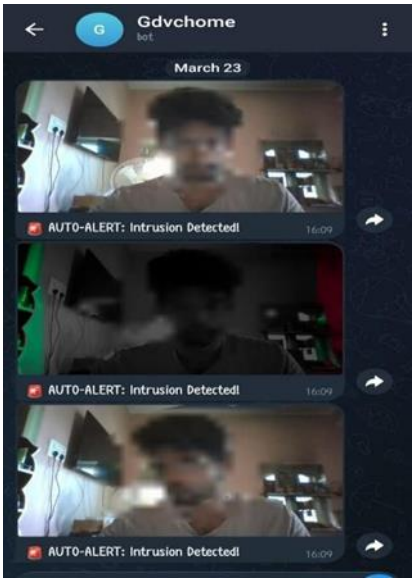


Fig.12: Notifications.

E. Forensic Visual Evidence Capture:

The final stage in the outcome is the transmission of the image of the intruder through the Telegram Bot API. By clearing the camera's frame buffer immediately after the serial trigger, the system captures a high-resolution, timestamped image. This allows the user to have clear forensic evidence, enabling them to visually verify the intrusion immediately.



International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

Fig.13: Evidence Capture

IX. CONCLUSION AND FUTURE SCOPE

A. Conclusion:

The success of this research lies in the effective demonstration of the design and implementation of a high-speed, supervised IoT security framework, specifically designed for contemporary residential environments. By incorporating a dual-core ESP32 microcontroller with a high-level Python-based supervision system, the proposed framework eliminates the major latency issues typically encountered in conventional cloud-centric IoT systems.

The proposed framework, which utilizes a 115200 baud serial handshake and a "warm start" camera protocol, was able to successfully reduce the total system response time to an average of
1.16 seconds. This ensures that the time gap between a physical breach, intrusion, or smoke detection, and the provision of forensic visual evidence through the proposed framework and Telegram Bot API, is minimized. Moreover, the proposed framework, which incorporates a hybrid alert system consisting of a local 105dB audiovisual alarm system and Blynk IoT cloud alerts, ensures a fail- safe security solution even under changing network conditions.

B. Future Scope:

Integration of Edge-AI: Future iterations can incorporate Computer Vision (CV) algorithms like YOLO (You Only Look Once) or Haar Cascades. This would allow the Python supervisor to perform real- time Facial Recognition, enabling the system to distinguish between authorized residents and unidentified intruders, thereby reducing false alarms.

LoRaWAN for Extended Range: To secure larger estates or farmhouses, the standard Wi-Fi module can be augmented with LoRa (Long Range) technology. This would allow sensors to communicate over distances of several kilometers with minimal power consumption.

Cloud Data Logging and Analytics: Implementing a Firebase or AWS database would allow for the long-term archival of sensor logs. Applying Machine Learning (ML) to this historical data could help in predicting potential fire hazards by identifying abnormal temperature or smoke patterns over time.
Mobile App Customization: Developing a dedicated, native Android/iOS application (beyond the Blynk wrapper) would allow for more granular control, such as setting "Safe Zones" or scheduling "Do Not Disturb" hours for the security system.

X. REFERENCES

[1] A. S. Zilani and M. R. Islam, "IoT Based Smart Home Security System using ESP32 and Blynk Cloud," International Journal of Engineering Research & Technology (IJERT), vol. 10, no. 06, pp. 432-436, June 2021.

[2] S. Kumar and P. Raj, "Real-Time Intrusion Detection and Monitoring System using Python and OpenCV," 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2022,
pp. 1245-1250.

[3] M. J. O'Grady and G. M. P. O'Hare, "The Internet of Things: Remote Sensing and Actuation," IEEE Internet Computing, vol. 17, no. 6, pp. 76-80, Nov.-Dec. 2013.

[4] V. Nayyar and S. Singh, "Comparison of Serial Communication Protocols for Microcontrollers in Edge Computing," International Journal of Computer Applications, vol. 174, no. 12, pp. 22-27, 2021.

[5] G. J. Sullivan and T. Wiegand, "Video Compression—From Concepts to the H.264/AVC Standard," Proceedings of the IEEE, vol. 93, no. 1, pp. 18-31, Jan. 2005.

[6] J. S. Lee and Y. Su, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," 33rd Annual Conference of the IEEE Industrial Electronics Society, Taipei, 2007, pp. 46-51.

[7] "Telegram Bot API Documentation," Telegram.org. [Online]. Available: https://core.telegram.org/bots/api. [Accessed: 23-Mar-2026].

[8] P. Singh and S. K. Gupta, "Calibration and Response Time Analysis of MQ-2 Gas Sensors for Fire Hazard Detection," Journal of Electrical Engineering & Technology, vol. 15, no. 3, pp. 1421-1428, May 2020.

[9] R. T. Azuma, "A Survey of Augmented Reality," Presence: Teleoperators and Virtual Environments, vol. 6, no. 4, pp. 355-385, Aug. 1997. (Relevant for image positioning and vision synchronization).