# HOMESHIELD A Home Monitoring and Danger Alert System

Karthik H[1], Nived Krishna A[2], Anitha A S[3], Mohammed Rishal K[4], Sidhinchandran C S[5]
[1,2,3,4,5]*Department. Of Computer Science and Engineering Vidya Academy of Science and Technology Thalakkottukara, Thrissur, India*

***Abstract*—There is a need for a centralized and automated home security system that analyzes data from certain sensors and identifies potential threats. This system will enhance user experience by notifying homeowners of hazards as they arise and attempt to prevent them by offering personalized, data-driven, and sensor-based interventions. If a user is at risk, they will be the first to know. This lab aims to create the world's first innovative Smart Home Security System (SHSS) powered by the Internet of Things (IoT) to detect different categories of hazards, SMSS along with real-time personalized alerts and recommended courses of action, which will be determined based on sensor data from an MQ2 gas and smoke sensor, PIR, and a facial recognition system paired with an anti-burglary system, a flame sensor, a water level sensor to detect flood, a vibration sensor to detect earthquake, data analytical processing powered by Raspberry Pi technology, and real time alerts will be received through SMS services and direct hosting on Raspberry Pi. This data which is received from sensors in the area will be collected in a centralized system, and will be successfully conveyed to homeowners. The hardware is powered through solar energy, thus offering an environmentally sustainable system. This project aims to demonstrate the efficacy of these systems during real life circumstances.**

***Index Terms*—IoT, OpenCV, YOLO, RaspberryPi, Home-Shield, ESP32-CAM**

## I. INTRODUCTION

Home safety and security have become increasingly vital in modern living environments, where potential hazards such as gas leaks, fire outbreaks, flooding, earthquakes, and unauthorized intrusions pose significant risks.

Our project, Home Shield, goes beyond old-school security systems. Instead of working as separate devices, it uses an integrated system of sensors, microcontrollers, and software together into one connected setup. The system detects various domestic threats simultaneously and facilitates responses through real-time monitoring and instant alerts.

## II. LITERATURE SURVEY

### A. A Face Recognition System Based on Cloud Computing and AI Edge

Vision-based systems can be categorized into two types according to their image processing methods: cloud-based processing and local processing. In a cloud-based system, the vision sensor captures image data and uploads it to a cloud service center for processing. In a local processing system, the image data acquired by the vision sensor is processed directly within the local hardware system. In a cloud-based face recognition system, the vision sensor uploads the captured image data and waits for the cloud center to perform a comparative analysis. The advantages of this approach include a simple structure, low cost on the device side, and scalability—the system can support varying numbers of vision sensors depending on the capacity of the cloud computing center. However, because image data must be transmitted over a network, the system remains highly susceptible to network conditions, even when compression techniques are used to reduce data size. When the network state is poor, delays can be significant. Moreover, the computational expense of concurrent processing becomes substantial when a large number of vision sensors are connected, placing heavy demands on the

cloud data center.

By contrast, in a local face recognition system, image data is acquired by the vision sensor and is sent directly to a local computing device for comparison and analysis. The main advantage is that processing is fast and unaffected by network conditions. The drawback, however, is that local hardware capable of running AI-based recognition tends to be expensive and can be inconvenient to install. Given the strengths and limitations of both systems, this paper proposes a face recognition system based on AI Edge and cloud computing (AE-FRS). This hybrid approach offers lower network delay than traditional cloud-based systems, while being more cost-effective and easier to install than fully local face recognition systems.

B. IoT based monitoring and control system for home automation

The project proposes an efficient implementation for IoT (Internet of Things) used for monitoring and controlling the home appliances via World Wide Web. Home automation system uses the portable devices as a user interface. They can communicate with home automation network through an Internet gateway, by means of low power communication protocols like Zigbee, Wi-Fi etc. This project aims at controlling home appliances via Smartphone using Wi-Fi as communication protocol and raspberry pi as server system. The user here will move directly with the system through a web-based interface over the web, whereas home appliances like lights, fan and door lock are remotely controlled through easy website. An extra feature that enhances the facet of protection from fireplace accidents is its capability of detecting smoke in order that within the event of any fireplace, associates an alerting message and an image is sent to Smartphone. The server will be interfaced with relay hardware circuits that control the appliances running at home. The communication with server allows the user to select the appropriate device. The communication with server permits the user to pick out the acceptable device. The server communicates with the corresponding relays. If the internet is down or the server isn't up, the embedded system board still will manage and operate the appliances domestically. By this we provide a climbable and price effective Home Automation system.

C. IoT-Enabled Home Automation System

Home automation is a crucial component of the Internet of things, sometimes known as Internet of things (IoT). For decades, home automation has been utilized to control basic household goods such as lighting and small appliances. The globe will be connected with a touch of a fingerprint or simple voice instructions, according to current technology. The existing system was not secure and safe because it did not have fire detection and notification system. And it was high cost and high-power consumption. The well- known IoT home automation system is typically straightforward to implement in a real home, allowing for real-time monitoring of home conditions and control of home equipment. Several sensors and actuators were connected to the NodeMCU controller, and the updated temperature, humidity, motion, and gas data could be seen on laptops and PCs using the MQTT Dash mobile application and the Adafruit IO website. For security and safety concerns, users receive warnings on their mobile phones when there is an odd scenario via the IFTTT server. Household appliances are regularly controlled in a simple and effective manner using the MQTT/Adafruit IO GUI or voice commands with Google Assistant. The proposed system aims to expand the Home Automation system with additional sensors, actuators and instead of utilizing batteries, solar panels are used to power the control box, making the system safe, secure, energy-efficient, and environmentally beneficial.

## III. SYSTEM DESIGN

For Home Shield, the design focuses on creating an integrated home monitoring and danger alert system that efficiently collects, processes, and communicates data from multiple sensors to provide real-time safety alerts.

A. Perception Layer (Sensors)

The perception layer forms the foundation of the Home-Shield system, comprising multiple sensors that monitor different environmental parameters:

- Flame Sensor: Detects fire outbreaks through infrared radiation sensing
- Gas Sensor (MQ Series): Identifies hazardous gas leaks including LPG, smoke, and carbon monoxide
- Vibration Sensor: Monitors structural vibrations indicative of earthquakes or physical intrusions

- Motion Sensor (PIR): Detects human movement for intruder detection
- Flood Sensor: Monitors water levels to prevent flooding damage

B. Arduino Uno Microcontroller

The Arduino Uno serves as the primary data acquisition unit, interfacing directly with all sensors. It performs initial data processing, formatting, and coordinates the transmission of sensor readings to the LoRa communication module. The Arduino's versatility allows for easy integration of additional sensors and provides reliable performance in continuous monitoring scenarios. This system is powered completely with the help of solar energy. Solar panels, whose output is 22W power, are connected to a 12V battery which is rechargeable and stores electrical energy for system usage. Since this voltage is too high for the individual electronics, it must be stepped down. This is done with a charge controller. Together, this is the power source for the Arduino in this system, thus making it sustainable and environmentally friendly.

C. Communication Layer (LoRa Modules)

The system employs LoRa (Long Range) technology for wireless data transmission, consisting of:

- LoRa TX Module: Connected to Arduino Uno for trans-mitting sensor data
- LoRa RX Module: Interfaces with Raspberry Pi 5 for data reception

LoRa technology provides long-range communication (up to several kilometers) with low power consumption, ensuring reliable operation even in areas with limited WiFi connectivity.

D. Raspberry Pi 5 Processing Unit

The Raspberry Pi 5 acts as the central processing hub, responsible for parsing and validating incoming sensor data from LoRa RX, implementing threshold-based hazard detection algorithms, coordinating system operations and data flow, running the YOLO model for facial recognition and managing database operations and backend communication

E. Backend System (Node.js & Express.js)

The backend infrastructure provides:

- Real-time alert mechanisms and notification management
- Database integration and data storage management
- WebSocket connections for live updates

F. MongoDB Database

Stores historical sensor data and event log, user information and system configurations, alert history and notification records and facial recognition data and intruder detection logs

G. Frontend Dashboard (React.js)

The React.js-based web application provides real-time sensor data visualization, live alert notifications and status up-dates, historical data analysis and reporting features, responsive design for both desktop and mobile access and user-friendly interface for system configuration

H. Advanced Intruder Detection System

A specialized security subsystem incorporating:

- mmWave Radar Sensor: Detects human presence with-out privacy concerns
- ESP32-CAM: Captures images when human presence is detected
- YOLO Model: Performs real-time human detection
- Multi-channel alert mechanisms for immediate notification

I. Mobile Application (Kotlin with WebView)

The Android mobile application, developed with Kotlin, shows all sensor data, and also uses WebView technology to provide seamless access to the web dashboard. In addition to basic monitoring, the mobile application performs real-time push-like updates using WebSocket connection. It also uses FCM technology to schedule notifications as per required, alerting users of sensors even if the application is not running in the background. Therfore, instant alert notifications are pushed for efficient hazard and intrusion warnings. The data is accessed securely and remotely, thus allowing users to monitor their homes from anywhere. For an even smoother user experience, an LLM chatbot has been implemented on the app using the Sarvam LLM API. This allows easy, cheap, and efficient tokenization of the input from 22 Indian languages, thus making it cheaper.

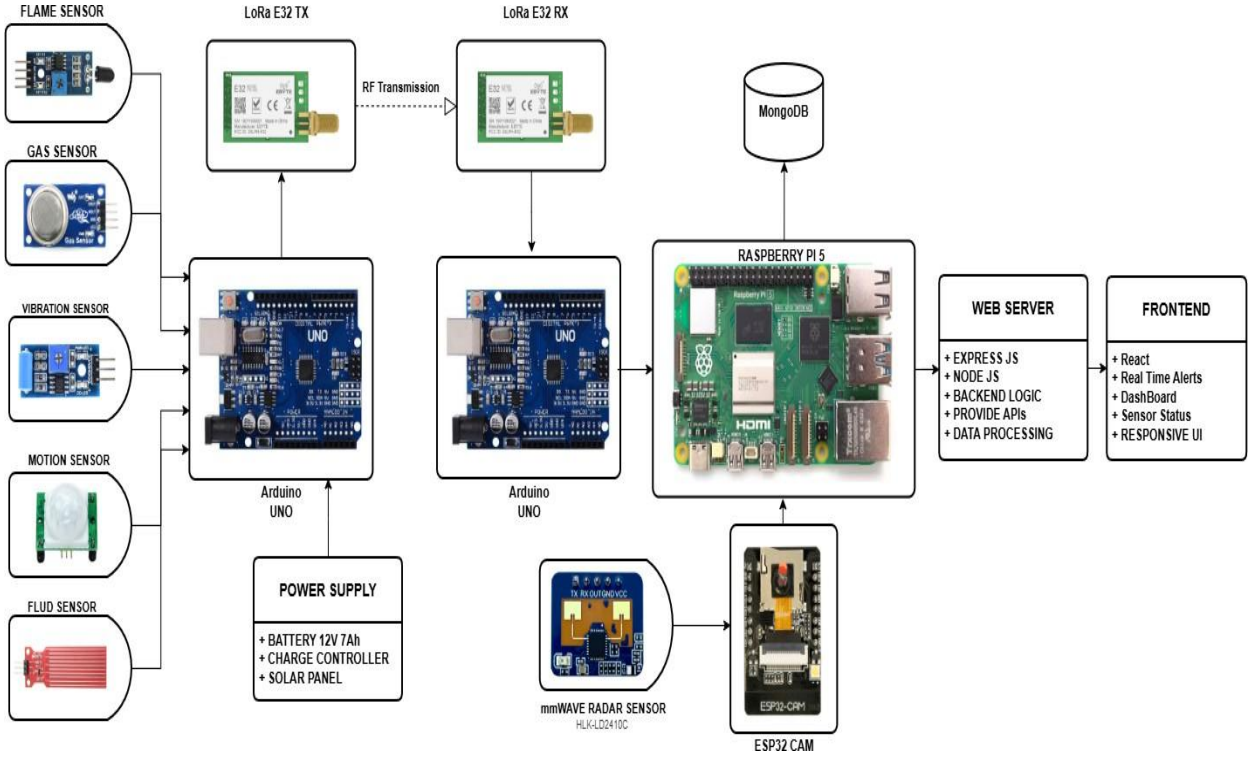


Fig. 1. System Architecture

## IV. METHODOLOGY

The Home Shield system starts with a complex hardware configuration that uses several sensors to keep an eye on different household dangers. While MQ-series gas sensors detect dangerous gas leaks, such as LPG, smoke, and carbon monoxide, flame sensors use infrared radiation to detect fire outbreaks. PIR motion sensors identify human movement for intruder warnings, while vibration sensors track building motions suggestive of earthquakes. To stop flooding damage, flood sensors monitor water levels. An Arduino Uno micro-controller, which interfaces with every sensor, handles the preliminary data collection, processing, and formatting prior to transmission.

For dependable wireless communication between components, the system makes use of LoRa (Long Range) technology. A LoRa RX module attached to a Raspberry Pi 5 receives formatted sensor data transmitted by the Arduino Uno via a LoRa TX module. This communication layer maintains system functionality even in places with spotty WiFi access by ensuring long-range data transmission (up to several kilometers) with minimal power usage. As the central processing unit, the Raspberry Pi 5 parses incoming data and gets it ready for storage and additional analysis.

To identify possible risks, the Raspberry Pi 5 uses threshold-based algorithms to assess incoming sensor data. The technology uses temperature data along with flame sensor readings to identify fires. In order to identify gas leaks, MQ sensor data must be compared against safety criteria. While flood sensors sound an alarm when water levels above predetermined thresh-olds, vibration data is examined to find patterns suggestive of earthquakes. Every processed piece of data is organized and ready to be stored in the MongoDB database, and real-time analysis makes it possible to identify hazards right away.

The Home Shield system's main processing capabilities are provided by the backend infrastructure, which was constructed using Node.js and Express.js. It uses RESTful APIs for system control, alarm management, and data handling. The backend handles user identification and system setups in addition to processing sensor data from the Raspberry Pi and storing it in MongoDB. In order to minimize false positives and guarantee reliable

warning creation, it also manages the business logic for hazard detection by cross-referencing several sensor inputs. React.js was used to create a responsive web dashboard that offers customers real-time monitoring capabilities. Through interactive charts and visualizations, the frontend presents historical trends, current system status, and real-time sensor data. It has a real-time alarm system that lets consumers know as soon as dangers are identified. In order to provide the best possible viewing experience on desktop, tablet, and mobile devices, the interface was created with responsiveness in mind. System settings and alert choices can also be configured on the dashboard.

The system has an advanced intruder detection subsystem that uses ESP32-CAM modules and mmWave radar sensors. The ESP32-CAM takes pictures when the mmWave radar senses human presence. The Raspberry Pi's YOLO (You Only Look Once) models are used to process these photos for real-time facial recognition and person detection. Based on the recognition findings, the system can differentiate between authorized individuals and possible intruders, sending out the proper alerts.

To guarantee prompt notifications, Home Shield uses a multi-channel alert system. The system immediately sends out alerts via the web dashboard with visual and aural cues when it detects dangers. Homeowners and authorities can also receive SMS messages, emails, and mobile push notifications via the system. Emergency situations trigger immediate reaction methods, and alert severity levels are calibrated based on the type and intensity of detected risks.

This system is powered completely with the help of solar energy. Solar panels, whose output is 22W power, are connected to a 12V battery which is rechargeable and stores electrical energy for system usage. Since this voltage is too high for the individual electronics, it must be stepped down. This is done with a charge controller. Together, this is the power source for the microcontrollers in this system, thus making it sustainable and environmentally friendly.

A. System Integration and Testing

Every component is combined into a single system, and extensive testing is done to guarantee performance and dependability. End-to-end system validation, integration testing of sensor networks, and unit testing of individual components are all included in the testing phase. Performance testing assesses the system's stability, alert accuracy, and reaction times under different circumstances. User acceptability testing guarantees that the interface satisfies usability standards and offers a user-friendly interface.

B. Deployment and Maintenance

The last stage is setting up maintenance procedures and installing the entire system in a household setting. User training, software configuration, and hardware installation are all part of the deployment. Software upgrades, sensor calibration, and system health monitoring are examples of routine maintenance processes. In order to improve system functioning and dependability over time, user feedback is integrated into ongoing performance evaluation, which aids in identifying areas for development.

## V. DATA FLOW

HomeShield's data flow architecture is an example of an advanced IoT ecosystem intended for real-time threat detection and thorough home monitoring. In order to ensure reliable performance and quick response times in a variety of emergency situations, the system uses a multi-layered approach to data collection, processing, analysis, and alarm generating.

The first stage entails ongoing environmental monitoring using a variety of specialized sensors placed thoughtfully throughout the house. By recognizing particular radiation patterns linked to combustion, flame sensors using infrared spectroscopy techniques identify fire outbreaks. MQ2 gas sensor use chemical detecting techniques to identify dangerous gases like LPG, natural gas, carbon monoxide, and smoke particles. Vibration sensors equipped with SW-420 modules keep an eye on the stability of the structure and identify any efforts at forced entry or seismic activity. Flood sensors use conductivity principles to monitor water levels in susceptible locations, and passive infrared (PIR) motion sensors assess human movement patterns. Specific sample rates and sensitivity thresholds tailored to each sensor's detecting objectives are used in its operation.

The Arduino Uno microcontroller serves as the

primary data aggregation point, implementing several critical functions. It performs analog-to-digital conversion on raw sensor signals, applies initial noise filtering algorithms, and normalizes data into standardized formats. The microcontroller also handles sensor calibration routines and implements basic threshold checking to reduce false positives. Through its serial communication interface, the Arduino packages sensor data into structured JSON format and manages the timing and protocol requirements for LoRa transmission. The system incorporates error-checking mechanisms to ensure data integrity during this preliminary processing stage.

The LoRa (Long Range) communication system establishes a robust wireless link between sensor nodes and the central processing unit. Using spread-spectrum modulation techniques, the LoRa TX modules transmit sensor data packets over distances up to several kilometers while maintaining low power consumption. The communication protocol includes packet acknowledgment, retransmission mechanisms for failed deliveries, and frequency hopping to mitigate interference. The LoRa RX modules on the Raspberry Pi 5 side receive these transmissions, verify data integrity through checksum validation, and forward the decoded information to the main processing pipeline.

The Home Shield system's intelligent core is the Raspberry Pi 5, which uses complex algorithms for data processing and decision-making. It uses several analytical layers to process incoming sensor streams, such as anomaly detection for unexpected emergency circumstances and temporal pattern identification for gradual danger development. In order to improve detection accuracy and lower false alarms, the system uses machine learning models to correlate data from several sensors. For instance, fire identification is more accurate when rising temperature readings from flame sensors and smoke detection from gas sensors are combined than when either sensor is used alone.

Computer vision and radar technology are used in a simultaneous processing stream to detect intruders. Using frequency-modulated continuous wave (FMCW) radar principles, the mmWave radar sensors continually track spatial movements and can detect human presence through walls and in total darkness. The system activates ESP32-CAM modules to take high-resolution pictures when it detects suspicious movement. YOLO (You Only Look Once) convolutional neural networks, which are optimized for real-time human detection and facial recognition, process these photos. The system can differentiate between residents, anticipated guests, and possible invaders and keeps a database of authorized persons.

Home Shield initiates a complex multi-tier alert distribution mechanism upon hazard identification. Using WebSocket connections for real-time updates, primary notifications with color-coded severity indicators and auditory warnings show instantly on the React.js online dashboard. Secondary notifications are distributed via a variety of channels, such as smartphone push notifications for prompt attention, email systems for comprehensive reporting, and SMS gateways for urgent alerts. Specific hazard information, location information, suggested actions, and real-time updates as the situation changes are all included in the alert material. The system employs escalation protocols; whereby unanswered critical alerts result in further warnings to local authorities and emergency contacts.

## VI. HARDWARE REQUIREMENTS

### A. Arduino Uno

The main data collecting component of the Home Shield system is the Arduino Uno, a microcontroller board built on the ATmega328P. Its digital and analog input pins allow it to communicate with a variety of sensors, including flame, gas, vibration, motion, flood, and LDR sensors. In order to identify abnormal situations, the Arduino continually receives sensor data at regular intervals and carries out preliminary preprocessing tasks like noise filtering, signal stabilization, and threshold comparison. The processed data is then for-matted into structured packets that include timestamps, sensor readings, and alert status. Additionally, the Arduino controls wireless transmission connectivity with the LoRa module. It is perfect for real-time monitoring systems because of its ease of use, dependability, and interoperability with a variety of sensors.

### B. Sensors

The system incorporates a set of sensors to monitor various environmental parameters and detect hazardous conditions.

Flame Sensor: uses infrared radiation released during combustion to detect fire. It is appropriate for early fire detection because of its great sensitivity and quick reaction time.
MQ2 Gas Sensor: uses resistance changes to identify flammable gases like carbon monoxide, methane, LPG, and smoke. Because of its sensitivity and dependability, it is frequently used for gas leak detection.
Vibration Sensor (SW-420): detects shocks and vibrations brought on by impacts, entry attempts, or structural disruptions. When vibration beyond a predetermined threshold, it generates a digital signal.
PIR Motion Sensor: uses variations in infrared radiation to identify human movement. It is employed to detect motion in the area under observation and activate the intrusion detection system.
Flood Sensor: detects the presence of water and monitors water levels, helping to identify potential flooding conditions in real time.
LDR Sensor: measures ambient light intensity and helps detect unusual environmental changes such as sudden darkness or light variations.

C. LoRa E32 Module (TX/RX)
The sensor unit and the central processing unit may communicate wirelessly over long distances thanks to the LoRa E32 module. Sensor data is sent via the transmitter (TX) module attached to the Arduino and received by the receiver (RX) module attached to the Raspberry Pi. Low-power wide-area network (LPWAN) principles underpin LoRa technology, which allows for low-power communication over many kilo-meters. It is appropriate for continuous monitoring applications because it guarantees dependable data delivery even in places with inadequate network infrastructure.

D. Raspberry Pi 5
The Home Shield system's central computing unit is the Raspberry Pi 5. After receiving sensor data from the LoRa receiver, it verifies the incoming packets and uses threshold-based algorithms to interpret the data in order to identify potentially dangerous situations. The Raspberry Pi does sophisticated tasks including image processing for intruder detection using the YOLO model in addition to processing sensor data. In addition, it regulates data flow between hardware and software components, hosts the backend server, and controls database communication. It is perfect for managing IoT and AI workloads because to its high processing capacity and support for several interfaces.

E. ESP32-CAM
The ESP32-CAM is a small microcontroller module that has Wi-Fi and a built-in camera. When motion is detected, the intrusion detection module uses it to take pictures. The module takes real-time pictures of the monitored area when it is activated by PIR or mmWave sensors. Using HTTP requests, these pictures are encoded and sent over Wi-Fi to the Raspberry Pi. The ESP32-CAM readily interfaces with IoT systems and offers an affordable solution for image-based monitoring.

F. mmWave Radar Sensor
For sophisticated human presence detection, the mmWave radar sensor is employed. In contrast to PIR sensors, it detects motion using radio waves and can detect human presence through obstructions or in low light. By lowering false positives, this improves the intrusion detection system's accuracy and dependability.

G. Solar Panel and Battery
This system is powered with solar energy. The solar panels that are used are 22W solar panels, and these generate electricity, stored in a 12V 7AH battery. This voltage is too high for the electronics, and must be stepped down properly using a charge controller. This powers the system as a whole.

## VII. REQUIREMENTS

A. Node.js and Express.js
Express.js serves as the web framework for creating RESTful APIs, while Node.js serves as the backend runtime environment. Data processing, system control, alarm generating, and communication between system components are all handled by the backend. After receiving data from the Raspberry Pi and processing it according to predetermined rules, it transmits answers to the mobile application and frontend.

B. MongoDB
System data can be stored in a flexible and structured

fashion using MongoDB, a NoSQL database. Sensor readings, alert logs, timestamps, user configurations, and intruder detection outcomes are all stored there. Real-time and historical data may be efficiently stored, retrieved, and analyzed because to its document-based format.

C. React.js

The frontend dashboard is created using React.js, which offers a responsive and interactive user experience. It uses visual components including graphs and color-coded indicators to provide real-time sensor data, system status, and alarm signals. WebSocket connections and APIs are used by the frontend and backend to exchange real-time updates.

D. Kotlin

The Android mobile application for remote monitoring was created using Kotlin. Users may access system functions on mobile devices thanks to the application's integration with WebView, which loads the React-based dashboard. It ensures continuous monitoring from any place by supporting user interaction, remote accessibility, and real-time alarm messages.

E. WebSocket

Socket is used to implement WebSocket technology. Real-time bidirectional communication between the frontend and backend is made possible using IO. It ensures low latency and effective data delivery by enabling the immediate transmission of sensor updates and alarms without the need for repeated queries.

F. YOLO

The intruder detection module uses YOLO, a deep learning-based object detection system, to detect people in real time. It enables intelligent intrusion detection by processing photos taken by the ESP32-CAM and accurately identifying human presence.

G. Python

The system's AI and image processing components are implemented using Python. It facilitates the effective creation and implementation of real-time detection algorithms by sup-porting libraries like OpenCV and YOLO frameworks.

## VIII. RESULT

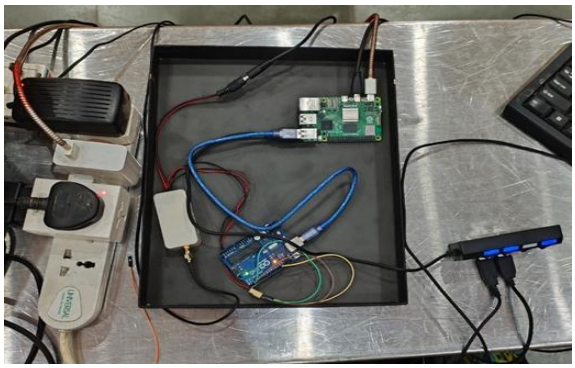

Fig. 2. Raspberry Pi LoRa connection

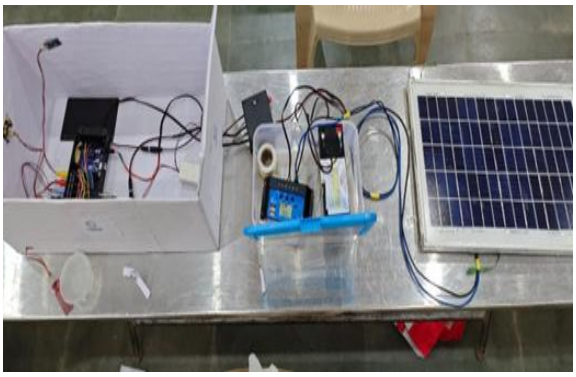

Fig. 3. Solar panel, battery and charge converter connected to system

This project's development produced a smart home security system with state-of-the-art Internet of Things applications, wireless data transmission over long distances, a responsive, contemporary dashboard that offers real-time hazard detection through interactive charts and tables, and a system powered sustainably by solar energy. The introduction of smooth picture upload capabilities, which enables users to submit photographs with ease using a drag-and-drop interface, was a significant accomplishment. The integrated YOLO model, which has shown excellent human detection accuracy, processes these photos in real-time. The solution creates an effective end-to-end pipeline from user input to AI-powered danger identification by producing unambiguous visual results, such as bounding boxes surrounding observed persons and conclusive classification.

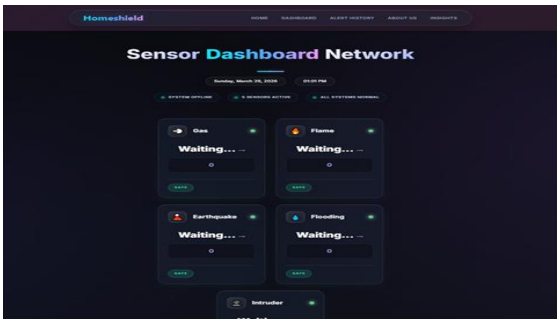


Fig. 4. Sensor Dashboard in Website

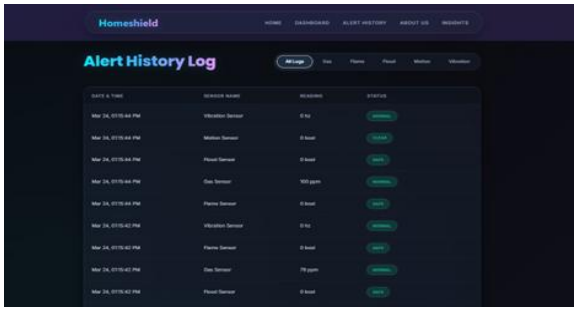


Fig. 5. Alert History

A strict testing regimen was put in place to make sure the system fulfilled its design requirements and was dependable for end-user deployment. Sensor accuracy, IoT connection, real-time data processing, and machine learning capability were all carefully assessed during this procedure. Inconsistent sensor readings, sporadic network connectivity issues, and false positives in hazard recognition were among the difficulties found during the early stages of testing. Through sensor calibration, enhanced signal processing methods, and incremental improvements to the detection algorithm, these problems were methodically resolved. Using the Sarvam LLM API, an LLM chatbot has been added to the app for an even more seamless user experience. This makes tokenization of the input from 22 Indian languages simple, affordable, and

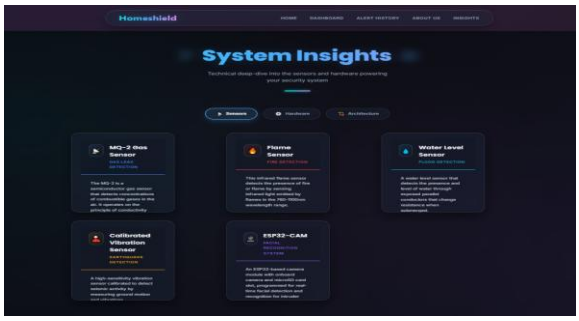


Fig. 6. System Insights

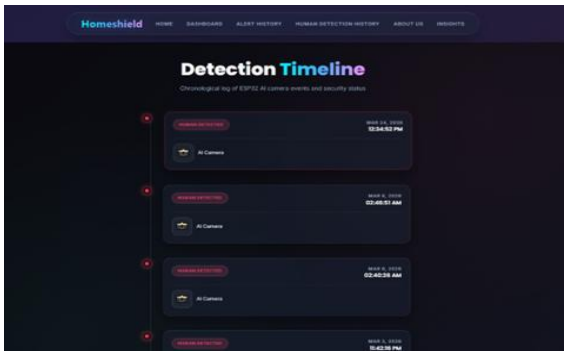


Fig. 7. Human detection history

effective, which lowers the cost. Additionally, performance and compatibility testing were carried out in a variety of environmental


Fig. 8. App sensor data representation

A reliable and strong Advanced IoT-Powered Hazard Detection System is the result of this development and iterative testing procedure. High detection accuracy, a user-friendly interface, and dependable real-time performance define the final system. It is now ready to deliver reliable early warning capabilities, giving communities and homeowners a reliable tool to improve home security and safety.

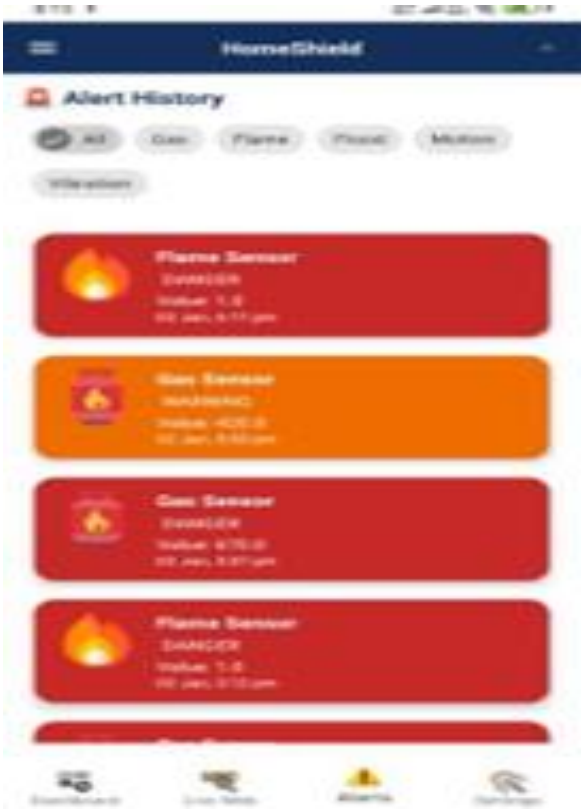


Fig. 9. Alert history in App

The Advanced IoT-Powered Hazard Detection System was evaluated based on its functionality, accuracy, performance, reliability, and security to determine its effectiveness in real-world applications. The system successfully demonstrated precise monitoring of environmental parameters, including gas leak detection, flame detection, water level monitoring, earthquake detection, human intruder detection, which are key indicators of potential hazards we see in our day to day lives. The integration of human detection using YOLO v8 S significantly improved detection accuracy, reducing false positives and enhancing detection capabilities. Extensive testing confirmed that sensor readings remained consistent and reliable across different environmental conditions, ensuring the system's adaptability to diverse terrains. The human detection framework, built on the ESP32-CAM, facilitated real-time data transmission and processing of images, enabling rapid decision-making and timely alert generation across various platforms.


Fig. 10. In-app LLM for convenient use

Performance assessments indicated low-latency data processing and optimized and sustainable power consumption, making the system suitable for deployment in various different types of environments with limited power sources. Network stability tests validated the system's resilience to intermittent connectivity issues, with built-in failover mechanisms, ensuring seamless data transmission. Additionally, compatibility testing confirmed smooth integration across multiple hardware components, web-based platforms and alerts through other platforms such as gmail and telegram, allowing for efficient remote monitoring. Based on these findings, the system is deemed highly reliable for home safety, offering a scalable and effective solution for hazard detection and risk mitigation.

## IX. CONCLUSION

One of the most significant advancements in the field of intelligent home safety monitoring and danger detection systems is the deployment of the SHSS. The project has successfully moved away from the theoretical concept of such solutions and created a useful, effective, and superior hardware and software combination. The system's multi-layer architecture, which includes essential components like data collection via sophisticated sensors and the ESP32

microcontroller, edge computing capabilities via the Raspberry Pi device, and the dynamic, user-oriented dashboard based on React.js, is what makes it successful.

## X. FUTURE SCOPE

While the current iteration of the SHSS is a fully functional and effective system, its architecture provides a fertile ground for several promising enhancements and expansions in future work:

Advanced AI Model and Multi-Class Detection: The YOLO model can be expanded beyond human detection to identify specific hazardous events or objects, such as fire, smoke, water leaks, or unauthorized intrusion attempts (e.g., detecting broken windows). Training a custom model on a broader dataset would significantly increase the system's utility and proactive warning capabilities.

Predictive Analytics and Alert Triage: Integrating ma-chine learning for time-series analysis of sensor data could enable predictive maintenance and early warnings. For instance, the system could learn normal patterns of temperature and humidity to predict potential electrical fault conditions before they become critical. Furthermore, implementing a smart alert system that prioritizes or filters notifications based on severity would reduce false alarms and ensure users attend to the most critical alerts first.

Enhanced Connectivity and Protocol Support: Future versions could incorporate a wider range of communication protocols, such as Zigbee, for even lower power consumption and greater range for sensor nodes. Additionally, integrating directly with smart home ecosystems (e.g., Amazon Alexa, Google Home, Apple HomeKit) would allow for automated responses, such as turning on lights upon detecting an intrusion or shutting off water mains in case of a leak.

Cloud Data Analytics: Using a cloud-based data analytics platform would enable long-term trend analysis, generating comprehensive reports on home safety and providing insights that could be valuable for home insurance or energy efficiency. Scalability for Community and Industrial Use: The system's scalable design can be adapted for larger deployments, such as multi-apartment buildings, office complexes, or small industrial sites. A centralized dashboard for property managers could monitor the status of multiple units simultaneously, representing a significant step towards smarter, safer buildings, and communities.

## REFERENCES


[1] Zeng,J., Li, C., Zhang, LJ. (2018). A Face Recognition System Based on Cloud Computing and AI Edge for IOT.In: Liu, S., Tekinerdogan, B., Aoyama, M., Zhang, LJ. (eds) Edge Computing– EDGE 2018. EDGE 2018.

[2] Pavithra and R. Balakrishnan, "IoT based monitoring and control system for home automation, " 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, India, 2015, pp. 169-173

[3] IoT-BASED SMART HOME FGM DETECTION SYSTEM. International Research Journal of Engineering and Technology (IRJET)Volume: 07 Issue: 05 — May 2020.

[4] G. R. Son et al., "Internet of Things in Geohazard Monitoring: Current Trends and Future Prospects," IEEE Internet of Things Journal, vol. 9, no. 5, 2022.

[5] Al-Fuqaha, A., Guibene, W., Mohammadi, M., Aledhari, M., Ayyash,

[6] M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. IEEE Communications Surveys Tutorials, 17(4), 2347-2376.

[7] Chen, J., Ran, X. (2019). Deep Learning with Edge Computing: A Review. Proceedings of the IEEE, 107(8), 1655-1674. 2021.

[8] Redmon, J., Farhadi, A. (2018). YOLOv3: An Incremental Improve-ment. arXiv preprint arXiv:1804.02767.

[9] Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., Zhao, W. (2017). A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. IEEE Internet of Things Journal, 4(5), 1125-1142.

[10] Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. (2013). Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. Future Generation Computer Systems, 29(7), 1645-1660.

[11] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L. (2016). Edge Computing: Vision and

Challenges. IEEE Internet of Things Journal, 3(5), 637-646.

[12] Stojkoska, B. L. R., Trivodaliev, K. V. (2017). A review of Internet of Things for smart home: Challenges and solutions. Journal of Cleaner Production, 140, 1454-1464.

[13] Bylykbashi, K., Elmazi, D., Ozera, K., Ikeda, M., Barolli, L. (2020). Effect of Security and Privacy Issues on Smart Home IoT for Fog Computing and Edge Computing. In 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS).

[14] Wang, X., Han, Y., Leung, V. C. M., Niyato, D., Yan, X., Chen, X. (2020). Convergence of Edge Computing and Deep Learning: A Com-prehensive Survey. IEEE Communications Surveys Tutorials, 22(2), 869-904.

[15] Moysiadis, V., Sarigiannidis, P., Moscholios, I. (2018). Towards Dis-tributed Data Management in Fog Computing. Wireless Communications and Mobile Computing, 2018.

[16] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770-778).

[17] Kopetz, H. (2011). Real-Time Systems: Design Principles for Distributed Embedded Applications. Springer Science Business Media.

[18] Andrea, I., Chrysostomou, C., Hadjichristofi, G. (2015). Internet of Things: Security vulnerabilities and challenges. In 2015 IEEE Sympo-sium on Computers and Communication (ISCC) (pp. 180-187).

[19] Bonomi, F., Milito, R., Zhu, J., Addepalli, S. (2012). Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing (pp. 13-16).

[20] Kiranyaz, S., Gastli, A., Ben-Brahim, L., Alemadi, N., Gabbouj, M. (2019). Real-time fault detection and identification for MMC using 1D convolutional neural networks. IEEE Transactions on Industrial Electronics, 66(11), 8760-8771.

[21] Ray, P. P. (2018). A survey on Internet of Things architectures. Journal of King Saud University-Computer and Information Sciences, 30(3), 291-319.

[22] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1-9).