

Hybrid online/offline Internet of Things-based smart home automation with voice control, motion-based security, and environmental monitoring for developing regions

Ebot Stanis Brian-Etta and Njitacke Tabekoueng Zeric*

Department of Electrical and Electronic Engineering, College of Technology, University of Buea, Buea, Southwest Region, Cameroon

Article History:

Received: March 25, 2026

Revised: April 30, 2026

Accepted: May 26, 2026

Published online: June 19, 2026

ABSTRACT

Smart home technology facilitates wireless control and monitoring of lighting, security, and heating/cooling systems. Although these systems have evolved significantly since the 1970s, adoption in developing regions is still hindered by high equipment costs and unreliable internet access. This study details the design and testing of a low-cost, hybrid (online/offline) Internet of Things-based smart home automation system, specifically built for developing areas. Using an ESP32 v4 microcontroller and the ESP RainMaker platform, the system can be managed through a mobile app, Google Assistant, or automated schedules when online. An infrared (IR) remote and physical manual switches work regardless of internet connectivity. Environmental monitoring includes a DHT22 temperature and humidity sensor for automated fan control, and an HC-SR501 passive IR motion sensor activates security lights and sends real-time alerts. Experimental bench testing ($n = 25$ per method) conducted over seven days showed average response times of 1.56 s for voice control, 1.43 s for app control, and 0.91 s for IR remote operation, while manual switches responded in under 0.4 s. Success rates were 94.33% under normal Wi-Fi conditions, 88.0% under poor connectivity, and 98.6% in offline mode. Energy savings were estimated to be around 21% based on scheduling performance. A six-month field trial in Cameroon confirmed that although cloud-based features may be affected during network congestion, the hybrid architecture maintains continuous system operation. Supporting up to 10 controllable loads, the system provides a low-cost and scalable solution for smart home deployment in developing regions.

Keywords: Internet of Things-based smart home automation; Google Assistant; ESP RainMaker; Multi-modal control; Temperature monitoring; Motion-based security; Infrared remote control; Hybrid system



*Corresponding author:

Njitacke Tabekoueng Zeric (njitacke.tabekoueng@ubuea.cm)

Citation: Brian-Etta ES, Zeric NT. Hybrid online/offline Internet of Things-based smart home automation with voice control, motion-based security, and environmental monitoring for developing regions. *Nonlinear Sci Cont Eng.* 2026;2(2):026130010. doi: 10.36922/NSCE026130010

Copyright: © 2026 The Author(s). This is an Open Access article distributed under the terms of the Creative Commons Attribution License, permitting distribution, and reproduction in any medium, provided the original work is properly cited.

1. Introduction

1.1. Background and problem statement

Smart home automation systems centralize control of lighting, heating, security, and appliances, reducing the need for manual operation.¹ The field has been evolving since the 1970s, moving from simple powerline communication to ecosystems that leverage artificial intelligence (AI), Internet of Things (IoT), WiFi, Bluetooth, the Global System for Mobile Communications (GSM), and cloud computing to improve convenience, security, and energy savings. The recent technology behind smart homes is the IoT, which enables wireless communication among sensors and devices, or “things,” such as light bulbs and thermostats, and the internet.^{2,3}

The integration of IoT into smart home systems allows homeowners to control their homes remotely.⁴ Contemporary systems like Google Home, Amazon Alexa, and Apple HomeKit platforms are common in Europe, America, and parts of Asia, and are increasingly available in some developing African countries.⁵ However, in other developing regions worldwide, adoption is slow due to high upfront costs, unreliable internet connectivity, and limited awareness. Moreover, most existing solutions require continuous internet connectivity, rendering them inoperable during network outages that are common in these regions.^{6,7}

Furthermore, a significant majority of current smart home solutions require stable bandwidth (>10 Mbps), most especially those with high-definition security cameras. This cloud dependency makes them unreliable during the frequent network outages in developing regions.^{8,9} Therefore, there is a need to develop solutions specifically engineered to operate within these constraints. To improve the comfort, security, and energy efficiency of homes in these areas, systems must prioritize a hybrid architecture that balances sophisticated cloud features with robust, offline local control.¹⁰

1.2. Objectives of this system

This work combines several control methods into a single system that operates with or without an internet connection. While not establishing a fundamental new architecture concept, it emphasizes the practical synchronization of existing hybrid online/off technologies to ensure reliability in developing regions with poor or unreliable internet. To achieve this, the specific objectives are:

- (i) To design a low-cost hybrid online/offline smart home system using the ESP32 microcontroller, ESP RainMaker platform, and Google Assistant integration that operates reliably with poor or no internet.
- (ii) To evaluate and optimize system performance in terms of response time, success rate, reliability, and energy consumption and efficiency through average measurements.
- (iii) To conduct a six-month field trial in Southwest Cameroon to evaluate real-world performance in a developing region.

- (iv) To improve user comfort, safety, and accessibility, including support for elderly and physically challenged users.

1.3. Related literature

Recent advancements by researchers and engineers show the transition toward semi- and fully-automated domestic environments, facilitated by the implementation of various wireless communication systems such as ZigBee, Bluetooth, infrared (IR), and Wi-Fi¹¹⁻¹³ as well as microcontroller families like the Arduino¹⁴, Raspberry Pi^{15,16}, and NodeMCU ESP series.¹⁷⁻¹⁹ The evolution of these systems has moved from simple point-to-point communication to complex cloud-integrated systems, but each advancement introduces specific limitations regarding cost, range, and reliability.^{20,21}

Early systems used short-range wireless protocols. Bluetooth-based technologies implemented by Asadullah and Ullah¹² and Thombare *et al.*¹⁴ are limited to about 10 m^{20,22}, which is often insufficient for multi-story residences or larger architectural footprints. Similarly, IR control is low-cost and reliable but requires line of sight.^{11,23} These approaches lack the bidirectional communication needed for environmental monitoring.

To overcome range limitations, Rakib *et al.*²⁴ and Tamakloe and Kommey²⁵ used GSM modems (SIM800/SIM900) for SMS-based control. GSM supports long-distance operation via predefined SMS commands, effectively bypassing the range limitations of Bluetooth and the IR line-of-sight control protocol, but it suffers from latency and recurring operational costs compared to local wireless systems.^{24,25} Al-Doori *et al.*²⁶ used MATLAB to develop a graphical user interface to automate heating, ventilation, and air conditioning units and lighting controls without internet connectivity.

More recent IoT systems target global connectivity. Findawati *et al.*²⁷ built a NodeMCU-based controller with Telegram, adding fire, motion, temperature, and humidity sensors, but limited to four load controls. Stolojescu-Crisan *et al.*¹⁵ created the “qToggle,” a modular API on Raspberry Pi and ESP8266. Mehra *et al.*¹⁶ also used a Raspberry Pi. Sayeduzzaman *et al.*²⁸ combined Blynk and Google Assistant over Message Queuing Telemetry Transport (MQTT)/If This Then That protocols. Kadiyan *et al.*²⁹ furthered this development by combining both Wi-Fi and Zigbee with Transport Layer Security (TLS) encryption and Open Authorization protocols. Two recent papers^{17,18} used ESP32 and ESP RainMaker.

These IoT systems depend heavily on a stable internet. Netinani *et al.*³⁰ reported 91.5% voice accuracy but saw failures during network congestion. To reduce dependence on the internet, Eduku *et al.*³¹ built an offline voice system using an Arduino Uno, though it experienced 2–3-s delays due to noise. Nkechinyere *et al.*¹⁹ used ESP32 and Python to improve offline processing. Reis and Serodio³² reviewed modern IoT architectures and mechanisms designed to address these gaps.

Table 1. Summary of literature review in comparison with the proposed work

Year	Authors	Methodology	Performance/Merits	Limitations
2017	Asadullah & Ullah ¹²	Arduino-based Bluetooth automation system for multi-device switching via smartphone	Low cost; high accessibility for residential control	Communication range restricted to about 10 m
2021	Bonia <i>et al.</i> ¹³	Universal IR remote integration with Arduino Uno for hardware-based appliance control	High reliability for line-of-sight tasks; minimal hardware investment	Lacks bidirectional feedback; requires unobstructed transmission paths
2021	Stolajescu-Crisan <i>et al.</i> ¹⁵	“qToggle” modular API solution utilizing Raspberry Pi and ESP8266 modules	Highly modular architecture; flexible API for diverse system integration	Requires significant technical expertise for setup and configuration
2022	Tamakloe & Kommey ²⁵	GSM-based remote control via SIM900 modem utilizing SMS command sets	Long-range control functionality independent of internet availability	Notable message latency and recurring operational costs (SMS)
2024	Sayeduzzaman <i>et al.</i> ²⁸	NodeMCU and Blynk app integration via MQTT and IFTTT protocols	Facilitates global remote monitoring and voice assistant integration	High vulnerability to internet downtime and third-party server stability
2024	Al-Doori <i>et al.</i> ²⁶	Localized automation system driven by a custom-designed MATLAB GUI	Strong analytical visualization and robust local hardware control	No mobile accessibility; tethered to a local PC environment
2024	Kadiyan <i>et al.</i> ²⁹	Multi-protocol (Wi-Fi/Zigbee) secured hub using TLS and OAuth	Enhanced data encryption and multi-standard device support	Elevated initial hardware cost and complex interoperability
2025	Pajkos <i>et al.</i> ¹⁸	ESP32-based lightweight TLS 1.3 client for secure IoT communication	Optimized security for low-power microcontrollers; high encryption standards	Computationally intensive for basic 8-bit microcontrollers
2025	Eduku <i>et al.</i> ³¹	An offline voice recognition system using a dedicated Arduino module	Internet-independent operation; provides local auditory feedback	High sensitivity to ambient noise; limited command vocabulary; response latency (2–3 s)

Abbreviations: API: Application Programming Interface; GSM: Global System for Mobile Communications; GUI: Graphical User Interface; IFTTT: If This Then That; IoT: Internet of Things; IR: Infrared; MQTT: Message Queuing Telemetry Transport; OAuth: Open Authorization; PC: Personal Computer; TLS: Transport Layer Security.

Table 1 summarizes the recent studies in comparison with the proposed work. From this table, each existing approach has drawbacks for developing regions with unreliable internet. Bluetooth is limited to a 10 m range¹², similar to IR, which requires line-of-sight with a 10–15 m range. GSM covers longer distances but suffers from latency and SMS costs^{24,25}; cloud-based systems fail without internet access^{28,30}; TLS 1.3 is complex to implement on lowcost microcontrollers¹⁸; and offline voice recognition is sensitive to noise.³¹ None of these options provides a practical balance between online convenience and offline reliability, especially for households that cannot afford expensive hardware or stable broadband.

In response to these challenges, this work introduces a low-cost, hybrid IoT home automation system using the ESP32 v4 microcontroller and the ESP RainMaker platform. It combines Google Assistant, the ESP RainMaker app, an

IR remote, and manual switches, and works both online and offline. The design prioritizes local control as the fallback, so users never lose the ability to turn lights or fans on or off, even during prolonged internet outages.

The remaining part of this work is organized as follows: Section 2 presents the system architecture and methodology, including hardware implementation, system flow chart, software design, and the hybrid control flow. Section 3 reports the experimental results, performance evaluation, and the sixmonth field trial. This includes measured response times, success rates across different network conditions, projected energy savings, the system’s theoretical lifespan, and a comparison with existing systems. Section 4 concludes the paper, summarizes the main contributions, acknowledges limitations, and outlines directions for future work, such as adding cybersecurity features and automated energy metering.

2. System architecture and methodology

The system's design prioritizes modularity and local-first reliability to maintain functionality during internet outages common in developing regions.^{5,6} The system integrates multiple control and sensor interfaces with household appliances. Each subsystem (control, sensing, actuation, user interface) can be tested independently, and the modular layout makes it easy to replace a single faulty sensor or relay without redesigning the whole board.

2.1. Hardware implementation and control unit

The hardware core is an ESP32V4 microcontroller and development board with builtin WiFi, Bluetooth Low Energy (BLE) v4.2, and a dualcore Xtensa 32bit LX6 CPU (Cadence Design Systems, United States) that handles network and sensor tasks in parallel.^{33,34} A DHT22 (AM2302) measures temperature and humidity from -40°C to 80°C and 0–100% RH (accuracy ± 1)³⁵, and an HCSR501 passive IR (PIR) motion sensor provides a 7 m range, 120° field of view, and an adjustable delay from 0.3 to 200 s.³⁶ The PIR delay setting was kept at 1 s during the bench test to avoid missing short movements, but it can be increased in a home environment to prevent false triggers from pets or falling objects.

For user interaction, the online options were the Google Voice Assistant and the ESP RainMaker mobile app; offline controls consisted of a KY022 VS1838 IR remote (10–15 m lineofsight)³⁷ and standard physical switches. The IR remote uses the NEC protocol, which the VS1838 receiver decodes reliably even under moderate room lighting. A multichannel 5 V relay module connects the ESP32's lowvoltage direct current signals to 220 V alternating current household loads. The relay was rated for 10 A at 250 VAC or 8 A at 30 VDC, with a switching response time of 5–10 ms (max).³⁸ Each channel was optically isolated, reducing electrical noise feedback to the microcontroller. The total component cost (ESP32, sensors, relays, IR receiver, cables) was under \$55 USD, keeping the system affordable for its target users.

2.2. The system functional flow chart

The proposed system has three main functionalities: multi-modal load control, environmental monitoring, and motion-based security. When powered on, the ESP32 microcontroller begins its initialization phase to configure sensor I/O pins and the relays, restoring previous load states. This setup was conducted before performing a firmware-level network connection. If cloud access is available, it enters online mode and synchronizes with the ESP RainMaker for remote monitoring and Google Assistant voice control. Otherwise, it fails over to offline mode to maintain local functionality through the IR remote and manual switches. Simultaneously, the DHT22 sensor loop reads the temperature and activates the fan when the temperature exceeds the theoretical 29°C threshold used in this work. The PIR sensor monitors motion and triggers the security lights and alerts. These functions operate

independently of network status.

When multiple control commands are received simultaneously for the same load, the system processes them in the order they arrive, with no hardcoded priority among manual switches, IR remote, voice commands, or app toggles. The most recently received command takes effect. Temperaturecontrolled fan automation and PIR security alerts operate independently and do not conflict with manual or wireless load control. **Figure 1** shows the system flow: network connection decision, multimodal load control (voice, app, IR, manual), and parallel monitoring of temperature and motion. A lost connection never blocks local control.

2.3. Software architecture and ESP RainMaker platform

The firmware was developed using Espressif's IoT Development Framework (ESP-IDF) and the ESP RainMaker platform. RainMaker is a serverless IoT solution built on Amazon Web Services that reduces maintenance workload for developers.³⁹ Key configuration steps include node and device mapping, security setup, and thirdparty integration. First, the ESP32 is defined as a single node, and each relay and sensor is defined as a device or parameter within that node. The mobile app User Interface updates automatically from the firmware configuration. Second, the initial setup uses BLE bonding for authentication, and TLS 1.3 encrypts cloud data for privacy and security. TLS 1.3 encrypts all data sent to the cloud, protecting user privacy and preventing unauthorized access. The prototype does not include advanced security mechanisms beyond ESP RainMaker's baseline TLS 1.3 and BLE bonding. It is designed for lowcost offline functionality, not for highsecurity applications. Third, the node is synchronized with the Google Home Graph. This enables Google Assistant to execute voice commands without requiring additional third-party bridges like If This Then That. With these steps completed, the mobile app dashboard will be as shown in **Figure 2**.

2.4. Physical design and circuitry

The circuit schematic in **Figure 3** and the two-dimensional printed circuit board (PCB) layout showing component placement and copper trace routing in **Figure 4** were designed using the Proteus software (version 8.16 SP3) with ISIS and ARES, respectively. **Figure 5** shows the three-dimensional representation of the final PCB design. The schematic diagram provides an overview of the system's connections and the interconnections among the components. It also describes the multi-modal capabilities of the system. To reduce complexity for traceable physical interpretation, wiring, and troubleshooting, the circuit was implemented on a pre-perforated board. We used 4channel and 8channel relay modules instead of designing a custom PCB, and all external connections (sensors, switches, IR receiver) use detachable connectors, making the whole system easy to repair or replace individual components (**Figure 6**). The ESP32 pins were connected to the relay inputs, the IR receiver, the sensors, and the switches.

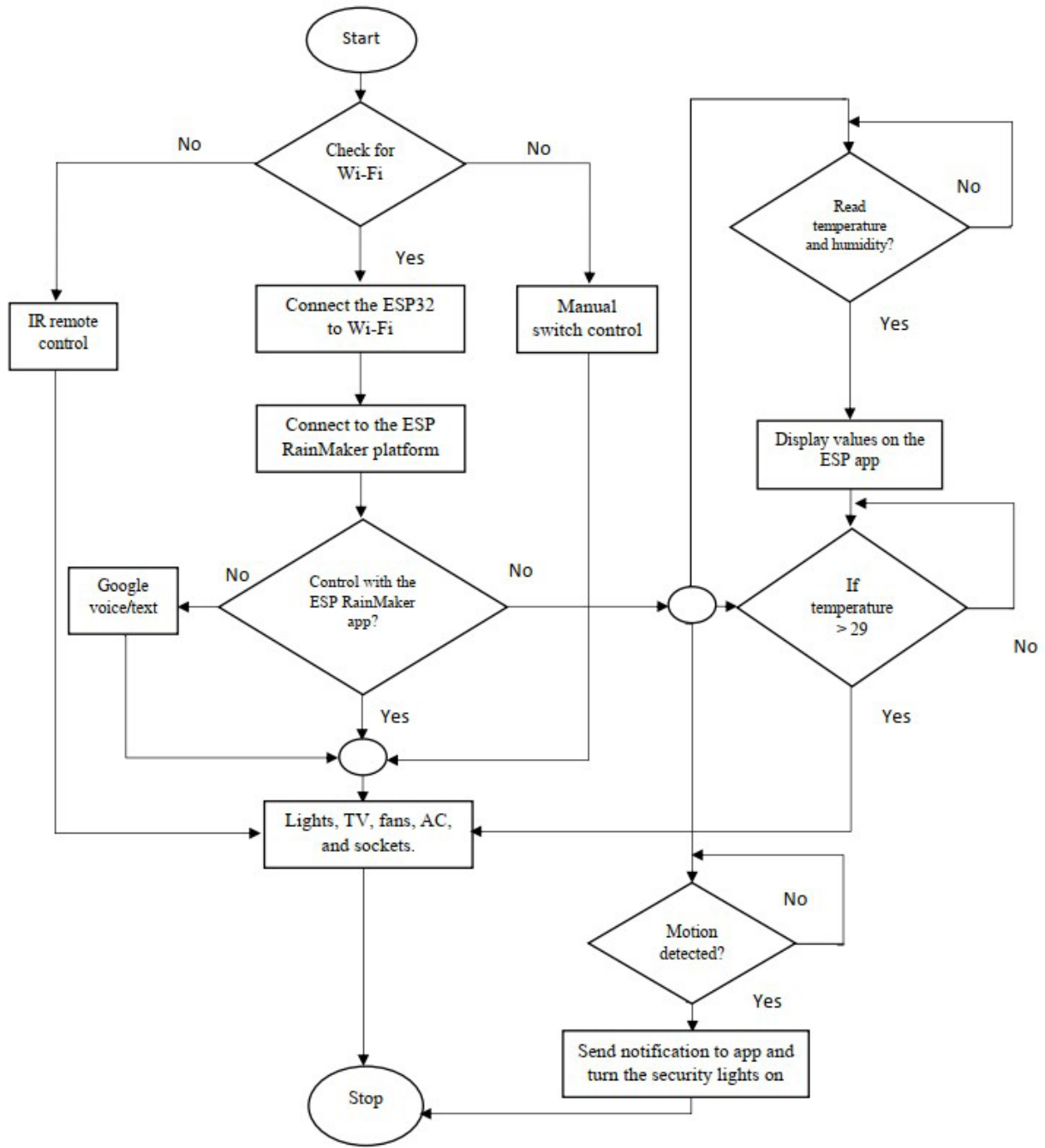


Figure 1. System functional flow chart showing initialization sequence (left), network connection decision tree and multi-modal control methods (center), and parallel monitoring loops for DHT22 temperature and passive infrared (PIR) motion sensing (right).

3. Results and discussion

The physical setup of the system underwent dual-phase evaluation: a seven-day bench test incorporating all functionalities (May 2024) and a subsequent six-month field trial (October 2024 to March 2025) in a residence in the southwest region of Cameroon.

3.1. Performance evaluation of the bench test prototype

The bench prototype simulated a standard residential environment consisting of a living area, kitchen, two bedrooms, and outdoor zones. As shown in **Figure 6**, the

physical setup included an ESP32V4 microcontroller, two relay modules (8channel and 4channel), a KYVS1838 IR receiver, an HCSR501 PIR motion sensor, a DHT22 temperature sensor, and indicator light-emitting diodes.

This layout implements the hybrid online/offline architecture described in Section 2 and the flow chart in **Figure 1**. The operation of the system is further divided into the following subsections (3.1.1 to 3.1.3), which cover three main functions, as shown in the flow chart (**Figure 1**): wireless control of lights and appliances, temperature and humidity automation, and motionbased security.

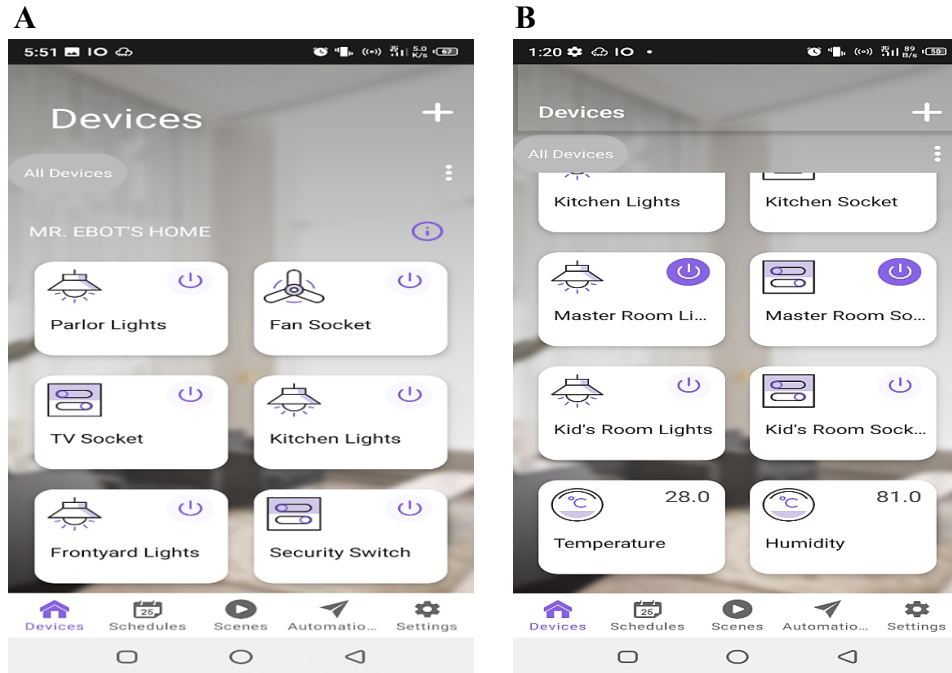


Figure 2. ESP RainMaker app dashboard showing the list of controllable devices with (A) ON/OFF toggles. (B) The top bar displays the temperature (28.0 °C) and humidity (81.0%). The bottom navigation bar provides access to schedules, scenes, automation, and settings.

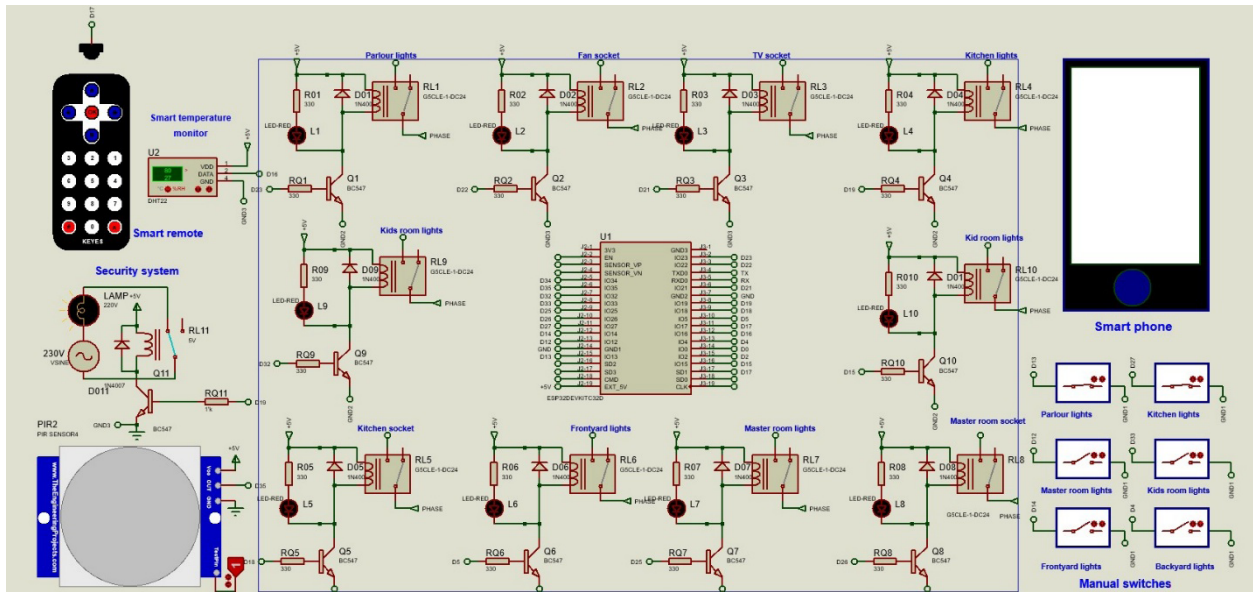


Figure 3. Detailed schematic diagram of the hybrid control system showing the interconnection between the ESP32 V4 microcontroller, the 4 and 8-channel relay, the KY-VS1838 receiver, the passive infrared (PIR) and DHT22 sensors, and the manual switches.

3.1.1. Hybrid control of the lights and appliances

The system has two control paths to ensure continuous operation regardless of network status. Online commands are executed by the Google Assistant and the ESP RainMaker app. The voice interface can check device states and turn multiple devices on/off either independently or all at once (Figure 7). Knowing which lights are on allows a user to turn off forgotten loads remotely, contributing to energy savings. The app also supports scheduled automation. As shown in Figure 8, a user can schedule the front-yard lights

to turn on at sunset and off at sunrise each evening. At any time, and especially during internet failures, the IR remote and physical manual switches work without needing the cloud.

3.1.2. Temperature and humidity environmental monitoring

The DHT22 sensor reads ambient temperature (°C) and relative humidity (%). It sends this data to the ESP RainMaker cloud in real time (Figure 9A). The app stores

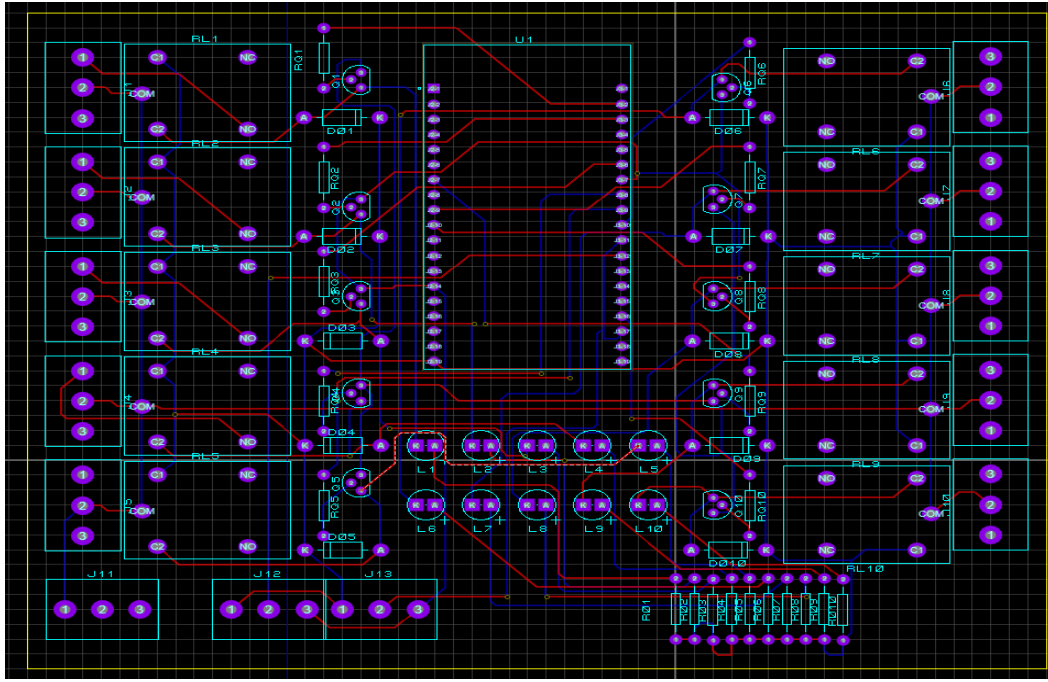


Figure 4. Two-dimensional printed circuit board layout design showing component placement and copper trace routing for the hybrid smart home controller.

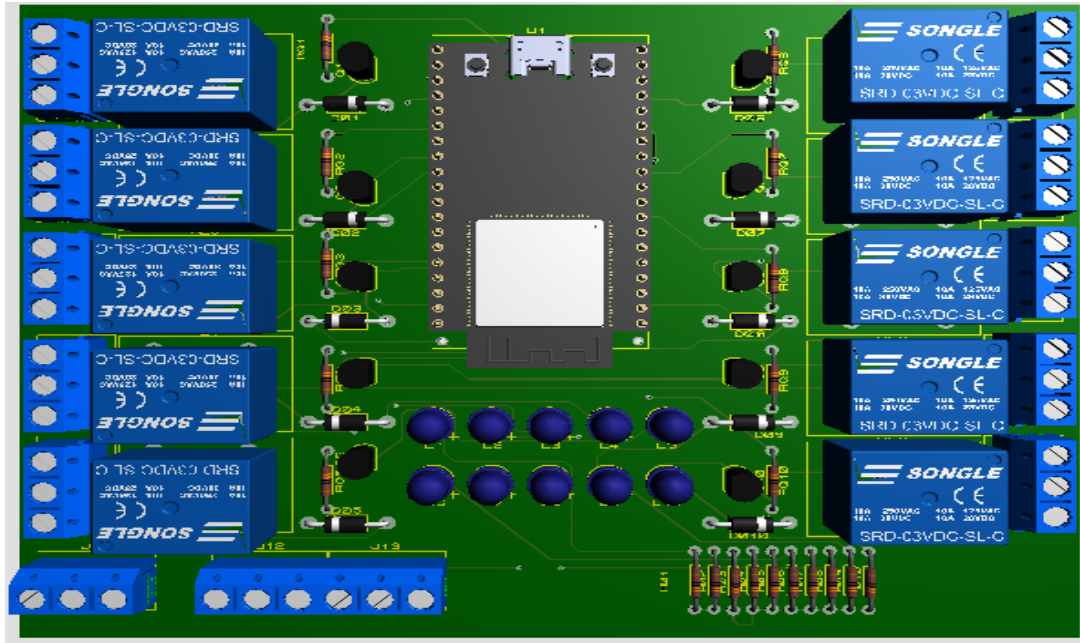


Figure 5. Three-dimensional visualization of the final printed circuit board design of the system, illustrating the physical arrangements of the ESP32 module, relay boards, sensors, connectors, and other components.

historical readings and shows daily, weekly, and seasonal trends as bar charts. The system turns on the fan when the temperature exceeds a userdefined threshold. Ideal room comfort is 21–24 °C, but for our tests the threshold was set to 29 °C (**Figure 9B**) because average May temperatures in Southwestern Cameroon reach 31 °C. Setting a lower threshold would have kept the fan running almost continuously.

Users have several ways to control the fan. A voice command like “It is too hot” to Google Assistant turns the fan on immediately, overriding the automatic threshold. The mobile app also provides a simple ON/OFF toggle for direct manual control. Users can adjust the temperature threshold in the app. Predefined schedules and timers were also tested, allowing the fan to run at specific hours regardless of temperature. This provided flexible management of thermal comfort.

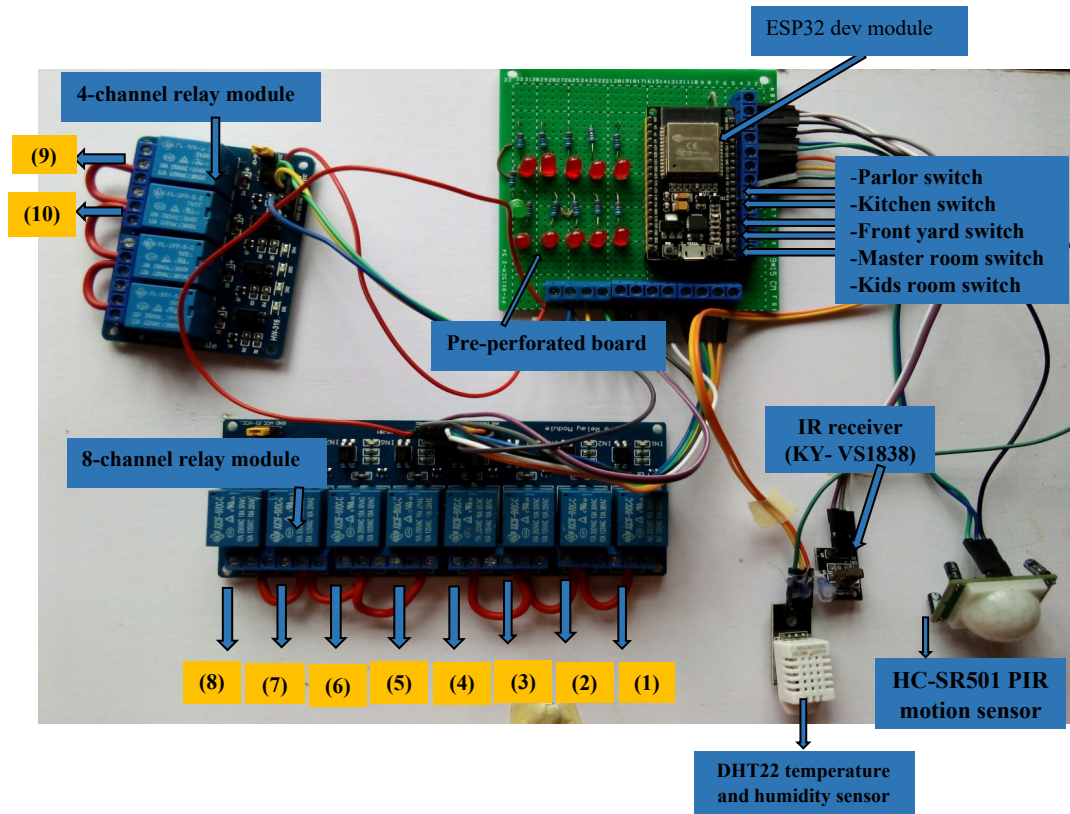


Figure 6. Bench prototype showing the physical connections of the main components of the smart home automation system and output pins labelled (1–10) for the alternating current load. The output pins connect to: (1) parlor lights, (2) TV socket, (3) fan socket, (4) kitchen lights, (5) front-yard lights, (6) security switch, (7) master room light, (8) master room socket, (9) kids’ room lights, (10) kids’ room socket.

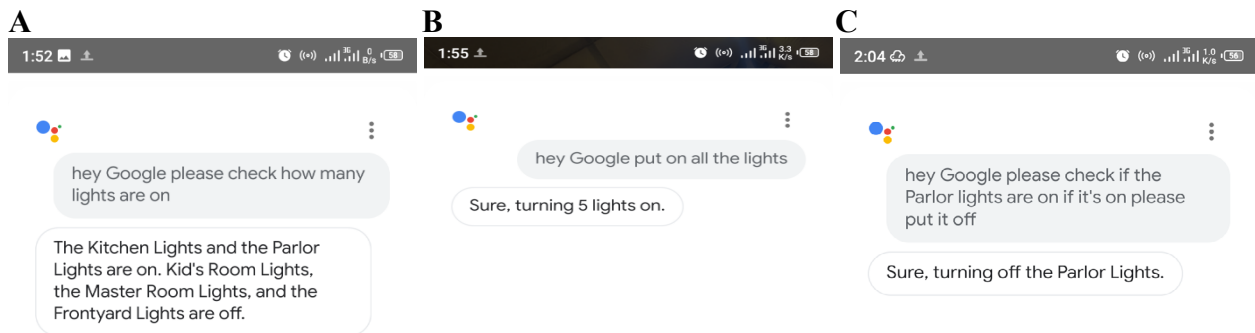


Figure 7. Google Assistant Voice control demonstration. (A) Verification status of how many lights are currently on. (B) Command to turn on all lights. (C) Command to turn off parlor lights after status verification.

3.1.3. The Motion-based security system

The security system is activated either manually via the security toggle switch in the mobile app (**Figure 10A**) or automatically by a scheduled timer (every day at 7:00 pm). When activated, the PIR motion sensor monitors the protected area within a range of 3–7 m (9–23 feet) and a 120° field of view.³⁷ Additional PIR sensors can be connected to cover other entry points or blind spots, extending security to the main angles of the home.

Upon detecting motion, the system triggers the security lights for 10 s and sends a push notification to the mobile

app predefined as “Security alert, Lights ON! Someone’s outside.” (**Figure 10B**). The lighting can be replaced with an audible alarm, depending on user preference. Motion detection and local lighting work even without internet, and the notification is sent later when connectivity is stable. All alert message data, including dates and timestamps, is stored in the app’s database.

3.2. Performance evaluation of the bench test prototype

3.2.1. Definition of network conditions

Network conditions were defined in terms of three states with respect to latency and packet loss. A normal network

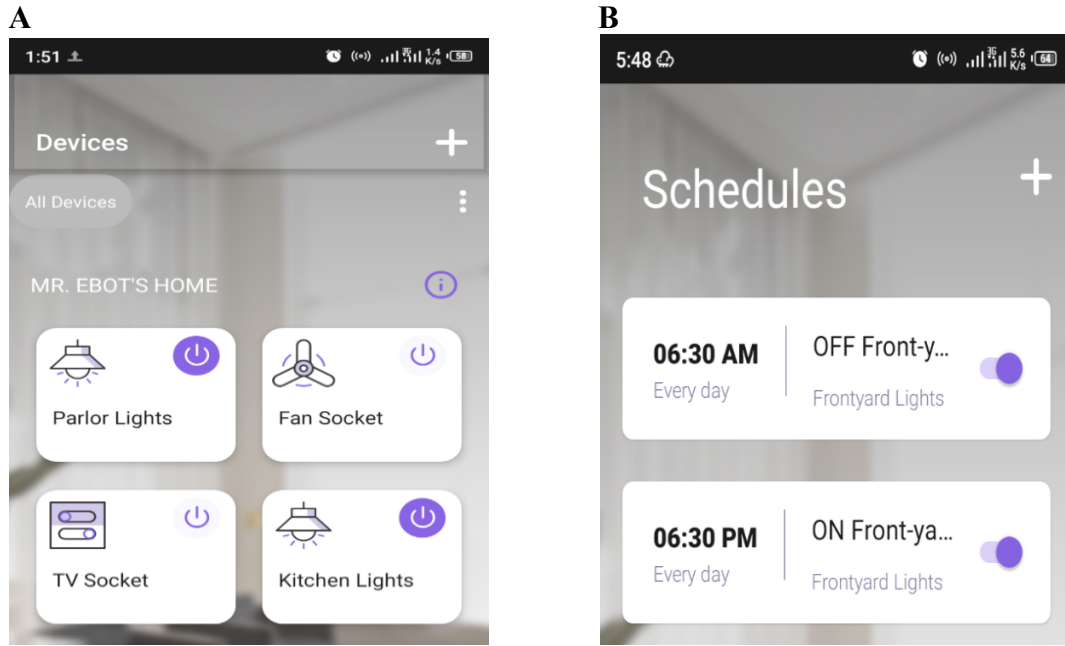


Figure 8. Wireless control and automation from the system's ESP RainMaker mobile app dashboard. (A) Control of the parlor and kitchen lights using ON/OFF toggle. (B) Scheduling interface showing a daily timer set to turn ON/OFF the front-yard lights.

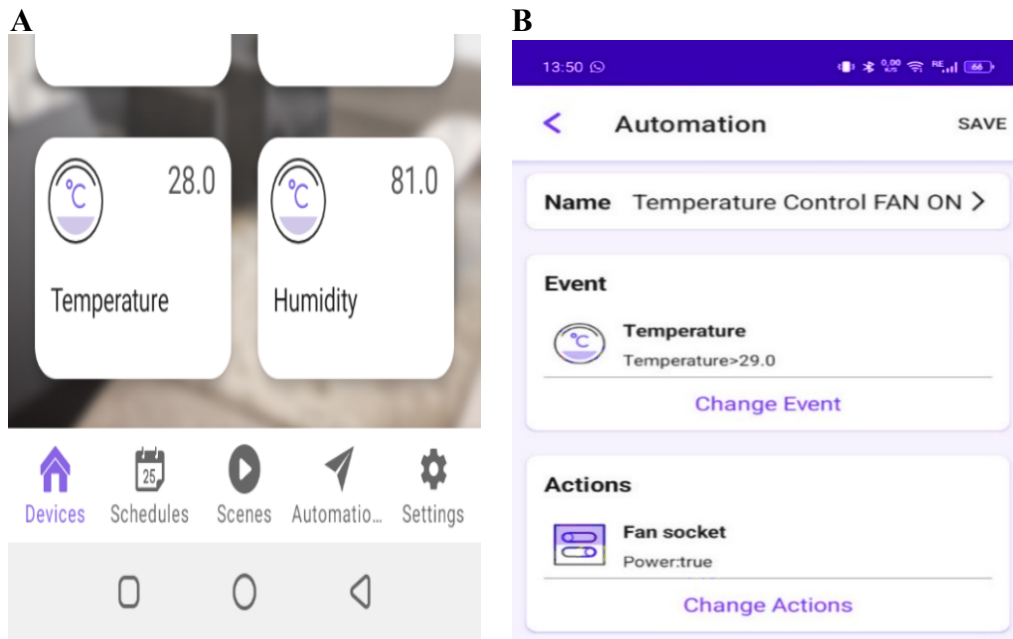


Figure 9. Environmental monitoring and automation from the ESP RainMaker app dashboard. (A) Real-time ambient temperature (28.0 °C) and humidity (81.0%) readings. (B) Temperature-controlled fan automation with threshold set at 29 °C.

means latency below 50 ms and packet loss below 1%, which is the typical stability benchmark for the ESP32.⁴⁰ These thresholds were chosen because cloudbased voice assistants like Google Assistant often time out above 50 ms of additional network delay. A degraded network means latency above 250 ms with packet loss between 5% and 10%; this occurred during peak congestion (7:30 pm–10:30 pm). These values correspond to the worst congestion observed during the field trial in Cameroon, when many households stream video or browse simultaneously. Offline

mode means no internet connection. When the network was degraded or offline, the system continued running by falling back to communication via 38 kHz IR pulses or local general-purpose input/output interrupts (via external manual switches).³⁷ The IR receiver reliably decodes signals from up to 12 m in line of sight, even through glass doors, which was sufficient for the test house layout. The test environment was standardized to meet a real residential setting, following recent indoor IoT practice.⁴¹ We also kept 5–8 other devices on the same network to create realistic traffic.

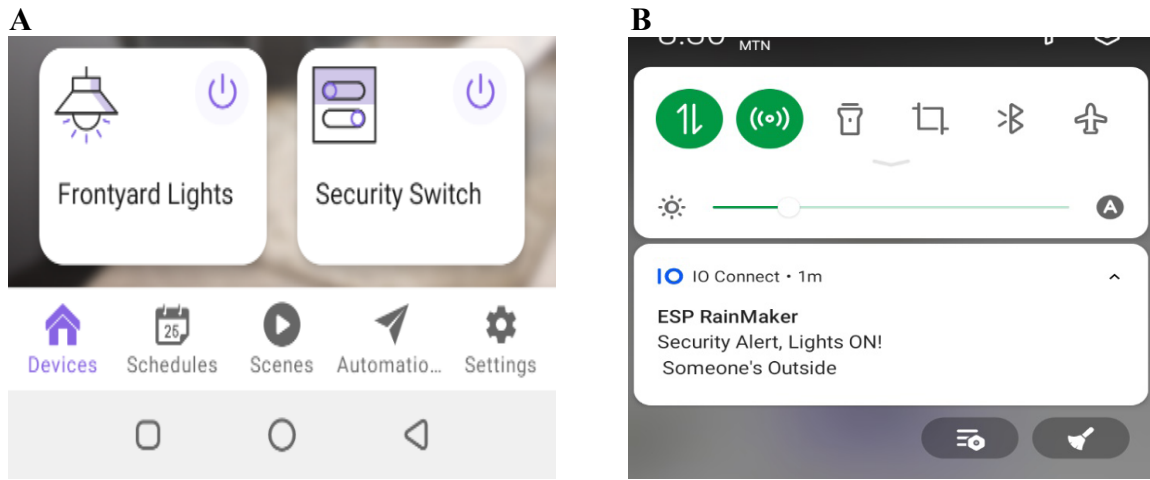


Figure 10. Security monitoring system in the ESP RainMaker mobile app dashboard. (A) The “Security Switch” toggle used to activate or deactivate the passive infrared motion sensor. (B) Security push notification received when motion is detected.

3.2.2. Average response time and latency analysis

The average response time $\bar{t}$ for each control modality was measured using manual timing tools over 25 trials per method ($n = 25$). Manual timing captures the delay that a user actually experiences, which is practical for this comparative evaluation. This method is similar to that used by Zuhelmi *et al.*⁴² They used a timer tool to manually record response times over 10 repeated trials per device. Additionally, Ahmed and Primajaya⁴¹ evaluated an ESP32-based early warning system with a digital stopwatch over 30 independent trials, measuring local response times and notification latency. The mathematical model for the analysis is defined by the arithmetic mean of the observed values, as given in **Equation 1**.

$$\bar{t} = \frac{1}{n} \sum_{i=1}^n t_i \quad (1)$$

where $\bar{t}$ is the average response time (seconds), n : number of measurements ($n = 25$), and t_i is the individual response time for the i -th trial (seconds).

Total system latency (SL) is the sum of local processing (t_p), network transmission (t_n), and cloud propagation (t_c).⁴⁰ For the Google Voice Assistant, the most complex modality, the measured components yielded a cumulative average of 1.56 s. The breakdown is as follows.

Cloud processing (t_c) took 0.67 s. This included speechtotext, natural language processing, texttospeech for confirmation, and API exchange between Google and Amazon Web Services. This matches the “CloudEdge Interaction” delay reported by Benaboura *et al.*⁴⁰ MQTT broker routing (t_n) added 0.45 s, covering the publish-subscribe delivery from the cloud to the ESP32 using a lightweight protocol suited for lowbandwidth links.³⁹ Local network and actuation (t_p) took 0.44 s. The ESP32 microcontroller sets a general-purpose input/output pin, the relay switches (max 10 ms according to³⁸), and the firmware sends an acknowledgment. For the IR remote, the average latency was 0.91 s, primarily due to decoding the 38 kHz NEC pulses. Manual switches were the fastest at 0.38 s, as they rely on direct hardware interrupts with no network or cloud delays. These short local latencies, below 1 s for IR and under 0.4 s for manual, are the reason the system feels responsive even when the internet is down. **Table 2** summarizes the average response time for each control method. Cloud-dependent methods are noticeably slower, while offline controls are much faster, which is a clear advantage of our system.

3.2.3. Command success rate and reliability analysis

During the seven-day field trial, we quantified the system reliability by recording the total number of command attempts (N_c) against the number of successfully executed

Table 2. Comparative average response time by control modality

Control mode	Communication protocol	Average response time (s)	Observations
Google Voice	NLP + Cloud Integration	1.56	Involves cloud-based STT/TTS
ESP RainMaker app	MQTT over TLS 1.3	1.43	Optimized via protocol buffers encoding
IR remote control	38kHz Infrared (Local)	0.91	Effective within line-of-sight
Manual switch	Hardware interrupt (GPIO)	0.38	Direct local actuation

Abbreviations: GPIO: General-purpose input/output; IR: Infrared; MQTT: Message Queuing Telemetry Transport; NLP: Natural language processing; STT: Speech-to-text; TLS: Transport Layer Security; TTS: Text-to-speech.

commands (N_s). The command success rate was calculated using Equation 2⁴¹:

$$S.R = \left(\frac{N_s}{N_t} \right) \times 100 \quad (2)$$

The success rate was 94.33% under normal network conditions. It dropped to 88.00% under the defined degraded network conditions. Offline mode (IR and manual) achieved 98.60%, with a small 1.2–2% failure rate due to ESP32 processing overhead, IR signal decoding time, and relay mechanical switching. Table 3 shows that success rates drop slightly under degraded network conditions but remain acceptable under normal conditions. Offline mode works reliably, which is a necessity when the internet is unavailable.

3.2.4. Theoretical energy savings projection

Smart Home Energy Management Systems demonstrate that integrating IoT-enabled control can reduce residential energy consumption by 10% to 30% compared to manual configurations.⁴³ The hybrid IoT system uses programmable scheduling and realtime remote monitoring. Based on these features and the system's operation, we estimate an average energy saving of 21%. As noted in recent 2025 energy efficiency studies, the effectiveness of such systems depends heavily on the transition from manual to autonomous regulation to ensure environmental sustainability.⁴⁴

3.2.5. Operational lifespan estimation (Theoretical model projection)

The projected operational lifespan of the system is estimated from the specifications of the main components and our system's performance, using manufacturer data and standard IoT platform benchmarks. The ESP32 dev module is rated for 20 years of operation (based on the GD25Q32 flash memory).^{34,45} The DHT22 and HCSR501 sensors are rated for 10–20 years.^{35,46} The 5 V relay module is rated for 50,000 to 100,000 switching cycles.³⁸

The electromechanical relay is the critical failure point for load control. To estimate its longevity, we conducted a community survey and found an average of eight toggles per day (f_{daily}). The annual switching frequency (f_{annual}) was projected using Equation 3:

$$f_{annual} = f_{daily} \times 365 \quad (3)$$

Given the relay's rated cycle life (N_{rated}), the theoretical lifespan (L) was determined using Equation 4:

$$L = \frac{N_{rated}}{f_{annual}} \quad (4)$$

Using the maximum rating of 100,000 cycles³⁸, the upper-bound lifespan was projected at 34.2 years. However, accounting for the 20-year flash endurance of the ESP32 and potential environmental degradation, the practical system lifespan was projected to be 17.1 years. This is a conservative theoretical mean and should be interpreted as a durability projection rather than an empirical result from accelerated life testing.

3.3. Performance analysis of the six-month field trial prototype

Following the bench test, the system underwent a six-month field trial from October 2024 to March 2025 in a residential setting in Southwest Cameroon. The goal was to observe the system under real-world environmental conditions. **Figure 11** shows the schematic of the trial module simulated in Proteus. The 16 × 2 LCD screen was added to display the temperature readings and Wi-Fi status (ON/OFF). The hardware components were an ESP32 v4 microcontroller mounted on a pre-perforated board, a DHT11 temperature and humidity sensor, an externally connected HC-SR501 PIR motion sensor, 8- and 4-channel relay modules, a VS1838B IR receiver, and the liquid crystal display screen connected to the board via connectors. The assembly was enclosed in a conduit box (**Figure 12**). The total cost of these components was under \$80 USD, confirming the system's lowcost claim for developing regions.

During the six-month trial, the system had no hardware failure or firmware instability. The MQTT delay stayed at 0.45 s, and the IR latency was 0.35 s under normal operations. The relays completed approximately 1,440 switching cycles without mechanical lag, which supports the 17.1-year projected lifespan. An 8channel relay controlled the main loads (parlor, kitchen, front yard, master room, and kids' room lights), while the 4channel relay handled the fan, the security lights triggered by the PIR sensor, and other auxiliary sockets. The module was powered by a 5 V, 2.0 A charger for the ESP32, sensors, and relay inputs. A red light-emitting diode indicated power, and a green light-emitting diode indicated active Wi-Fi. The unit was centrally located in the living room to ensure optimal

Table 3. Success rates at normal, degraded network, and offline conditions

Condition	Successfully executed commands	Total number of command attempts	Success rate
Normal network	23–24	25	94.33%
Degraded network	20–22	25	88.00 %
Offline mode	24–25	25	98.60%

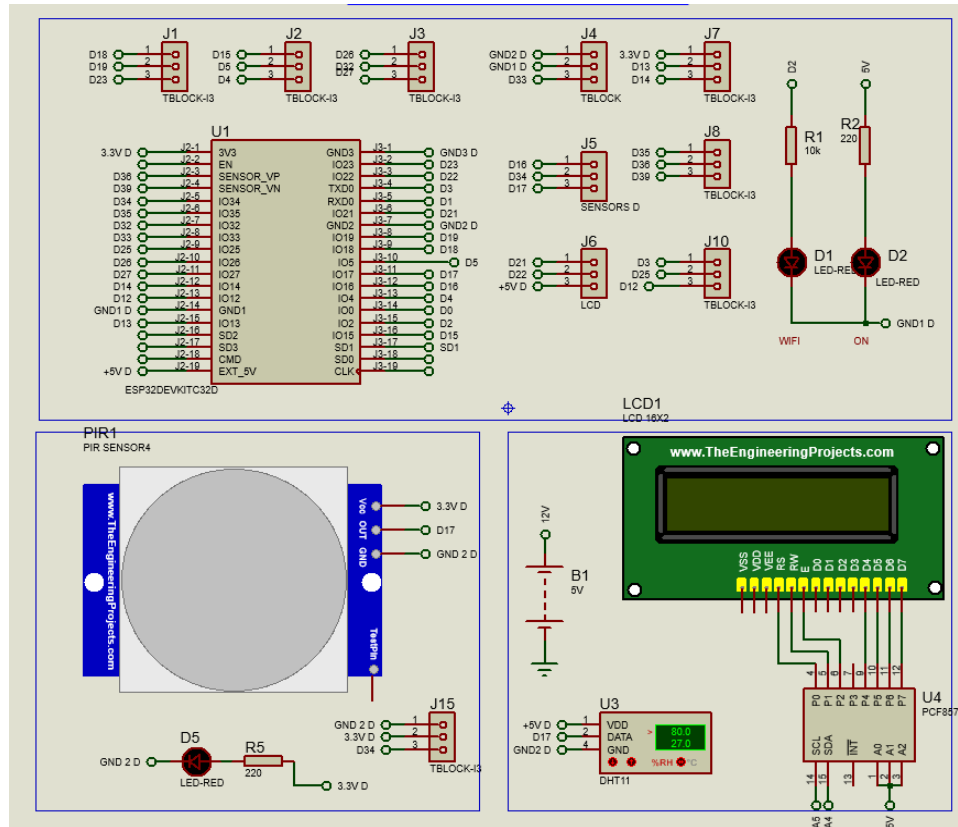


Figure 11. Proteus schematic of the six-month field trial module comprising an ESP32 v4 microcontroller, a DHT11 temperature and humidity sensor, an HC-SR501 passive infrared (PIR) motion sensor, connectors for the relays, and light-emitting diodes for indication of power and Wi-Fi availability.



Figure 12. The prototype of the six-month trial module enclosed in a conduit box with the ESP32 v4. Mounted on a pre-perforated board with other components.

signal distribution and easy access. This field trial module continued to operate correctly despite daily temperature variations ranging from 22 °C to 34 °C, occasional cooking-related dust, and humidity levels above 80%. No sensor drift or relay misbehavior was observed. **Figure 13** shows the trial module installed in the house.

3.5. Comparison of existing systems with the proposed system

This comparative analysis is necessary to demonstrate the need for a hybrid system with multi-modal control that

works both online and offline in places with unreliable internet. It is qualitative, not based on rerunning experiments from the cited works, but it highlights practical differences in design and reported performance. **Table 4** compares the proposed system with three common approaches from the literature: short-range wireless using Bluetooth and IR remote control, wide-area cellular using GSM, and cloud-only IoT platforms.

Most existing systems, as reviewed in the literature^{12,13,25,28,30}, rely on a single communication method. Bluetooth and IR operate well locally but do not cover



Figure 13. The field trial module installed in a residence in the Southwest Region of Cameroon, showing an ambient temperature reading (28 °C) on the 16 × 2 liquid crystal display screen, power availability (red light-emitting diode), and Wi-Fi status (green light-emitting diode).

Table 4. Comparison of existing systems with the proposed system

Feature	Proposed system	Existing works from the literature
Control methods	Hybrid (Online/Offline): Voice, ESP app, Sensors, IR remote, and manual switches	Bluetooth ¹² , SMS ²⁵ , IR only ¹³ , API ¹⁵ , app/cloud only ^{17,27,28} , offline voice ^{19,31}
Connectivity	Wi-Fi (RainMaker), IFTTT, Google Services	Bluetooth (10 m range) ¹² , GSM (Costly SMS) ²⁵ , IR (Line-of-sight) ¹⁷ , Wi-Fi/Zigbee ²⁶
Internet dependency	Partial: Works locally with IR/Manual during outages.	High: Most IoT systems ^{17,27,28,30} fail without internet. None: ^{19,31} but limited features
Latency (resp. time)	App: 1.43 s, Voice: 1.56 s, IR: 0.91 s; manual: 0.38 s	0.5 s Bluetooth ¹² , 2–3 s IoT/Blynk ²⁸ , 2–3 s (Noisy) offline voice ³¹
Success rate	Degraded Wi-Fi: 88.0%, Offline: 98.6%	Unstable; complete failure without internet in cloud-dependent models ^{28,29}
Energy management	21% reduction through scheduling	Mentioned generally 10-30% ^{43,44}
Lifespan/durability	17.1 years (projected by performance and system components).	Unreported in cited literature ^{12–32}

Abbreviations: API: Application Programming Interface; GSM: Global System for Mobile Communications; IFTTT: If This Then That; IoT: Internet of Things; IR: Infrared; SMS: Short Message Service.

an entire residential setting or provide remote access. GSM works over long distances but introduces delays and permessage costs. Cloud-only systems offer excellent features but require stable internet conditions, which is a problem in places where outages are common.

4. Conclusion

This paper presented a low-cost hybrid IoT smart home automation system using the ESP32 v4 microcontroller and the ESP RainMaker platform. The system combines four multi-modal controls through the ESP RainMaker mobile application, Google Assistant, IR remote, and manual switches. It also includes temperature and humidity monitoring for automated fan control, as well as a PIR-based security system that sends real-time mobile alerts. Experimental results from the seven-day bench test and six-month field trial showed an average response of 1.56 s for voice, 1.43 s for the app, 0.91 s for IR, and below 0.4 s for manual switches. The system achieved a 94.33% success rate with stable WiFi, 88.00% under degraded network conditions, and 98.60% in offline mode.

Beyond ESP RainMaker's baseline TLS 1.3 and BLE bonding, the prototype does not include advanced cybersecurity features. The reported values are average response times and success rates and are based on timing measurements, as in other IoT studies. No statistical dispersion was calculated. The energy saving of 21% and the lifespan of 17.1 years are theoretical projections based on the system's performance, such as autonomous scheduling and components' reliability, not direct measurements.

Future work may implement hardware-based security using a Secure Element for encrypted key storage and mutual TLS to prevent man-in-the-middle attacks. Automated energy metering using ACS712 and ZMPT101B alternating current and voltage sensors can be added to verify the projected 21% savings with real consumption data. By addressing the specific technical gap of internet dependence and the need for multimodal control that works both online and offline, the hybrid design ensures that essential functions remain usable during network outages. Through these semiautomated controls and monitoring, this system could improve the quality of life for elderly and physically challenged individuals in developing regions.

Acknowledgments

We would like to acknowledge the staff at the College of Technology, University of Buea, for providing guidance and laboratory facilities during the bench-testing phase of this research. Special appreciation is also extended to the household participants who facilitated the six-month field test in Cameroon.

Funding

None.

Conflict of interest

The authors declare they have no competing interests.

Author contributions

Conceptualization: Ebot Stanis Brian-Etta

Formal analysis: All authors

Investigation: All authors

Methodology: All authors

Writing—original draft: Ebot Stanis Brian-Etta

Writing—review & editing: Njitacke Tabekoueng Zeric

Availability of data

Some data are available from the corresponding author upon reasonable request.

AI tools statement

The authors confirm that AI tools were not used in this work and the preparation of this manuscript.

References

- Wang J, Zhang YC. A review on Internet of Things based smart home. In: *Proceedings of the 2021 International Conference on Culture-Oriented Science & Technology (ICCST)*. IEEE; 2021:273-277.
<https://doi.org/10.1109/ICCST53801.2021.00065>
- Li W, Yigitcanlar T, Erol I, Nguyen A. Motivations, barriers and risks of smart home adoption: from systematic literature review to conceptual framework. *Energy Res Social Sci*. 2021;80:102211.
<https://doi.org/10.1016/j.erss.2021.102211>
- Ezugwu AE, Taiwo O, Egwuche OS, *et al*. Smart homes of the future. *Trans Emerg Telecommun Technol*. 2025;36(1):e70041.
<https://doi.org/10.1002/ett.70041>
- Zhou H, Luo Y. IoT-based control systems for smart home automation: a comprehensive review. *Int J Res Adv Electron Eng*. 2024;5(1):17-19. Accessed April 24, 2026. <https://www.electrojournal.com/article/35/5-1-5-220.pdf>
- Del Rio DDF, Sovacool BK, Griffiths S. Culture, energy and climate sustainability, and smart home technologies: a mixed methods comparison of four countries. *Energy Clim Change*. 2021;2:100035.
<https://doi.org/10.1016/j.egycc.2021.100035>
- Kimutai MS, Omieno KO, Ondulo JM. Challenges and opportunities for smart homes deployment in developing countries: a case study of the user perspective in Kenya. *Open Access Lib J*. 2022;9(7):1-18.
<https://doi.org/10.4236/oalib.1107679>
- Ejidike CC, Mewomo MC, Olawumi TO, Wang S, Buniya MK. Barriers to the adoption of smart building technology in developing countries: an empirical survey. *J Constr Eng Manage*. 2025;151(6).
<https://doi.org/10.1061/JCEMD4.COENG-15466>
- Hassan A, Uddin N, Quddus A, Hassan S, Rehman A, Bharany S. Navigating IoT security: insights into architecture, key security features, attacks, current challenges and AI-driven solutions shaping the future of connectivity. *Comput Mater Contin*. 2024;81(3):3499-3559.

- <https://doi.org/10.32604/cmc.2024.057877>
9. Huang C, Chen Q, Lin L, Su W. The economic impact and application challenges of IoT technology in smart home and infrastructure. *J Innov Dev*. 2024;6(1):35-39.
<https://doi.org/10.54097/y54jpkp60>
10. Valencia-Arias A, Cardona-Acevedo S, Gómez-Molina S, Gonzalez-Ruiz JD, Valencia J. Smart home adoption factors: a systematic literature review and research agenda. *PLoS ONE*. 2023;18(10):e0292558.
<https://doi.org/10.1371/journal.pone.0292558>
11. Habib MR, Yusuf MA, Warnasuriya WMHN, Sunny K. A comprehensive review on the advancement of home automation system. In: *Proceedings of the 2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI)*. IEEE; 2024:638-642.
<https://doi.org/10.1109/ICoICI62503.2024.10696135>
12. Asadullah M, Ullah K. Smart home automation system using Bluetooth technology. In: *Proceedings of the 2017 International Conference on Innovations in Electrical Engineering and Computational Technologies (ICIEECT)*. IEEE; 2017:1-6.
<https://doi.org/10.1109/ICIEECT.2017.7916544>
13. Bonia K, Basan M, Sohkhlet N, Mawphlang CL, Selvaraj K. Controlling home appliances by IR remote control using Arduino Uno. *ADBU J Electr Electron Eng*. 2021;4(2):22-26.
<https://journals.dbuniversity.ac.in/ojs/index.php/AJEEE/article/view/2931/872>
14. Thombare N, Hendre S, Hinge V, Jadhav S, Sanas E. IoT based smart home automation system using Bluetooth. *Int J Sci Res Sci Technol*. 2025;12(5):55-59.
<https://doi.org/10.32628/IJSRST2513116>
15. Stolojescu-Crisan C, Crisan C, Butunoi BP. An IoT-based smart home automation system. *Sensors*. 2021;21(11):3784.
<https://doi.org/10.3390/s21113784>
16. Mehra MS. IoT based smart home automation system using Raspberry Pi. *Int J Eng Res Technol*. 2022;11(7):232-236. Accessed April 24, 2026. <https://www.ijert.org/research/iot-based-smart-home-automation-system-using-raspberry-pi-IJERTV11IS070165.pdf>
17. Deva SY, Babu BSS, Sabarinath KH, et al. Smart home using ESP32 and ESP RainMaker. *Int J Res Appl Sci Eng Technol*. 2025;13(2):1416-1421.
<https://doi.org/10.22214/ijraset.2025.67083>
18. Pajkos J, Kupcova E, Pleva M, Drutarovsky M. ESP32 microcontroller based lightweight TLS 1.3 client for IoT applications. In: *Proceedings of the 2025 35th International Conference Radioelektronika (RADIOELEKTRONIKA)*. IEEE; 2025:1-6.
<https://doi.org/10.1109/RADIOELEKTRONIKA65656.2025.11008381>
19. Nkechinyere E, Iyoloma CI, Chima OB. Offline voice-controlled home automation system using Python and Esp-32. *Int J Multidiscip Res Publ*. 2025;8(5):55-60. Accessed April 24, 2026. <http://ijmrapp.com/wp-content/uploads/2025/10/IJMRAP-V8N5P38Y25.pdf>
20. Ahmed MR, Rahman MO, Hoque MJ. Smart home: an empirical analysis of communication technological challenges. *Eur J Eng Technol Res*. 2020;5(5):571-575.
<https://doi.org/10.24018/ejeng.2020.5.5.1905>
21. Khan S, Ali H, Shah Z. Systematic analysis of smart homes: current trends and future recommendations. *Cogent Eng*. 2024;11(1):2344452.
<https://doi.org/10.1080/23311916.2024.2344452>
22. Baker B, Woods J, Reed MJ, Afford M. A survey of short-range wireless communication for ultra-low-power embedded systems. *J Low Power Electron Appl*. 2024;14(2):27.
<https://doi.org/10.3390/jlpea14020027>
23. Ali ASA, Bao XN. Design and research of infrared remote control based on ESP8266. *Open Access Libr J*. 2021;8(4):1-14.
<https://doi.org/10.4236/oalib.1107314>
24. Rakib MAA, Rahman MM, Rana MS, Islam MS, Abbas FI. GSM based home safety and security system. *Eur J Eng Technol Res*. 2021;6(6):69-73.
<https://doi.org/10.24018/ejeng.2021.6.6.2580>
25. Tamakloe E, Kommey B. A smart GSM-based home electrical appliances remote control system. *IPTEK J Technol Sci*. 2022;33(1):1-8.
<https://doi.org/10.12962/j20882033.v33i1.12226>
26. Al-Doori VS, Abbas SQ, Kulikov O, Ismail MN. Home automation system with a GIU that is powered by Arduino and MATLAB. In: *Proceedings of the 2024 35th Conference of Open Innovations Association (FRUCT)*. IEEE; 2024:63-70.
<https://doi.org/10.23919/FRUCT61870.2024.10516380>
27. Findawati Y, Idris A, Suprianto S, Rahmawati Y, Suprayitno E. IoT-based smart home controller using NodeMCU Lua V3 microcontroller and Telegram chat application. *IOP Conf Ser Mater Sci Eng*. 2020;874(1):012009.
<https://doi.org/10.1088/1757-899X/874/1/012009>
28. Sayeduzzaman M, Hasan T, Nasser AA, Negi A. An Internet of Things-integrated home automation with smart security system. In: *Automated Secure Computing for Next-Generation Systems*. Wiley; 2024:243-273.
<https://doi.org/10.1002/9781394213948.ch13>
29. Kadiyan V, Kaur G, Arti, Narang A, Rudola K. IOT based smart home automation system. *SSRN*. 2024. Accessed May 29, 2026. <https://ssrn.com/abstract=4910700>
30. Netinant P, Utsanok T, Rukhiran M, Klongdee S. Development and assessment of internet of things-driven smart home security and automation with voice commands. *IoT*. 2024;5(1):79-99.
<https://doi.org/10.3390/iot5010005>
31. Eduku S, Sekyi-Ansah J, Yeboah S, Appah IK. Design and implementation of a voice-controlled smart home automation system. *Int J Innov Technol Explor Eng*. 2025;14(8):5-12.
<https://doi.org/10.35940/ijitee.F1093.14080725>
32. Reis MJCS, Serôdio C. IoT architecture for smart environments: mechanisms, approaches, and applications. *Future Internet*. 2026;18(4):182.

- <https://doi.org/10.3390/fi18040182>
33. Pratama EW, Kiswantono A. Electrical analysis using ESP-32 module in realtime. *JEECS (J Electr Eng Comput Sci)*. 2023;7(2):1273-1284.
<https://doi.org/10.54732/jeecs.v7i2.21>
34. Espressif Systems. ESP32 Datasheet. Version 5.2. Published 2025. Accessed April 24, 2026. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
35. Choudhary V, Guha P, Pau G, Mishra S. An overview of smart agriculture using internet of things (IoT) and web services. *Environ Sustain Indicators*. 2025;26:100607.
<https://doi.org/10.1016/j.indic.2025.100607>
36. Wahyuni R, Rickyta A, Rahmalisa U, Irawan Y. Home security alarm using Wemos D1 and HC-SR501 sensor-based Telegram notification. *J Rob Control*. 2021;2(3).
<https://doi.org/10.18196/jrc.2378>
37. Esabunor T, Atikpakpa AA. Application of infrared technology in the design of a remote-controlled electric ceiling fan. *Innov J Eng Technol*. 2022;3(3):55-77. <https://journals.rasetass.org/index.php/ijet/article/download/206/202/>
38. Ningbo Songle Relay Co., Ltd. SRD-05VDC-SL-C Datasheet. Accessed May 28, 2026. <https://www.alldatasheet.com/datasheet-pdf/pdf/1132639/SONGLERELAY/SRD-05VDC-SL-C.html>
39. Espressif Systems. ESP RainMaker Documentation: Cloud-based IoT Solutions for ESP32. Published 2024. Accessed May 28, 2026. <https://rainmaker.espressif.com/docs/>
40. Benaboura A, Bechar R, Kadri W, Ho TD, Pan Z, Sahmoud S. Latency-aware and energy-efficient task offloading in IoT and cloud systems with DQN learning. *Electronics*. 2025;14(15):3090.
<https://doi.org/10.3390/electronics14153090>
41. Ahmed A, Primajaya A. Design and experimental evaluation of an ESP32-based IoT early warning system for LPG gas leakage. *Jurnal Janitra Inf Sist Inf*. 2026;6(1):1-7.
<https://doi.org/10.59395/zfqb4n92>
42. Zuhelmi TP, Sulistiyanti SR, Setyawan FXA, Adnan AR. Smart home controlling and monitoring system using multiboard client-server Internet of Things (IoT). *J Eng Sci Res*. 2020;1(2):69.
<https://doi.org/10.23960/jesr.v1i2.19>
43. G K, Ediga P, S A, *et al*. Smart energy management: real-time prediction and optimization for IoT-enabled smart homes. *Cogent Eng*. 2024;11(1).
<https://doi.org/10.1080/23311916.2024.2390674>
44. Weerawan N, Suriyawong P, Samae H, Sampattagul S, Phairuang W. Optimizing residential energy usage with smart devices: a case study on energy efficiency and environmental sustainability. *Sustainability*. 2025;17(14):6359.
<https://doi.org/10.3390/su17146359>
45. Zhang H, Wu C, Zhang DW, Chen G. A new technology-adaptable design for high-endurance EEPROM. *Electronics*. 2025;14(4):712.
<https://doi.org/10.3390/electronics14040712>
46. Yulizar D, Soekirno S, Ananda N, Prabowo MA, Perdana IFP, Aofany D. Performance analysis comparison of DHT11, DHT22 and DS18B20 as temperature measurement. In: *Proceedings of the 2nd International Conference on Science Education and Sciences 2022 (ICSSES 2022) (Advances in Physics Research)*. Atlantis Press International BV; 2023:37-45.
https://doi.org/10.2991/978-94-6463-232-3_5