EAI Endorsed Transactions
on Internet of Things
Research Article EAI.EU

A Low-Cost Multi-Node Architecture with ESP32 Microcontrollers for Assisted Home Automation of Elderly and Disabled Residents

Paulo Cesar Pérez-Ayala[1], Lendy Delims Salvador-Boza[1], Michael Anderson Lopez-Lopez[1], Carlos Alejandro Soto-Oroya[1], Maritza Raquel Cabana-Cáceres[1], and Cristian Castro-Vargas[1,*]

[1]Escuela Profesional de Ingeniería Electrónica, Facultad de Ingeniería Electrónica e Informática (FIEI), Universidad Nacional Federico Villarreal (UNFV), Lima 15082, Perú

Abstract

INTRODUCTION: Elderly and disabled populations worldwide are growing faster than care systems can absorb them, creating strong demand for affordable home-automation tools that let residents remain safe at home without continuous caregiver presence. Most low-cost IoT prototypes reported in the literature cover only one or two functional subsystems and send alerts to a single recipient, leaving a practical gap for multi-subsystem, multi-recipient solutions.
OBJECTIVES: To design, build, and test a modular home-automation system that brings together four specialised sensor nodes—radio-frequency access control, ultrasonic proximity detection, combustible-gas monitoring, and rain-triggered physical protection—plus a fifth node combining multi-recipient alert coordination with remote lighting control across five areas of the dwelling, all over a coordinated wireless network with real-time alerts sent to multiple users simultaneously.
METHODS: A waterfall lifecycle combined with test-driven development guided component selection and firmware coding. Five ESP32 microcontrollers exchange data through a lightweight publish-subscribe protocol; each node handles its own control logic and actuator independently, while a companion mobile application allows remote supervision and manual override.
RESULTS: All four subsystems were functionally validated through virtual simulation followed by physical demonstration on a scale dwelling model: the RFID module granted or denied access correctly for authorised and unauthorised cards, the ultrasonic module opened and closed the back door in response to detected presence, the gas sensor triggered visual, audible, and remote alerts above the configured threshold, and the rain sensor activated the laundry-protection servo on detected precipitation. The Telegram notification channel delivered alerts to the user with an observed average response time of under two seconds. Total component cost came to approximately 73 United States dollars, far below comparable commercial products.
CONCLUSION: The five-node distributed architecture demonstrates functional feasibility and meets the cost target needed for practical use in urban homes in developing countries, and offers a reproducible testbed for future assistive-technology research that includes controlled reliability and latency testing.

Keywords: home automation; Internet of Things; ESP32; embedded systems; publish-subscribe protocol; assistive technology; elderly care; low-cost prototype

Received on 01 May 2026, accepted on 21 June 2026, published on 29 June 2026

doi: 10.4108/12871

*Corresponding author. Email: ccastrov@unfv.edu.pe

1. Introduction

By 2030, the number of people aged 60 and over is expected to reach 1.4 billion worldwide, up from about 1.1 billion in 2023 [1]. Longer lives are welcome, but they place growing pressure on families and formal care services. The World Health Organization points out that age-related functional loss and disability frequently leave older adults isolated and dependent on others for basic safety tasks [2]. Smart-home systems built around sensor networks and automated alerts offer a concrete way to close this gap: systematic evidence shows they can detect hazards, trigger physical responses, and relay alerts to caregivers within seconds, reducing the need for round-the-clock human supervision [3]. For people living with physical disabilities, the same principle holds—remotely controlled locks, automated doors, and instant alerts can restore a meaningful degree of daily independence [4].

Energy considerations add another layer of motivation. Systematic reviews confirm that IoT technology applied to residential and commercial buildings can cut energy consumption by as much as 30% while also reducing operating expenses [5]. In developing-country contexts specifically, the lack of affordable home energy management tools means that a large share of residential consumption remains unoptimised, as demonstrated in low-income household studies in Latin America [6]. Bringing sensing and control into those homes would therefore benefit both occupants and national energy goals.

The pace of development in IoT-based home automation has accelerated noticeably over the past decade. Systems that once required expensive proprietary hardware can now be assembled from off-the-shelf components costing tens of dollars [7][8]. Wi-Fi-enabled microcontrollers and open API frameworks have lowered both the cost and the complexity of building multi-device home automation prototypes, since Wi-Fi connectivity removes the need for a dedicated gateway and the devices themselves are already widely available [9]. Within this landscape, the MQTT publish-subscribe protocol has become a standard choice for connecting distributed sensor nodes in real time [10]. Recent reviews and framework proposals show that smart home research is moving steadily toward architectures that combine health monitoring with everyday automation, providing broader service coverage within a single integrated platform [11][12].

On the hardware side, the ESP32 microcontroller has established itself as a practical building block for such systems. Its dual-core processor runs at 240 MHz and includes native 2.4 GHz Wi-Fi [14], while its unit cost in the Peruvian market sits below 30 soles (roughly 8 USD), making it accessible for low-budget prototype development. These figures explain why ESP32-based designs now dominate recent low-cost automation literature. Studies confirm that an ESP32 node can complete a full sensor read, process the value, and transmit real-time data via Wi-Fi in well under one second [34], and edge-computing benchmarks using MQTT on the same hardware show similarly low transmission latency [13]. That speed matters when a gas alarm or an open door must trigger an immediate response for a resident who cannot move quickly.

Figure 1 shows the breakdown of global electricity consumption by sector. Residential use stands out as the segment with the greatest untapped potential for improvement through automation. This figure provides context for why low-cost distributed architectures, built specifically for ordinary homes in developing countries, deserve dedicated engineering attention.

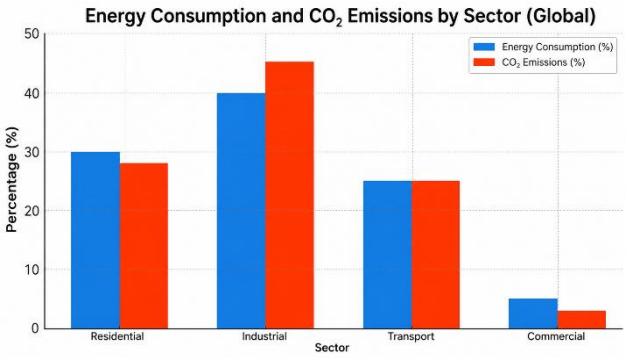

Figure 1. Global electricity consumption by sector. The residential segment holds the greatest potential for improvement through IoT-based automation [5]

Despite growing component availability, published low-cost IoT prototypes for assisted living continue to show a recurring pattern of limitations: they address at most two subsystems, they rely on passive monitoring rather than physical actuation, and they send notifications to only one recipient [15]. Combining real-time gas detection, RFID door control, ultrasonic proximity sensing, and rain-responsive laundry protection in a single hardware system—while keeping total cost below 80 USD—has not been reported in the peer-reviewed literature for developing-country residential settings [16][17].

The contribution of this paper is not any single sensor, protocol, or board in isolation—each of those is commercially available and individually well documented in the literature reviewed in Section 2. The contribution is an architectural pattern absent from that literature: a fully decentralised, per-node decision topology in which each of four independent sensor-actuator nodes runs its own control logic and acts locally before any network message is sent, combined with a dedicated coordinator node that fans alert delivery out to an entire caregiver group rather than a single recipient, all assembled within an 80 USD budget. Section 2.3 shows that no reviewed prior system combines local-decision actuation, multi-recipient alerting, and this subsystem coverage at this cost point, and Section 2.4 situates this contribution specifically against prior ESP32-based and assistive-IoT work published in this journal; Section 3 describes how that pattern was

implemented, and Section 5.4 details the controlled-testing work still required to validate it quantitatively. This paper describes the design, implementation, and laboratory evaluation of that architecture, built from five coordinated ESP32 nodes covering four sensor subsystems. The rest of the paper is organised as follows: Section 2 reviews closely related prior work; Section 3 explains the methodology and system design; Section 4 reports test results and prototype costs; Section 5 discusses advantages, limitations, and directions for future work; Section 6 states the conclusions.

2. Literature Review

Low-cost home automation research has shifted steadily from wired, single-board systems toward wireless, multi-microcontroller architectures [7][9]. The following subsections group the most relevant prior work by platform, highlighting what each generation achieved and where each fell short.

2.1. Arduino-Based Systems

Jabbar et al. produced a fully wired smart home prototype on an Arduino Mega with a Wi-Fi card and an Android companion app, covering temperature, humidity, and motion sensing with remote appliance switching [18]. All decision logic ran on a single microcontroller, however, and the design included no actuated safety response and no caregiver notification path. Dinmohammadi et al. moved to a Raspberry Pi as the central unit, pairing it with multiple environmental sensors to monitor residential energy use and safety conditions in real time [19]; the architecture was effective under normal operation but depended entirely on the hub remaining online—a failure there would leave sensors unread and alarms unsent. Benítez-Pina pursued a tighter focus on energy savings and reached comparable measurement accuracy [20], yet the same hub-dependency recurred: when the central device went offline, the system lost all remote visibility.

2.2. ESP32-Based Systems

A notable departure from pure sensor-based designs is the work of Villegas et al., who coupled a Siemens programmable logic controller with an ESP32 to offer dual-mode control—manual pushbutton and voice commands via a smart assistant [21]. Manual operations reached 100% reliability and voice-controlled actions exceeded 95%, though response times differed measurably between the two modes. Chandrashekar et al. targeted smart-building management with an ESP32 node that combined PIR occupancy sensing, LDR-based lighting, and ultrasonic water-level monitoring with cloud dashboards through ThingSpeak [22]; the system functioned well in isolation but showed instability when deployed in radio-dense environments. Taking a messaging-centric approach, Hutajulu et al. let residents send plain-text commands to an ESP32 via a chat application, keeping hardware costs low and setup simple [23]—though the design was limited to a single notification recipient, which rules it out for households where several caregivers need simultaneous alerts.

2.3. Multimodal Systems

On the protocol side, extending a home Wi-Fi network through cellular IoT and serverless MQTT functions is technically feasible, though it adds recurring data and infrastructure costs that local Wi-Fi avoids [12]. Hardware latency is not the limiting factor: dedicated performance studies show that ESP32 nodes transmit MQTT messages in real time with sub-second end-to-end delay under typical indoor Wi-Fi conditions [33]. The most capable prior system reviewed is that of Attar et al., who combined gas and air quality sensors with a Telegram chatbot to relay security and environmental alerts via messaging [24]. That design worked well in testing, but carried a structural weakness: all remote functions depended on the same internet connection, so a router outage silenced every notification channel at once.

2.4. Prior Work in EAI Endorsed Transactions

Two prior contributions published in EAI Endorsed Transactions on Internet of Things help situate the present work within the journal's own record. Martinez-Blanco et al. designed an ESP32-based smart electrical meter with an embedded web application for residential power monitoring, demonstrating the platform's suitability for affordable home instrumentation; the system addressed energy consumption alone, with no actuation, alerting, or accessibility focus [35]. Closer to the assistive-technology angle of this paper, Kanna et al. proposed an IoT-based smart assist module aimed at supporting paralysed patients, centred on a single sensor-triggered assistance function for one user need [36]. Neither EAI-published system combines multiple safety-relevant subsystems, multi-recipient alerting, and local-decision actuation within a single low-cost architecture; that combination, assembled within an 80 USD budget, is the specific gap this paper addresses relative to the journal's existing body of work, in addition to the broader gap identified against the international literature reviewed in Sections 2.1–2.3.

What the reviewed literature lacks is a single prototype that combines per-user RFID door control, proximity-triggered actuation with a safe closing delay, physical rain protection, and alerts sent to every caregiver at the same moment—while staying within the budget of a developing-country household. Table 1 places the most relevant systems side by side to make that gap visible.

Table 1. Comparative summary of prior low-cost home automation systems

Ref.	Platform	Connectivity	Notification	Main limitation
[18]	Arduino Mega	Wi-Fi	None	No safety actuation or alerts
[19]	RPi + sensors	Wi-Fi	None	Central server required
[21]	PLC+ESP32	Wi-Fi	Voice/App	No sensing or alerting
[23]	ESP32	Wi-Fi	Messaging	Single recipient only
[24]	ESP32	Wi-Fi	Telegram	Single recipient only
This work	5 × ESP32	Wi-Fi/MQTT	Multi-user	—

3. Methodology

3.1. Development Process

The project used a waterfall lifecycle, progressing through discrete phases in a fixed order with a concrete deliverable at the end of each one. Test-driven development was layered on top, so every firmware function was preceded by a written test case. The five phases were: (1) requirements—capturing functional needs (gas alarm, card-based door control, proximity door trigger, rain retraction) and non-functional targets (alert latency under two seconds, unit hardware cost under 35 USD, operation without a central server); (2) design—block diagrams, per-node flow diagrams, wiring schematics, and component selection based on datasheet review and market availability; (3) implementation—firmware coding in the Arduino IDE and MQTT topic structure definition; (4) verification—unit tests on each sensor node followed by integration tests on the assembled physical model; and (5) maintenance—threshold recalibration, latency tuning under simulated weak-signal conditions, and debounce adjustment.

Figure 2 shows the waterfall phases as applied to this project. Completing each phase before starting the next meant that design errors were caught on paper rather than in wiring, which saved considerable time during the integration phase.

Figure 2. Waterfall lifecycle phases applied to the proposed system development

Figure 3 shows the test-driven loop used for each sensor subsystem. Writing the acceptance test first forced the team to state precisely what "correct behaviour" meant for each threshold and timing parameter before any firmware was written.

Figure 3. Test-driven development cycle applied to each sensor subsystem

3.2. System Architecture

The system uses three stacked layers, illustrated in Figure 4. At the bottom, the perception layer holds four specialised sensor nodes, each built around a dedicated ESP32 that reads its sensor, drives its actuator, and runs its control logic independently—no node needs to ask another for permission before acting. The middle communication layer uses MQTT over Wi-Fi; a fifth ESP32 combines two roles: it acts as notification coordinator, subscribing to alert topics from all sensor nodes and forwarding messages to the user group through the messaging service API, and it drives a seven-LED smart lighting subsystem covering the living room, kitchen, bedroom, bathroom, and patio, switched remotely through the same MQTT broker and the mobile application. The top application layer is the mobile interface for real-time monitoring and manual override; mobile applications integrating MQTT-based IoT monitoring with user control interfaces have been successfully deployed in residential settings for energy and safety management [25].

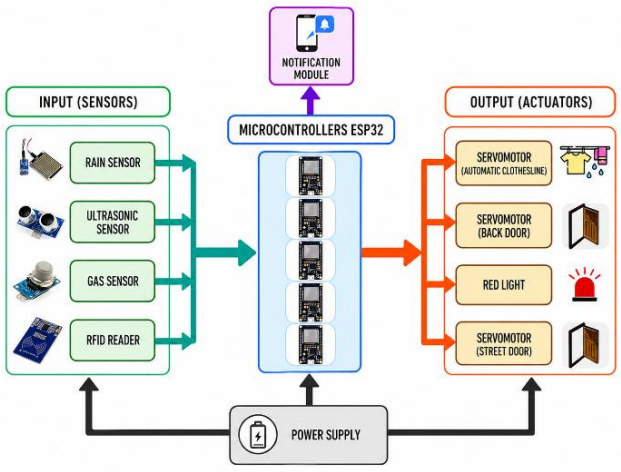

Figure 4. Block diagram of the proposed system general architecture

Figure 5 traces the path that an alert message takes from a sensor node to the end user. The coordinator node is the only component that needs an active internet connection to deliver notifications; the sensor nodes can still actuate locally if the network is unavailable.

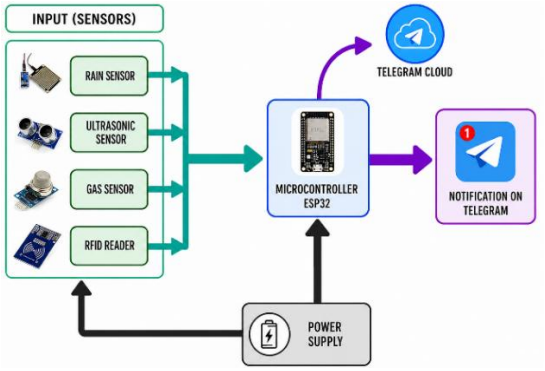

Figure 5. Notification module architecture: from sensor nodes to the remote user

3.3. System Flow

Figure 6 shows the top-level flow shared by all sensor nodes. On power-up, each node initialises its peripherals and connects to the Wi-Fi access point and the MQTT broker. It then enters a sensor-read loop. If the reading crosses the configured threshold, the node fires its actuator immediately and publishes an alert to the MQTT topic. The coordinator ESP32 picks up that alert and forwards it to all members of the shared messaging group. If the reading is within normal limits, the node publishes a routine status update and loops back. Crucially, the actuator response happens before the MQTT publish call, so a caregiver alert latency of over two seconds does not delay the physical safety action.

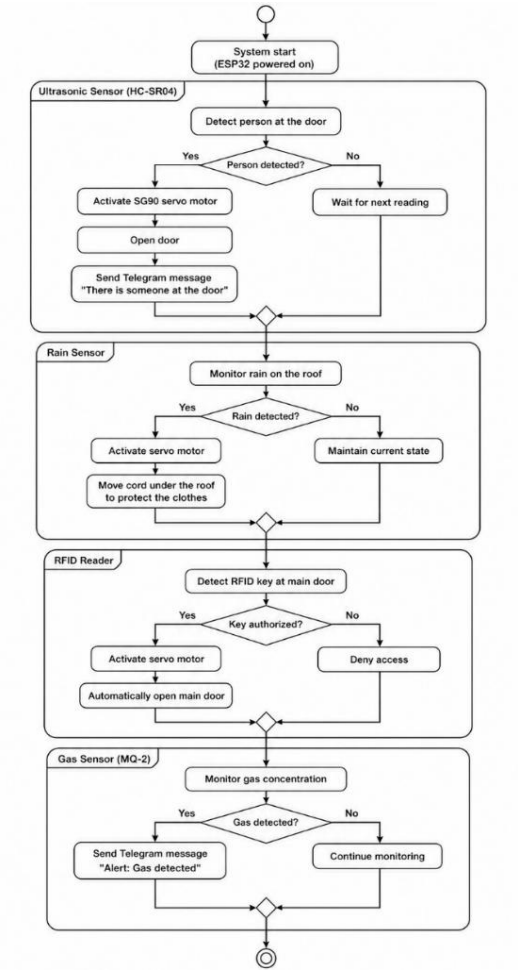

Figure 6. Proposed system general flow diagram. Every sensor subsystem follows this read-decide-actuate cycle

3.4. Hardware Components

Table 2 lists the main components and their key technical parameters. The ESP32 DevKit V1 serves as the processing and communication unit for all five nodes. Its 38 GPIO pins support analogue inputs, PWM outputs, and SPI buses simultaneously, which is why a single board can drive a servo, read a sensor, and maintain a Wi-Fi connection without a co-processor [14]. Performance evaluations confirm that this microcontroller completes sensor acquisition, local processing, and Wi-Fi data transmission in well under 500 ms, meeting the latency requirement set during the requirements phase [33][34].

Before any physical wiring was done, all four sensor subsystems were connected virtually in the Wokwi online simulator [31]. This step allowed the team to verify threshold logic and timing without risking hardware damage. Recent work on IoT education confirms that Wokwi-based pre-validation reduces integration errors and

accelerates firmware debugging compared with going straight to physical prototyping [30][32].

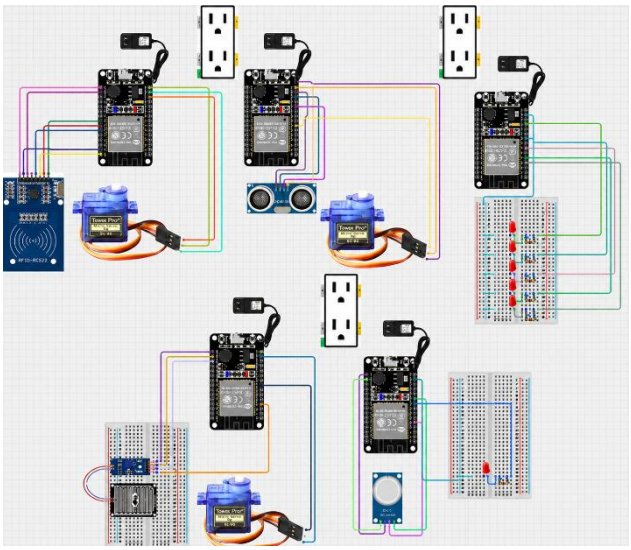

Figure 7. Joint simulation sketch of all four proposed sensor subsystems in the virtual environment

Table 2. Main hardware components and their key technical parameters

Component	Model	Key specification
Microcontroller	ESP32 DevKit V1	Dual-core 240 MHz, Wi-Fi, 38 GPIO
Ultrasonic sensor	HC-SR04	Range 2–400 cm, accuracy ±3 mm
Rain sensor	FC-37	Analogue 0–3.3 V and digital outputs
Gas sensor	MQ-2	LPG 300–10,000 ppm, response < 10 s
RFID reader	RC522	13.56 MHz, SPI, read range 0–5 cm
Servo motor	SG90 (×3)	50 Hz PWM, 0°–180° rotation
Lighting LEDs	LED (×7)	GPIO digital output, one per room/area

3.5. Subsystem Descriptions

3.5.1. Main Access Control (RC522)

The RC522 module operates at 13.56 MHz through the SPI bus. On each card presentation, it reads a four-byte UID and compares it with a list stored in the node's flash memory [29]. Cards whose UIDs are on the list trigger a 90-degree servo rotation, holding the door open for five seconds before it closes automatically. Cards not on the list leave the servo still and fire an immediate alert. Because a single swipe can generate several reads in quick succession, the node ignores further reads for two seconds after processing one, which prevents duplicate alerts from a single access event. This per-card identification logic follows the same approach successfully used in laboratory access control systems built on similar RFID hardware [26].

3.5.2. Proximity Detection (HC-SR04)

The HC-SR04 fires a 40 kHz ultrasonic pulse and measures how long the echo takes to return. Distance is then simply half of (echo time × speed of sound) [27]. The detection threshold is set between 1 and 7 cm, covering the clearance zone in front of the door frame. As long as the sensor continues detecting an object within that range, the door stays open. Only after three consecutive seconds with no detection does the servo close the door. That delay prevents the door from shutting on a resident who pauses momentarily in the threshold [27].

3.5.3. Combustible-Gas Monitoring (MQ-2)

The MQ-2 uses a tin-dioxide element whose electrical resistance drops when combustible gas is present. The node reads the resulting voltage through a divider circuit that steps the sensor's 5 V supply down to the 3.3 V range accepted by the ESP32 ADC [28]. When the reading exceeds the equivalent of 300 ppm of LPG, the node simultaneously activates the LED indicator, sounds the buzzer, and sends an MQTT alert containing the measured concentration. Recent multi-sensor IoT deployments using the same sensor confirm a typical response time under ten seconds at concentrations above the threshold, consistent with what was informally observed during our demonstrations [16].

3.5.4. Rain Protection (FC-37)

The FC-37 sensor functions as a voltage divider: moisture bridges the conductive tracks on its surface, lowering their resistance and shifting the output voltage. That shift is detectable in both analogue and digital modes [28]. A five-second debounce delay rejects isolated drops that do not represent sustained rain. When rain is confirmed, the coordinator publishes the event and the associated servo retracts the laundry line. Once precipitation stops, the system can restore the previous position either automatically after a configurable dry period or on a manual command from the mobile app.

3.5.5. Smart Lighting Control (Coordinator Node)

Beyond alert coordination, the fifth ESP32 drives a seven-LED lighting array covering five areas of the dwelling model: two LEDs in the living room, and one each in the kitchen, bedroom, and bathroom, plus two on the patio. Each LED is switched through a dedicated GPIO pin and toggled by a Boolean state published to its own MQTT topic, so each area can be controlled independently from the mobile application without affecting the others. Because lighting control shares the same MQTT broker and topic-naming convention as the four safety subsystems, no additional protocol or gateway was required to add this feature, illustrating the architecture's extensibility: a new functional area can be introduced by adding GPIO mappings and topics rather than by redesigning the communication layer. Figures 8 and 9 show the dedicated lighting-system panel and its physical wiring alongside the four sensor subsystems.

3.6. Communication and Notifications

All five nodes share a single Wi-Fi access point and connect to a public MQTT broker (broker.hivemq.com) at the library's default quality-of-service level; no broker-level authentication is configured, which Section 5.4 identifies as a deployment-readiness gap. The network topology is a star: every node publishes and subscribes independently to the same broker rather than to one another, so no single sensor node is a dependency for any other. Topics are structured as SensHogar/<subsystem>/<type>, where subsystem identifies the sensor (e.g. gas, rain, main_door) and type is one of state, value, alert, or control. Each node generates a randomised client identifier at connection time to avoid ID collisions on the shared broker, and both the Wi-Fi and MQTT connections are monitored by a reconnection loop that retries at a fixed interval if either link drops. Alert messages are additionally dispatched by the coordinator node through the Telegram Bot API, authenticated with a bot token and a fixed chat identifier obtained from Telegram's BotFather service; this channel is independent of the MQTT broker and continues to function if the broker connection is temporarily unavailable. Whereas prior designs limited notifications to one recipient [23], this system broadcasts every alert to an entire shared Telegram group, so any family member or caregiver in that group receives the message at the same time.

4. Results

4.1. Physical Prototype

The hardware was installed on a three-dimensional single-floor dwelling model designed in-house and 3D-printed in the laboratory of the Department of Electronic and Computer Engineering at Universidad Nacional Federico Villarreal. Testing followed two phases: virtual simulation in Wokwi [31], then physical integration tests on the assembled model. Virtual validation caught several threshold and timing errors that would otherwise have appeared only at the wiring stage.

Figure 8 shows the control unit with all five ESP32 boards mounted, wired, and labelled. Grouping the electronics on one panel simplified troubleshooting and made it straightforward to swap a node if it failed during tests.

Figure 8. Three-dimensional control unit of the proposed prototype with integrated electrical panel and wiring

Figure 9 shows the complete prototype on the scale dwelling model. Sensor placement followed the positions that would be used in a real ground-floor apartment, giving realistic wireless propagation distances and realistic actuator loads.

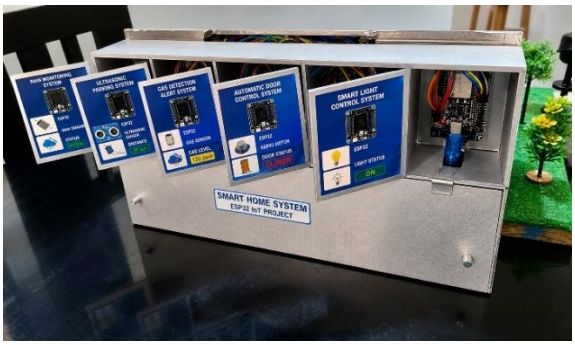

Figure 9. General view of the proposed prototype deployed on the scale dwelling model

4.2. Performance Metrics

Table 3 summarises the functional outcome observed for each subsystem during repeated informal demonstration on the scale dwelling model, run under stable domestic Wi-Fi conditions with the access point in the same room as the prototype. As noted below and revisited in Section 5.4, these are qualitative observations rather than a controlled, statistically powered measurement campaign; items marked (*) in the table reflect informal laboratory observation rather than a controlled, repeated-trial measurement, since no statistical reliability or latency campaign was conducted on the assembled prototype.

Table 3. Functional validation summary of the proposed system by subsystem

Subsystem	Aspect Observed	Outcome
RC522	Card authentication	Correctly granted/denied access in repeated demonstrations
RC522	Door-open duration	5 s (by design)
HC-SR04	Presence detection	Reliable door actuation within 1–7 cm range*
HC-SR04	Closing delay	3 s (by design)
MQ-2	Gas alert activation	LED, buzzer, and remote alert triggered above threshold*
MQ-2	Telegram alert latency	Average observed response under 2 s*
FC-37	Rain detection	Servo activated correctly on detected precipitation*
FC-37	False positives	None observed during informal testing*
MQTT network	Broker / QoS / authentication	Public broker, default QoS, no broker-level auth
Mobile app	Remote control	Functional manual override via Flutter app*

4.3. Cost Analysis

Table 4 gives a line-by-line breakdown of component costs. The total came to 274 Peruvian soles, which was approximately 73 USD at the exchange rate at the time of purchase. That figure places the proposed prototype well below the cheapest commercially available systems with equivalent multi-subsystem coverage found in the Lima market, which were quoted at between 600 and 800 USD at the time of the study (direct market survey, March 2025). Adding a new sensor subsystem in future work would cost roughly 38–45 soles more per node, given that the ESP32 (30 soles) dominates the per-node bill.

Table 4. Hardware cost breakdown for the proposed prototype

Component	Qty.	Unit (S/)	Total (S/)
ESP32 DevKit V1	5	30	150
MQ-2 gas sensor	1	5	5
SG90 servo motor	3	7	21
LED pack	1	5	5
FC-37 rain sensor	1	10	10
HC-SR04 sensor	1	5	5
RC522 RFID reader	1	8	8
3D scale model	1	50	50
Miscellaneous wiring	—	20	20
TOTAL	—	—	274 (≈ USD 73)

Relative to the cheapest commercial alternative identified above in the local market survey, the proposed prototype is roughly 8 to 10 times cheaper while covering more functional subsystems than any of the low-cost prototypes reviewed in Table 1; a chart-based comparison against the specific prior systems in Table 1 was omitted here, since their cost figures were not independently re-verified by the authors and a direct numeric overlay could misrepresent studies measured under different scopes and currencies.

5. Discussion

5.1. Advantages Over Arduino Platforms

The Arduino-based systems surveyed in Section 2 confirm that affordable home automation is achievable with commodity hardware. However, two structural gaps recur across all reviewed designs: no proactive notifications are sent to external caregivers, and no remote monitoring or control is possible from outside the local network. The proposed system closes both gaps without increasing the per-node cost. The architecture also

decouples local actuation from network notification by design: each sensor node fires its actuator immediately and only then publishes the corresponding MQTT message, so an alert path slowdown does not delay the physical safety response itself. A controlled measurement campaign quantifying this latency and reliability margin under varied network conditions remains a necessary next step, discussed in Section 5.4.

5.2. Differentiation from Prior ESP32 Systems

The closest prior work is Hutajulu et al. [23], which also pairs an ESP32 with a messaging application for home control. The functional difference is substantial: that system monitors and reports, whereas the proposed system additionally actuates. The access door opens autonomously when a resident approaches, the laundry line retracts upon rain detection, and the gas alarm activates before any manual intervention is required. Notification reach represents a second key distinction: the proposed architecture broadcasts alerts simultaneously to every member of a shared group, so a parent, a sibling, and a caregiver can all receive the same gas alert at the same instant rather than relying on a single designated recipient.

Compared with the most architecturally complex system reviewed, Attar et al. [24], the proposed design eliminates the single-board-computer gateway entirely. Each ESP32 node carries its own decision logic and actuates independently, so a failure in one subsystem does not propagate to the others. Table 1 already situates the proposed system against the reviewed prior work on a feature-by-feature basis; a quantitative head-to-head comparison was deliberately avoided here, since the cited studies were measured under heterogeneous hardware, network, and test-protocol conditions that a simple numeric overlay would misrepresent. A fair quantitative benchmark against representative baselines, reproduced under matched conditions, is identified as necessary future work in Section 5.4.

5.3. Relevance for Assisted Living

These outcomes carry direct relevance for the intended population. For a resident with limited mobility, a door that opens autonomously rather than requiring a physical key or button press offers a meaningfully different safety profile, particularly in emergency evacuation scenarios; quantifying that difference in opening speed under realistic conditions is left for the controlled testing described in Section 5.4. For households in the urban periphery of a developing city, routing gas and access alerts through an existing messaging application incurs no additional per-alert cost once the system is deployed [3][4]. Systematic reviews of healthcare-oriented smart home research consistently identify affordability and multi-recipient notification as the two features most strongly associated with sustained real-world adoption [11][15]. The proposed system addresses both requirements within the same architecture.

5.4. Limitations and Future Work

The most important limitation concerns the evaluation methodology itself. Testing to date has been functional rather than statistical: each subsystem was demonstrated repeatedly on a scale dwelling model and behaved as designed, but no controlled, repeated-trial campaign with logged timestamps, environmental variation, or formal accuracy counts was carried out. Reported timing figures such as the sub-two-second Telegram response are therefore qualitative laboratory observations, not measured statistics, and should be read accordingly. A planned next phase will instrument every node to log publish time, delivery time, and outcome for each message under varied Wi-Fi signal strength and distance, producing the kind of quantitative reliability and latency evidence this manuscript currently lacks. A related gap is at the protocol level: all nodes connect to a public MQTT broker (broker.hivemq.com) at default quality-of-service, with no broker-level authentication. This was an acceptable simplification for a laboratory prototype but is unsuitable for residential deployment, where a private, authenticated broker with TLS and per-node credentials is required.

A second, more familiar limitation is connectivity fragility. Every remote function in the system—notifications, manual override, status updates—depends on the same home router. A power cut or Wi-Fi outage leaves caregivers uninformed even while the local actuators keep working. Adding a small battery to the coordinator node and a secondary GSM or LoRa radio would close that gap without redesigning the sensor nodes.

The hardware layout also adds installation complexity. Spreading five independent boards across a dwelling means many cable terminations and physical connectors that can loosen over time. A custom PCB consolidating two or three subsystems would cut connection points and make the system more robust for long-term residential use. That redesign would be worth pursuing once the sensor node firmware is stable.

All performance figures in this paper come from a controlled laboratory prototype with a consistent, unobstructed Wi-Fi signal. A real apartment introduces reinforced concrete walls, neighbouring networks, and users who may not follow the expected interaction patterns. Those factors could push message latency above the two-second target in some rooms, and the debounce thresholds may need tuning for different climates. Field trials with actual elderly or disabled residents are the necessary next step before any deployment recommendation can be made.

Finally, the mobile application layer needs work on its MQTT reconnection handling. When a phone drops and rejoins the Wi-Fi network, the client currently takes several

seconds to re-subscribe to its topics, during which any alert published by a sensor node is missed. This is a known challenge in residential deployments [25] and is solvable with persistent session flags and local message buffering on the coordinator node.

6. Conclusions

The results show that a five-node distributed ESP32 architecture can deliver functional, responsive home automation for elderly and disabled residents at a component cost of approximately 73 USD. Each of the four sensor subsystems behaved as designed during repeated demonstration on a scale dwelling model: the RFID module correctly distinguished authorised from unauthorised cards, the ultrasonic module opened and closed the back door on detected presence, the gas sensor triggered layered visual, audible, and remote alerts above threshold, and the rain sensor activated laundry protection on detected precipitation. The Telegram channel delivered notifications with an observed average response time under two seconds. These findings establish functional feasibility; a controlled, statistically powered reliability and latency study, outlined in Section 5.4, is the necessary next step before any claim of field-ready performance can be made.

The waterfall-plus-test-driven methodology proved well suited to a project of this scope. Running the Wokwi simulation before physical assembly caught logic and timing errors early and reduced the number of wiring iterations needed. The modular architecture—one ESP32 per function—means a node that fails can be replaced without touching the rest of the system, and a new subsystem can be added by introducing a new node and a new MQTT topic.

Looking ahead, the most important next steps are field validation in real residential environments with elderly or disabled volunteers, integration of a backup power supply and secondary communication channel, and a formal usability study of the mobile interface with non-technical users. Exploring lightweight machine-learning models for anomaly detection—distinguishing a resident leaving the house from an intrusion, for example—could also upgrade the system from reactive alerting to genuinely predictive safety.

Acknowledgements

The authors thank the supervising faculty member for technical guidance throughout the project, and the Department of Electronic and Computer Engineering of Universidad Nacional Federico Villarreal for access to laboratory equipment and facilities.

Generative AI tools were used exclusively to assist with English text review and structural editing, under direct author supervision. All technical content, experimental data, and conclusions are the original work of the authoring team.

References

[1] United Nations, Department of Economic and Social Affairs, Population Division. World Population Ageing 2023. New York: United Nations; 2024. https://doi.org/10.18356/9789213586747. https://www.un-ilibrary.org/content/books/9789213586747.

[2] World Health Organization. Ageing and health [Internet]. Geneva: WHO; 2022 [cited 2025 Apr]. Available from: https://www.who.int/news-room/fact-sheets/detail/ageing-and-health.

[3] Wang Y, Sun H, Xu S, Xia Q, Ge S, Li M, et al. Smart home technologies for enhancing independence of living and reducing care dependence in older adults: a systematic review. J Adv Nurs. 2025;81(6):2885–2912. https://doi.org/10.1111/jan.16569.

[4] Jamwal R, Jarman HK, Roseingrave E, Douglas J, Winkler D. Smart home and communication technology for people with disability: a scoping review. Disabil Rehabil Assist Technol. 2022;17(6):624–644. https://doi.org/10.1080/17483107.2020.1818138.

[5] Poyyamozhi M, Murugesan B, Rajamanickam N, Shorfuzzaman M, Aboelmagd Y. IoT—a promising solution to energy management in smart buildings: a systematic review, applications, barriers, and future scope. Buildings. 2024;14(11):3446. https://doi.org/10.3390/buildings14113446.

[6] De la Cruz Severiche Maury Z, Fernández Vilas A, Díaz Redondo RP. Low-cost HEM with Arduino and Zigbee technologies in the energy sector in Colombia. Energies. 2022;15(10):3819. https://doi.org/10.3390/en15103819.

[7] Ezugwu IM, Ukpai UO, Chima-Chukwu CU, Ugwuanyi JO, Emeghara GC. Smart homes of the future. Trans Emerg Telecommun Technol. 2025;36(1):e70041. https://doi.org/10.1002/ett.70041.

[8] Adhikary A, Halder S, Bose R, Panja S, Halder S, Pratihar J, et al. Design and implementation of an IoT-based smart home automation system in real world scenario. EAI Endorsed Trans IoT. 2024;10. https://doi.org/10.4108/eetiot.6201.

[9] 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.

[10] Froiz-Míguez I, Fernández-Caramés TM, Fraga-Lamas P, Castedo L. Design, implementation and practical evaluation of an IoT home automation system for fog computing applications based on MQTT and ZigBee-WiFi sensor nodes. Sensors. 2018;18(8):2660. https://doi.org/10.3390/s18082660.

[11] Park Y, Han J. Smart home advancements for health care and beyond: systematic review of two decades of user-centric innovation. J Med Internet Res. 2025;27:e62793. https://doi.org/10.2196/62793.

[12] Esposito M, Belli A, Palma L, Pierleoni P. Design and implementation of a framework for smart home automation based on cellular IoT, MQTT, and serverless functions. Sensors. 2023;23(9):4459. https://doi.org/10.3390/s23094459.

[13] Chang YH, Wu FC, Lin HW. Design and implementation of ESP32-based edge computing for object detection. Sensors. 2025;25(6):1656. https://doi.org/10.3390/s25061656.

[14] Hercog D. Design and implementation of ESP32-based IoT devices. Sensors. 2023;23(15):6739. https://doi.org/10.3390/s23156739.

[15] Okokpujie K, Jacinth D, James GA, Okokpujie IP, Vincent AA. An IoT-based multimodal real-time home control system for the physically challenged: design and

implementation. Inf Dyn Appl. 2023;2(2):90–100. https://doi.org/10.56578/ida020204.

[16] Easterline LM, Ferdiansyah C, Anggoro O, Isnanto RR, Somantri M. Smart air monitoring with IoT-based MQ-2, MQ-7, MQ-8, and MQ-135 sensors using NodeMCU ESP32. Procedia Comput Sci. 2024;245:815–824. https://doi.org/10.1016/j.procs.2024.10.308.

[17] Mota A, Serôdio C, Briga-Sá A, Valente A. Implementation of an Internet of Things architecture to monitor indoor air quality: a case study during sleep periods. Sensors. 2025;25(6):1683. https://doi.org/10.3390/s25061683.

[18] Jabbar WA, Kian TK, Ramli RM, Zubir SN, Zamrizaman NSM, Balfaqih M, et al. Design and fabrication of smart home with Internet of Things enabled automation system. IEEE Access. 2019;7:144059–144074. https://doi.org/10.1109/ACCESS.2019.2942846.

[19] Dinmohammadi F, Farook AM, Shafiee M. Improving energy efficiency in buildings with an IoT-based smart monitoring system. Energies. 2025;18(5):1269. https://doi.org/10.3390/en18051269.

[20] López Heredia RR, Ravelo Batista AA, Benítez Pina IF. Sistema de uso doméstico para monitorear el consumo energético combinando el uso de las plataformas Arduino y Android. LADEE. 2022;3(1):25–30. https://doi.org/10.17981/ladee.03.01.2022.2.

[21] Villegas MD, Paredes ED, Arévalo JA, Quito Carrión A, Pillajo R, Cuenca Sánchez A, et al. Smart automation for residential spaces with PLC-ESP32 architecture. Eng Proc. 2025;115(1):7. https://doi.org/10.3390/engproc2025115007.

[22] Chandrashekar G, Vaishnavi P, Neeharika D, Suchitha N, Rohith T. Smart building management system using ESP32. IRJAEM. 2025;3(5):1962–1966. https://doi.org/10.47392/IRJAEM.2025.0308.

[23] Hutajulu OY, Mendoza MD, Simamora Y. Feasibility study of smart house design with ESP32 and Telegram. In: ICIESC Conference Proceedings; 2022 Oct 11; Indonesia: EAI; 2022. https://doi.org/10.4108/eai.11-10-2022.2325332.

[24] Attar A, Redekar M, Khan N, Kumar A, Phule S. Smart home security and air quality monitoring using Telegram chatbot. IRJMETS. 2023;5(5):3397–3399. https://doi.org/10.56726/IRJMETS39197.

[25] Mohamed A, Ismail I, AlDaraawi M. IoT-driven intelligent energy management: leveraging smart monitoring applications and artificial neural networks for sustainable practices. Computers. 2025;14(7):269. https://doi.org/10.3390/computers14070269.

[26] Wijanarko Y, Alfarizal N, Pratama MR. Implementation of an RFID RC522 and IoT-based automatic door security system in an electrical engineering laboratory. Indones J Artif Intell Data Min. 2025;8(2):478–488. https://doi.org/10.24014/ijaidm.v8i2.37007. https://ejournal.uin-suska.ac.id/index.php/IJAIDM/article/view/37007.

[27] Pakpahan DMN, Purwadi H, Wajiansyah A. Arduino-based automatic door opener design using ultrasonic sensors. J Eng Electr Inform. 2026;6(1):17–24. https://doi.org/10.55606/jeei.v6i1.6663.

[28] AG Electrónica. OKY3435: Sensor detector de lluvia [Datasheet]; Pololu. MQ-2 smoke/gas sensor datasheet. https://agelectronica.lat/pdfs/textos/O/OKY3435.PDF; https://www.pololu.com/file/0j309/mq2.pdf.

[29] NXP Semiconductors. MFRC522 standard performance MIFARE and NTAG frontend [Datasheet]. Rev 3.9. Eindhoven: NXP; 2016. Available from: https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf.

[30] Prasetyo Tulodo R, Indah Fitria R, Sofyan A, Budiraharjo E. Penggunaan simulator Wokwi untuk meningkatkan literasi pemrograman mikrokontroler dalam proyek Internet of Things. EDUSAINTEK. 2024;12(1):72–81. https://doi.org/10.47668/edusaintek.v12i1.1442. https://journalstkippgrisitubondo.ac.id/index.php/EDUSAINTEK/article/view/1442.

[31] Widianto MH, Cahaya Putra VH. Utilization of Wokwi simulation application in supporting Internet of Things learning. In: 2023 International Conference on Information Management and Technology (ICIMTech); 2023 Aug; IEEE; 2023. pp. 807–812. https://doi.org/10.1109/ICIMTech59029.2023.10277981.

[32] Atanasković A, Dimitrijević T, Ilić NM, Čabarkapa M. Empowering IoT education utilizing free online Arduino simulators. In: 2024 59th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST); 2024; IEEE; 2024. pp. 1–4. https://doi.org/10.1109/ICEST62335.2024.10639688.

[33] Dzahir MASM, Chia KS. Evaluating the energy consumption of ESP32 microcontroller for real-time MQTT IoT-based monitoring system. In: 2023 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT); 2023 Nov 20; IEEE; 2023. pp. 255–261. https://doi.org/10.1109/3ICT60104.2023.10391358.

[34] Espinosa-Gavira MJ, Agüera-Pérez A, Palomares-Salas JC, Sierra-Fernández JM, Remigio-Carmona P, González-de-la-Rosa JJ. Characterization and performance evaluation of ESP32 for real-time synchronized sensor networks. Procedia Comput Sci. 2024;237:261–268. https://doi.org/10.1016/j.procs.2024.05.104.

[35] Martinez-Blanco M del R, Cesar Soriano-Romero J, Serrano-Muñoz A, Hernan Escobedo-Barajas M, del Rio de Santiago A, Alonso Guerrero Osuna H, Ortiz-Rodriguez JM. IoT based smart electrical meter for smart homes. EAI Endorsed Trans IoT. 2020;6(21):e2. https://doi.org/10.4108/eai.13-7-2018.165672.

[36] Kanna RK, Pradhan NR, Panigrahi BS, Basa SS, Mohanty S. Smart assist system module for paralysed patient using IoT application. EAI Endorsed Trans IoT. 2024;10. https://doi.org/10.4108/eetiot.5315.