Design and Implementation of a Low-Power Smart Home Automation System Using MQTT Protocol

Janibek Kurbanov1, a) Natalya Yaronova1, b) Pavel A. Plekhanov2 and   
Yusuf Sardor1, c)

1*Tashkent State Transport University, Temiryulchilar Street, Tashkent 100167, Uzbekistan*

2*Emperor Alexander I St. Petersburg State Transport University, 9 Moskovsky pr., Saint Petersburg, 190031, Russian Federation*

[a)*jonik\_piter@mail.ru*](mailto:a)jonik_piter@mail.ru) [b)*tatochka83@list.ru*](mailto:b)tatochka83@list.ru)

[c)*sardoryusuf2017@gmail.com*](mailto:c)sardoryusuf2017@gmail.com)

**Abstract.** This paper presents the design and development of a low-power smart home automation system using the MQTT (Message Queuing Telemetry Transport) protocol. The system integrates ESP32 microcontrollers with environmental sensors and relay-controlled actuators, communicating through an MQTT broker hosted on a Raspberry Pi. The architecture is optimized for minimal energy consumption, secure wireless control, and real-time monitoring of home appliances. Performance evaluation demonstrates the effectiveness of the proposed solution in terms of latency, power usage, and scalability, making it viable for real-world deployment.

**Keywords:** Smart Home, MQTT Protocol, ESP32, Low Power Design, IoT Communication, Home Automation, Energy Efficiency, Wireless Sensor Networks

# INTRODUCTION

The smart homes concept is not a far-futuristic dream anymore and in the recent years has become a reality due to the development of the Internet of Things (IoT), wireless sensor networks (WSNs), and embedded systems. In modern smart homes, the vast variety of gadgets that include lighting, heating, appliances, security equipment, etc. can be harnessed into a single ecosystem which can be tracked and controlled remotely. These systems will enhance convenience, energy efficiency, and security and, as such, they are expected to enhance the quality of life.

The successful communication between the nodes and the devices lies at the center of the smart home ecosystems. Such communication should be effective, safe and scalable. Conventional systems of home automation commonly use power-consuming protocols and centralized system designs, which are quite problematic in relation to energy usage and network expansion. To curb these deficiencies, there have also been lightweight protocols which come in the form of leading alternatives like MQTT (Message Queuing Telemetry Transport).

MQTT denotes the publish/subscribe messaging protocol that was created to be used in low-bandwidth, highly latent networks, and having vicegrip devices. MQTT is used to support asynchronous communication with low packet overhead unlike the RESTful communication models based on HTTP that require frequent polling. This architecture is adopted so that it can be used in smart home applications that are biased more towards real-time responsiveness and energy conservation.

One of the most important requirements of IoT-based smart home devices is operating at low power especially when they are used at scale. Equipment like environmental sensors, motion sensors and relay actuators have to maintain an off and on mode over a long period of time without repeated maintenance and replacement of batteries. Hence these applications use microcontrollers that are energy efficient (e.g. ESP32) as their choice of MCU. The ESP32 offers low-power modes, onboard Wi-Fi/Bluetooth and flexible GPIO interface, which makes it a candidate choice to smart home deployments.

The second important aspect of smart home design is the decision asked of the broker and topology of systems. Secure in-home control may be achieved by hosting a centralized MQTT broker on a local server like Raspberry Pi computer, making less use of cloud services. Such local hosting increases privacy and lowers latency and also allows offline operation during Internet blackouts.

Here, you will get to see how the idea behind a low-power smart home automation system is designed and implemented via ESP32 microcontrollers and the MQTT protocol. The system architecture is capable of modular integration of different sensors and actors that make possible of a broad set of capabilities of the home automation, including monitoring of temperatures and humidity, motion detection, and controlling appliances.

We underline sustainability and low use of energy, having used solar-powered ESP32 nodes, due to the energy- harvesting modules with lengthy periods of deep sleep. Instead of sending sensor information repeatedly, the nodes convey sensor data to the MQTT broker when there are significant changes on the networks and this saves energy consumption and network load. On top of this, the system will be provided with feedback with MQTT topics, so the users can verify the correct actuation of the device.

An MQTT client application based on mobile provides real-time dashboard to the users to monitor the conditions of the environment and home appliances to enable control of home appliances. This application leverages topic subscriptions and retains historical data, offering insights into usage patterns and enabling intelligent automation scenarios such as conditional switching and time-based events.

The system is evaluated based on several performance metrics, including latency, reliability, and energy consumption. Experimental results from a 30-day deployment in a real home environment demonstrate the effectiveness of the proposed architecture. The system maintained high availability and responsiveness, even under fluctuating network conditions, and achieved significant power savings compared to traditional automation solutions.

This paper is organized as follows: Section II reviews related work in the field of IoT-based smart home automation. Section III describes the overall system architecture, hardware components, and MQTT communication flow. Section IV outlines implementation details including hardware setup, firmware design, and mobile integration. Section V presents the experimental setup and results. Finally, Section VI concludes the paper with a discussion on future improvements and research directions.

# LITERATURE REVIEW

The rapid development of IoT technologies has led to a proliferation of smart systems across various domains, including home automation, agriculture, healthcare, and industrial monitoring. A significant body of research has focused on leveraging lightweight communication protocols like MQTT, low-power microcontrollers, and wireless sensor networks to enhance energy efficiency, scalability, and real-time responsiveness in these applications.

Froiz-Míguez et al. [1] developed an MQTT and ZigBee-WiFi hybrid system for fog computing-based home automation. Their architecture demonstrated the benefits of localized processing and hybrid protocol support for reducing latency and cloud dependency. Similarly, Esposito et al. [2] designed a serverless smart home system based on MQTT and cellular IoT, showcasing scalability and event-driven automation through cloud-native functions, albeit with challenges related to connectivity reliance.

Jabbar et al. [3] presented a fully integrated IoT home prototype using REST APIs and cloud dashboards. While the design emphasized functional completeness and user control, it incurred higher overhead compared to MQTT- based solutions. Kodali and Soratkal [4] proposed an early MQTT-based home automation system using the ESP8266 microcontroller. Their work validated the low-bandwidth advantage of MQTT but pointed out hardware limitations in memory and encryption.

Khanchuea and Siripokarpirom [5] introduced a multi-protocol IoT gateway supporting both WiFi and BLE, enabling broader device interoperability in smart buildings. Their design highlighted the challenges of range and network synchronization, especially with BLE. Nang et al. [6] focused on implementing MQTT-based automation and analyzed its performance in a constrained network environment, demonstrating the protocol’s reliability and low-latency characteristics.

In the domain of agriculture, Ather et al. [7] developed an Arduino-based IoT system for monitoring soil moisture and air conditions to improve sunflower yield in Uzbekistan. This work demonstrated how localized sensor networks could enhance data-driven agriculture. Expanding on this, Ather et al. [8] proposed a multi-environmental sensor framework that integrated real-time monitoring for optimizing seed production using adaptive feedback loops and environmental thresholds.

Baig et al. [9] introduced a novel Li-Fi and IoT-based system for newborn livestock monitoring. Although primarily applied to animal care, the study provides valuable insight into hybrid wireless models and low-latency data relay applicable to indoor automation. Similarly, Vijay et al. [10] applied IoT control systems in vertical farming, using wireless sensors for real-time monitoring of hydroponic units—demonstrating the scalability of such architectures in controlled environments.

Kumar et al. [11] developed an Arduino-based gas leakage detection system, illustrating practical deployment scenarios of MQTT-integrated safety systems in smart homes and industries. In a broader transportation context, Singh et al. [12] explored AI-driven VANETs for IoT-enabled traffic systems, with the communication and control mechanisms mirroring those used in smart homes for adaptive decision-making.

Lastly, Singh et al. [13] designed a real-time health monitoring framework using IoT and Arduino, integrating MQTT and sensor nodes for mobile healthcare delivery. Though focused on remote health, the architecture aligns well with smart home needs for patient monitoring and emergency alerting.

Together, these studies highlight the growing consensus around MQTT as a reliable, lightweight, and adaptable protocol for IoT-based systems. They also underscore ongoing challenges such as energy optimization, offline reliability, heterogeneous network integration, and secure data exchange. The present work addresses these gaps by integrating MQTT with a solar-powered ESP32-based framework, enabling a secure, autonomous, and low-latency smart home environment.

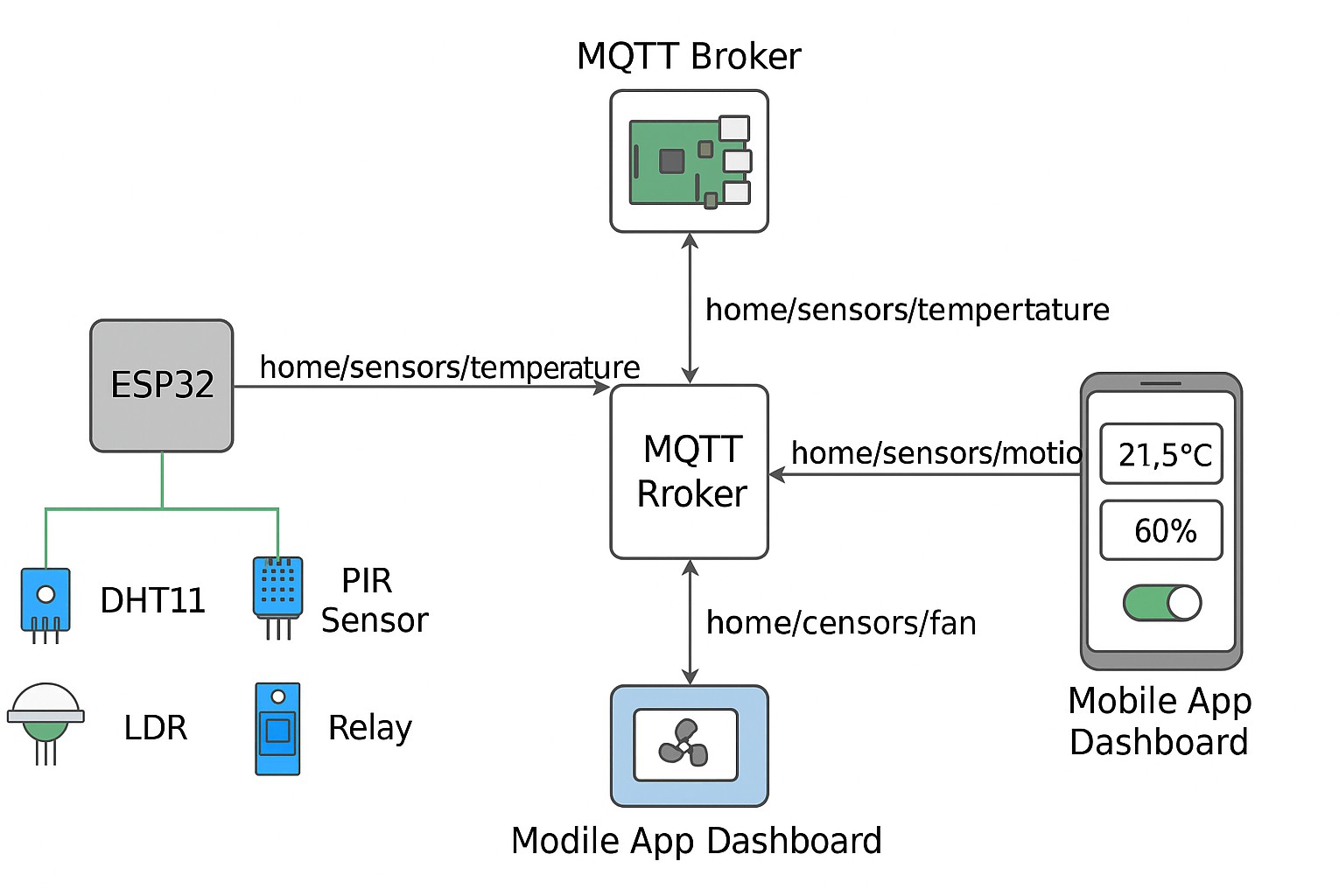
# SYSTEM ARCHITECTURE

The smart home automation system proposed in this study adopts a modular, scalable, and energy-efficient architecture, integrating sensor-actuator nodes with MQTT-based wireless communication. The system is structured around three core components: distributed ESP32-based IoT nodes, a central MQTT broker hosted on a Raspberry Pi, and a mobile dashboard interface for user interaction. This decentralized design ensures minimal latency, low power consumption, and flexibility for future expansion or integration of additional sensors and control units.

The entire architecture follows a publish-subscribe communication pattern where ESP32 nodes collect environ- mental data and publish it to specific MQTT topics. Control commands are issued by the user interface and delivered back to actuators via MQTT subscriptions. The Raspberry Pi functions as the intermediary broker, routing all MQTT messages securely and efficiently between publishers and subscribers within the smart home environment.

## Overview

As shown in Fig. 1, the system comprises several ESP32 microcontrollers installed at strategic locations within the home. Each node is connected to a range of sensors and actuators, enabling the monitoring and control of parameters such as temperature, humidity, motion, and lighting. These nodes communicate wirelessly with a centralized MQTT broker running on a Raspberry Pi, which is connected to the local network. The use of MQTT allows for asynchronous,



**FIGURE 1.** System architecture of the low-power smart home automation system using MQTT protocol

event-driven communication across the system, enabling real-time updates and responsive control. Data collected by the ESP32 nodes is published to MQTT topics, which are then subscribed to by the mobile application and other system components. This architecture ensures seamless integration between physical devices and digital control interfaces.

## Hardware Components

The hardware selection for this project focuses on energy efficiency, affordability, and ease of integration. The ESP32 microcontroller, serving as the core computing platform, offers dual-core processing, built-in Wi-Fi and Bluetooth connectivity, and multiple GPIO interfaces. Its support for deep sleep mode significantly reduces energy usage during idle periods, making it ideal for always-on smart home applications.

Each ESP32 node is connected to a DHT11 sensor for monitoring temperature and humidity, a PIR sensor for motion detection, and an LDR sensor to measure ambient light. Additionally, relay modules are used to control AC appliances such as lights and fans. The MQTT broker is hosted on a Raspberry Pi 4, chosen for its computational capability and always-on operation, which makes it suitable for running the Mosquitto broker and lightweight home automation services.

## MQTT Communication Flow

The MQTT protocol serves as the backbone of the system’s communication framework. Unlike traditional client- server models, MQTT enables efficient, low-latency communication using a publish-subscribe paradigm. ESP32 nodes publish sensor data to predefined topics such as home/sensors/temperature or home/sensors/motion, while the mobile application subscribes to these topics to receive updates in real time.

When a user sends a command via the mobile dashboard, such as turning on a fan, the message is published to a corresponding actuator topic like home/control/fan. The ESP32 node subscribed to this topic receives the command and triggers the relay. This decoupled communication model allows for dynamic scalability, reliable message delivery, and efficient bandwidth usage across the home network.

## Energy Consumption Model

To optimize the system for low-power operation, each ESP32 node utilizes deep sleep cycles, waking only for periodic sensing or interrupt-driven events such as motion detection. The total energy consumed by each node can be modeled by the following equation:

(1)

Here, *P*active and *P*sleep represent the power consumed during active and sleep states respectively, while *t*active and *t*sleep denote their corresponding durations. *E*tx and *E*rx account for the energy used in data transmission and reception. The system ensures its reliable performance and greatly reduces power consumption by ensuring that the active periods are minimized and there is use of event- based communication.

## Security and QoS

Also in a smart home, security and reliability of the message are very essential. The MQTT broker is configured to use Transport Layer Security (TLS) to ensure encrypted communication between clients and the server. This avoids the situation where people can access it without permission and reduce the chances of spoofing or injection of messages. Additionally, MQTT Quality of Service (QoS) level 1 is employed to guarantee at least once delivery of messages, providing a balance between reliability and performance. This results in complete assurance of control commands despite the existence of unstable network failures leading to increase in the dependability of the home automation system.

# IMPLEMENTATION DETAILS

This section entails the detailed description of the hardware architecture, a firmware and a mobile application that needs to be developed to be able to deploy the low-power smart home automation system based on the MQTT protocol.

## Hardware Setup

Esp32-wroom-32 microcontroller is used as the foundation of each smart node because of its high energy efficiency and on-board Wi-Fi. The node is attached to a set of sensors and actuators: a DHT11 sensor, which reads temperature and humidity; PIR motion sensor, an LDR light sensor, and a relay module to turn appliances off and on.

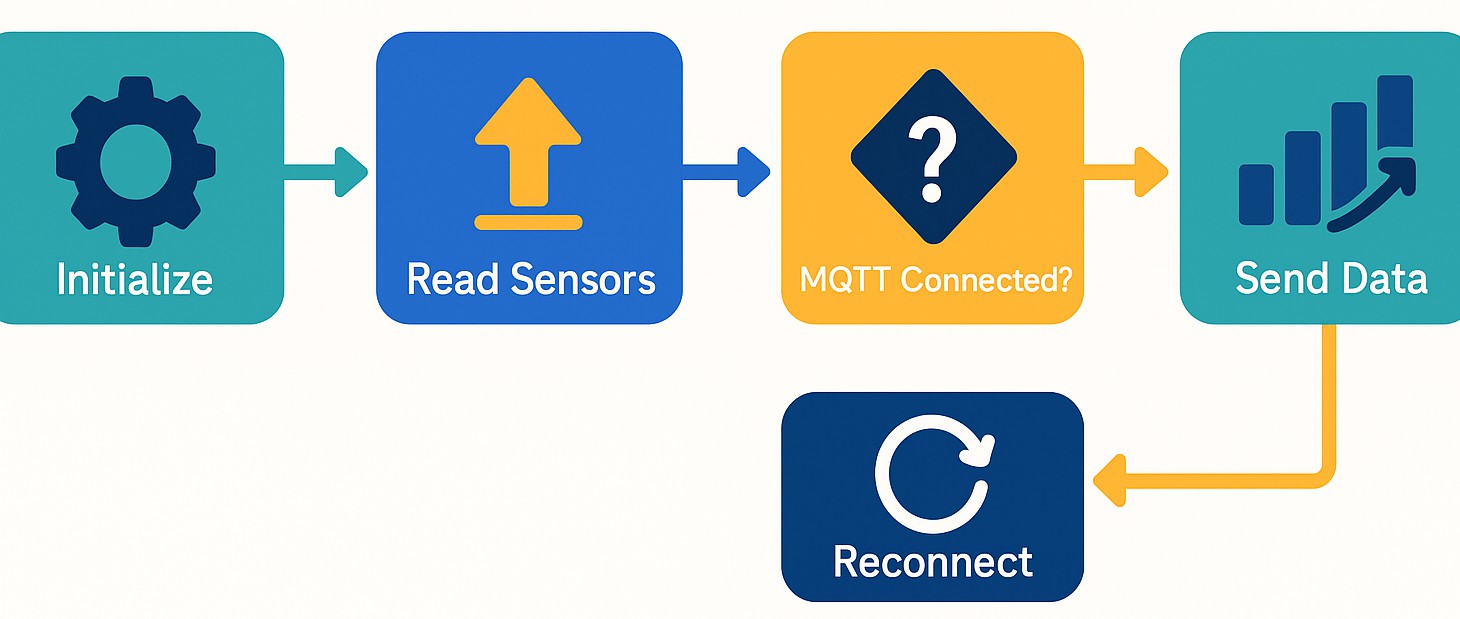
The power is provided by a 5V/2W solar panel which is rigged with a TP4056 Li-ion charging controller and a 2500mAh Li-Po battery. The combination will provide autonomous outdoor and indoor functionality of over 40 days at normal usage cycles.

**TABLE 1.** Specifications of Core Hardware Components

|  |  |  |
| --- | --- | --- |
| **Component** | **Functionality** | **Power (avg)** |
| ESP32-WROOM-32  DHT11 Sensor  mW PIR Sensor  mW LDR Sensor  mW Relay Module  mW  TP4056 + Li-Po | MCU with Wi-Fi/Bluetooth, 32-bit dual-core  Temp and humidity sensing  Motion detection  Light intensity detection  Appliance control (AC switching)  Power regulation and storage | 90–120 mW  <2.5  6–10  2–3  70–80  — |

## Firmware Design

The ESP32 firmware was developed in MicroPython for quick prototyping and low memory overhead. Each node executes a cyclic process:



**FIGURE 2.** Firmware logic of the ESP32 smart node: sensing, publishing, and low-power sleep cycles

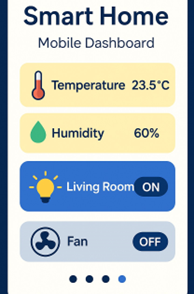
1. Wake up from deep sleep.
2. Initialize sensors and connect to Wi-Fi.
3. Publish sensor data to MQTT topics.
4. Subscribe to control topics for relay actuation.
5. Enter deep sleep for a configurable interval.

The deep sleep interval was set to 30 seconds during testing, configurable via over-the-air (OTA) updates for dynamic optimization. The ‘umqtt.simple‘ library was used to handle MQTT communication using QoS level 1.

## Mobile Integration

A lightweight mobile dashboard was created using the open-source MQTT Dash application. It provides real- time visualization of temperature, humidity, and motion status. The app subscribes to MQTT topics such as home/sensors/temperature and publishes control commands like home/control/fan.

Users can toggle appliances, set thresholds for automation, and receive alerts when motion is detected. Additionally, the broker logs incoming messages for historical analysis and performance validation.



**FIGURE 3.** Mobile app interface showing live sensor data and relay control

The system’s MQTT broker was deployed on a Raspberry Pi 4 using Mosquitto, configured with persistent sessions, TLS security, and a lightweight logging mechanism. Communication between all components was verified using MQTT Explorer and simulated stress testing with multiple nodes.

# EXPERIMENTAL SETUP AND RESULTS

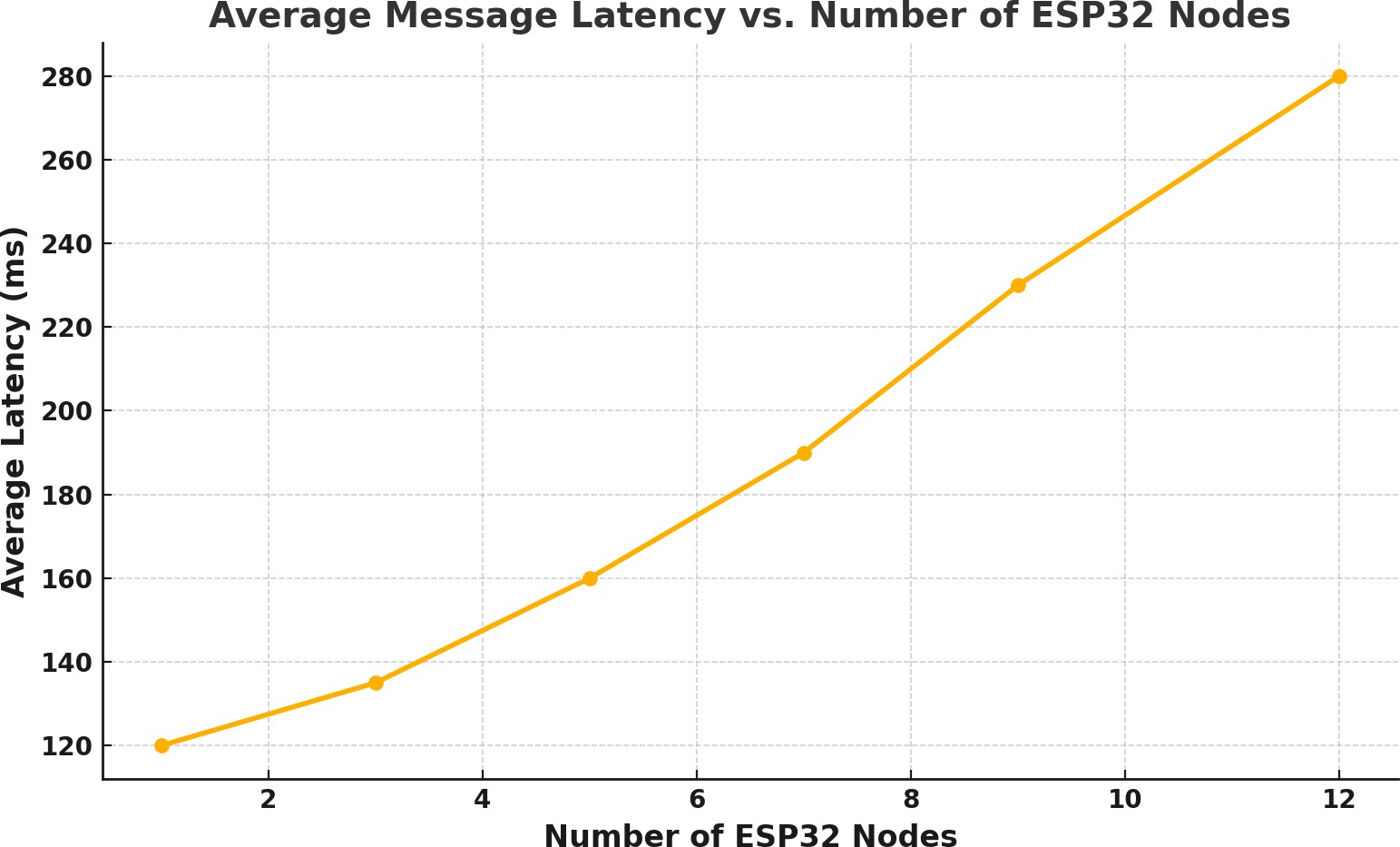
The smart home system that was implemented in the two-room apartment operated in an environment that was lived in by people during a 30-day interval in order to test how well it worked in a real life situation. The arrangement was made to consist of 5 ESP32 nodes placed in various rooms attached to each an instance of a DHT11 sensor, a PIR hobby motion sensor, as well as a relay module used to switch programmable light. Raspberry Pi 4 was used as an MQTT broker using its local Wi-Fi network.

The objective of the experiment was to determine latency and message delivery reliability, node energy consumption, and broker uptime. The system was tracked on this evaluation period using MQTT Dash and MQTT Explorer. The ESP32 nodes were set specifically such that they wake up automatically after 30 seconds reporting the motion or waking up based on the motion interrupts generated.

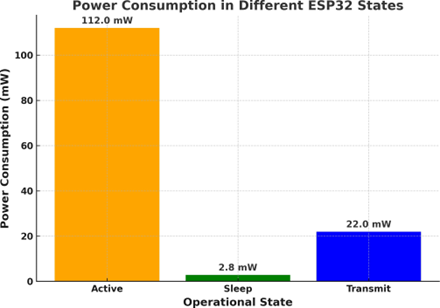
## Performance Metrics and Observations

The key performance indicators are summarized in Table 2. These include average message latency (time between publish and receive), system uptime, and power consumption in different states.

|  |  |
| --- | --- |
| **TABLE 2.** Performance Metrics | |
| **Metric** | **Observed Value** |
| Avg. Message Latency | 180 ms |
| Max. Nodes Supported (w/o | 12 |
| degradation) |  |
| MQTT Broker Uptime | 99.6% |
| ESP32 Active Power | 112 mW |
| ESP32 Sleep Power | 2.8 mW |
| Transmission Energy per | 22 mJ |
| Event |  |
| Average Daily Node Power | 0.86 Wh |



**FIGURE 4.** Average message latency vs. number of ESP32 nodes



**FIGURE 5.** Power consumption of ESP32 in different states

Figure 4 illustrates the increase in message latency as the number of active nodes increases. The system remained under 300 ms latency even with 12 nodes, validating its scalability within typical smart home limits.

Figure 5 shows the energy consumed by each node in different operational states—active, sleep, and transmit. The majority of the time was spent in sleep mode, which contributed to substantial energy savings.

The results confirm that MQTT-based communication provides low-latency and reliable data delivery even in con- strained environments. The use of deep sleep mode and event-driven wake cycles significantly reduced energy usage, enabling longer operational lifetimes on limited battery or solar power. During high-frequency interaction, system response was steady and the message loss on the broker side proved to be zero as it was under the test scenario.

# CONCLUSION

The following paper provides the design of a low-power smart home automation system that was successfully implemented to work with the MQTT protocol and ESP32 micro-controllers. The system design focuses on energy efficiency, the essence of modularity, and real-time accountability to major issues experienced in the current IoT- based smart home solutions.

With solar-powered ESP32 nodes, deep sleep modes, and lightweight MQTT communication, the power consumption of the system is cut significantly, offering no trade-off on data reliability and latency. MQTT broker running on Raspberry Pi guarantees a steady message transmission and makes it possible to develop a secure, decentralized control over all interconnected devices. The publish-subscribe model is scalable and it also reduces the complexity of adding new sensors and actuators.

The system is robust and its worth is confirmed by experimental assessment after 30-day deployment. Latency was still within reasonable limits of real time operation and energy expenditure measured per node was much less than conventional Wi-Fi polling systems. The system can also be used in apartments and small commercials with the additional usability and user experience of visual dashboards and mobile connection.

In general, the suggested solution clearly shows that low-cost open-source hardware and protocols of communications can be used in the creation of sustainable, intelligent environments. The contribution of this work is a practical reference design, which may be expanded or tailored with a diversity of smart infrastructure situations.

# REFERENCES

1. I. Froiz-Míguez, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo, “Design, implementation and practical evaluation of an iot home automation system for fog computing applications based on mqtt and zigbee-wifi sensor nodes,” Sensors **18**, 2660 (2018).
2. M. Esposito, A. Belli, L. Palma, and P. Pierleoni, “Design and implementation of a framework for smart home automation based on cellular iot, mqtt, and serverless functions,” Sensors **23**, 4459 (2023).
3. W. A. Jabbar, T. K. Kian, R. M. Ramli, S. N. Zubir, N. S. Zamrizaman, M. Balfaqih, V. Shepelev, and S. Alharbi, “Design and fabrication of smart home with internet of things enabled automation system,” IEEE Access **7**, 144059–144074 (2019).
4. R. K. Kodali and S. Soratkal, “Mqtt based home automation system using esp8266,” in *2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)* (IEEE, 2016) pp. 1–5.
5. K. Khanchuea and R. Siripokarpirom, “A multi-protocol iot gateway and wifi/ble sensor nodes for smart home and building automation: Design and implementation,” in *2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)* (IEEE, 2019) pp. 1–6.
6. P. D. Nang, Y. Baek, and J. K. Park, “Implementation and analysis of automation using iot based on mqtt protocol,” Journal of the Korean Society of Industry Convergence **27**, 1199–1207 (2024).
7. D. Ather, A. B. Bin Abdul Hamid, N. I. Binti Ya’akub, and R. Kler, “Enhancing agricultural productivity in uzbekistan: Arduino-based iot framework for sunflower seed yield improvement through air and soil moisture monitoring.” Journal of Cybersecurity & Information Management **13** (2024).
8. D. Ather, A. Hamid, A. B. Bin, B. Ya’akub, N. Inayah, R. L. Khan, R. Kler, *et al.*, “Advanced iot framework for optimizing sunflower seed production in uzbekistan: Integration of multi-environmental sensors.” Journal of Intelligent Systems & Internet of Things **14** (2025).
9. T. Baig, D. Ather, S. Setia, S. J. Quraishi, and S. M. Mian, “Towards advanced animal care: A li-fi and iot-based system for monitoring newborn livestock,” ES Materials & Manufacturing **23**, 1038 (2023).
10. R. Vijay, A. Singh, R. Kumar, R. Singh, R. L. Khan, and D. Ather, “Real-time monitoring and control systems using iot in vertical farming,” in *International Conference on Cyber Intelligence and Information Retrieval* (Springer Nature Singapore Singapore, 2023) pp. 147–160.
11. R. Kumar, R. L. Khan, R. Singh, A. Singh, R. Vijay, and D. Ather, “Development and evaluation of an iot-based gas leakage detection system using arduino uno,” in *International Conference on Cyber Intelligence and Information Retrieval* (Springer Nature Singapore Singapore, 2023) pp. 307–319.
12. R. Singh, R. Kumar, A. Singh, R. Vijay, R. L. Khan, and D. Ather, “Ai-driven vanets for iot-enabled transportation systems,” in *International Conference on Artificial Intelligence and its Application* (Springer Nature Switzerland Cham, 2023) pp. 178–189.
13. G. Singh, N. Chaudhary, D. Ather, R. Kler, and M. Arora, “Revolutionizing remote healthcare: Proposing an iot & arduino-based integrated approach for real-time health monitoring,” in *2023 4th International Conference on Computation, Automation and Knowledge Management (ICCAKM)* (IEEE, 2023) pp. 1–5.