Smart Home Automation Using IoT

Samarth Ramesh Chormale[1], Rehan Vasim Sayyad[2], Prof. Dr. Khatal S S[3], Prof. Sonawane P S[4]
[1,2,3,4]*Sharadchandra Pawar College of Engineering, Otur, Maharashtra, India*

***Abstract*—Smart homes connect household devices with sensors, embedded controllers, communication networks, and user applications. This paper examines a context-aware IoT architecture in which a Wi-Fi-enabled edge controller coordinates sensor-based rules and user commands while retaining essential control functions locally. The proposed design combines motion, temperature, and ambient-light sensing with an ESP32-class controller, a communication layer, and electrically appropriate output interfaces. Unlike a cloud-dependent design, basic routines are intended to continue locally while remote access remains optional. A literature-informed review of device capabilities, messaging protocols, interoperability, and IoT security is used to derive the design considerations. The paper describes the workflow, component roles, implementation methodology, and an evaluation plan covering response time, command reliability, offline behaviour, and security. No experimental performance results are claimed; the contribution is a structured design proposal suitable for an academic prototype. The discussion emphasizes that useful automation depends on safe electrical interfaces, reliable feedback, privacy, and maintainability.**

***Index Terms*—Internet of Things, smart home, home automation, edge computing, ESP32, MQTT, sensors, interoperability, cybersecurity.**

I. INTRODUCTION

A conventional home is operated through separate switches, remotes, and standalone appliances. Although these controls are simple, they do not coordinate with one another or respond automatically to changing room conditions. Internet of Things (IoT) technology connects physical devices so they can exchange status information and respond to authorized commands. In a smart home, a sensor can detect occupancy, a controller can interpret the event, and a switching device can operate a light or fan.

The engineering challenge is not merely switching an appliance through a phone. A useful system must decide where automation rules execute, how devices communicate, how the user knows whether a command succeeded, and what happens when Wi-Fi or cloud services fail. A system dependent entirely on a remote server may become unresponsive during an internet outage. The proposed approach therefore places basic rules on a local controller and uses network services mainly for supervision, configuration, and remote access.

This paper focuses on a conceptual model for lighting, occupancy indication, and temperature-aware appliance control. It is an academic design proposal, not a certified electrical product. Mains-powered loads require correctly rated, enclosed, isolated switching equipment and installation by a qualified person.

II. RELATED WORK

IoT architectures are commonly organized into sensing, processing, communication, and application functions. The ESP32 family integrates Wi-Fi and Bluetooth with programmable processing and peripheral interfaces, making it a practical controller platform for connected-device prototypes [1]. Lightweight publish/subscribe messaging is useful when devices exchange compact status updates and commands. MQTT, standardized by OASIS, is designed for constrained machine-to-machine and IoT environments [2].

Interoperability is another important concern. Smart-home products have historically used different applications and communication ecosystems. The Connectivity Standards Alliance describes Matter as an IP-based standard intended to improve communication between compatible smart-home devices across ecosystems [3]. Its architecture separates the application layer from network technologies such as Wi-Fi, Thread, and Ethernet. This direction indicates that future home automation should avoid unnecessary dependence on a single manufacturer's interface.

Security guidance is equally relevant because connected devices may expose occupancy patterns, device states, and access controls. NISTIR 8259A identifies core device cybersecurity capabilities including device identification, secure configuration, data protection, logical access control, software update, and cybersecurity-state awareness [4]. These principles inform the requirements discussed in this paper.

The present work differs from a simple remote-switch demonstration by emphasizing local-first automation, command acknowledgement, offline behaviour, and a measurable evaluation plan. It synthesizes established device and protocol capabilities into a student-oriented architecture; it does not claim a new protocol or a validated commercial implementation.

III. PROBLEM STATEMENT AND DESIGN OBJECTIVES

A basic remote-control system may allow a user to send an ON/OFF command, but it may not verify the actual device state, preserve automation during internet failure, or prevent conflicting sensor and manual commands. The design problem is to create a modular home automation architecture that supports automatic and manual control while remaining understandable, secure, and recoverable.

Design objectives:

- Provide local sensor-based operation for selected household functions.
- Allow authenticated remote monitoring when network access is available.
- Report device state instead of treating message transmission as proof ofoperation.
- Define safe behaviour during sensor, network, or controller faults.
- Allow modular expansion through compatible sensors and output devices.

IV. PROPOSED SYSTEM

The proposed system contains five functional layers: sensing, edge processing, communication, actuation, and user interaction. A PIR sensor reports motion, a temperature sensor measures room conditions, and an ambient-light sensor estimates whether a room is already bright. The ESP32 reads these inputs and applies configured rules. A relay interface or certified smart switch operates the selected load. A mobile or web dashboard provides manual commands and displays reported status.

A. Sensing layer

A PIR sensor indicates movement within its detection zone; it does not identify a person. A temperature sensor supplies a value for threshold-based rules, while an LDR or digital light sensor provides an approximate brightness measurement. Sensor placement, calibration, range, and false triggers affect automation quality.

B. Edge-processing layer

The controller validates readings, applies thresholds and timing rules, and maintains a state for each device. A lighting rule may switch a lamp on only when motion is detected and ambient light is below a configured threshold.

A delay can prevent rapid ON/OFF oscillation. Manual commands should have a defined priority and timeout so that a user is not surprised by an immediate sensor override.

C. Communication layer

Wi-Fi connects the controller to a home router. A mobile application may communicate through a local API or an IoT service. MQTT can carry messages through separate topics for commands, reported states, and sensor telemetry.

Commands should include device identity and, where practical, a unique identifier so the application can match a response to a request. Remote access should be authenticated and should not expose an unauthenticated controller directly to the public internet.

D. Actuation and feedback layer

The controller output must not directly drive a household appliance. It sends a low-voltage signal to an appropriately rated relay module or certified smart switch with suitable isolation and enclosure. The system should distinguish a requested state from a confirmed state.

Where feedback is available, it can be returned to the controller and shown in the application. A safe manual override should remain available.

V. SYSTEM ARCHITECTURE

LOCAL-FIRST SMART HOME AUTOMATION ARCHITECTURE

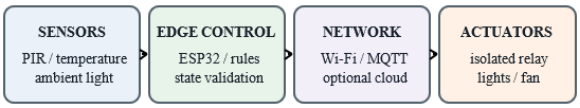

Fig. 1. Local-first IoT architecture proposed for the seminar study.

Reported device state and sensor status return to the controller and user interface.

Sensor events and user commands enter the controller. The controller evaluates both sources and generates an output action. Status feedback is returned to the controller and user interface. The cloud is optional for basic local routines; remote access may require an internet-connected gateway or service. This arrangement separates essential in-home control from functions that depend on external connectivity.

VI. COMPONENT SELECTION

Table I. Functional role of the main prototype components.

Component	Role in the design
ESP32 controller	Reads sensors, runs local rules, and provides wireless connectivity [1].
PIR sensor	Detects motion events in a defined area.
Temperature sensor	Measures room temperature for monitoring or threshold rules.
Light sensor	Estimates ambient brightness for lighting automation.
Relay / smart switch	Switches an appropriately rated appliance circuit with required isolation.
Router / MQTT service	Carries authorized commands, telemetry, and state updates.
Component	Role in the design
Mobile interface	Displays readings and allows manual control and configuration.

VII. METHODOLOGY

The proposed study follows a design-oriented methodology. First, the intended use cases are limited to a small number of household functions so each behaviour can be tested independently. Hardware is selected according to sensing range, voltage level, controller interfaces, communication support, and availability. Control logic is specified before implementation to avoid ambiguous interactions between automation and manual commands.

A. Requirement analysis

Functional requirements include reading sensor values, switching a selected load, accepting a user command, reporting device status, and recovering after restart. Non-functional requirements include reasonable response time, understandable feedback, secure access, maintainability, and safe failure behaviour. A manual switch or override is retained for essential loads.

B. Firmware and application development

Firmware is divided into input acquisition, rule evaluation, communication handling, output control, and status reporting. Sensor readings are checked against valid ranges. A state machine can represent each appliance as OFF, ON, or UNKNOWN when feedback is unavailable. The application sends a command and waits for acknowledgement or a state update. This avoids displaying success merely because a message was transmitted.

C. Rule design example

For automatic lighting, the controller may switch a light on when motion is detected and measured brightness is below a configured threshold. If no motion is detected for a configured interval, it may request OFF unless manual hold is enabled. Threshold and delay must be adjustable because room size, sensor placement, and user preference vary. A maximum-on timer can reduce the effect of a sensor stuck in the active state.

D. Communication and recovery

The local controller should continue essential automation if internet access is lost. Remote commands can be rejected while offline, but stale commands should not unexpectedly execute after reconnection. The controller should reconnect with bounded retry intervals, restore configuration safely, and publish current state after recovery. If MQTT is used, topic permissions should restrict each client to the commands and data it needs [2].

VIII. EVALUATION PLAN AND DISCUSSION

This paper presents a proposed architecture rather than a completed hardware experiment; therefore, no measured latency, energy saving, or reliability figures are reported. A future prototype should be evaluated using repeatable test cases and recorded observations.

Suggested measures include:

- Command response time: interval between issuing a command and receiving a confirmed state update.
- Command success rate: confirmed state changes divided by the total number of repeated commands.
- Sensor response: compare sensor-triggered actions with observed events, including false triggers.
- Offline continuity: disconnect internet access and verify which local rules continue.
- Recovery behaviour: restart the controller and check configuration, safe output state, and status reporting.
- Energy impact: compare appliance operating time under a defined baseline and automation schedule.

The local-first design is expected to reduce dependence on cloud availability for basic routines, but this expectation must be verified. Sensor quality and rule tuning influence whether automation is helpful. A PIR sensor may fail to detect a stationary occupant, while a poorly placed light sensor may not represent the brightness of the whole room. Manual override and user adjustment are therefore practical requirements.

Interoperability also requires attention. A vendor-specific API may work well within one ecosystem but be difficult to integrate with other brands. Standards such as Matter aim to reduce fragmentation for compatible products [3]. An ESP32 and MQTT prototype demonstrates an IoT control architecture; it is not automatically a Matter-certified device.

IX. APPLICATIONS AND POTENTIAL BENEFITS

The architecture can support smart lighting, occupancy-based control, temperature monitoring, scheduled appliance operation, door or window status alerts, and energy-use dashboards. A unified interface may reduce the number of physical actions required to operate selected devices. Automation can help avoid unnecessary operating hours, although actual savings depend on user behaviour, appliance efficiency, control settings, and local conditions.

A further benefit is modular expansion. Additional sensors or smart switches can be introduced without redesigning the entire system if device identity, message format, and access policy are defined consistently. This is useful for educational prototypes because individual functions can be demonstrated and tested in stages.

X. SECURITY, PRIVACY, AND SAFETY

Smart-home devices can reveal occupancy routines and may control physical equipment. Each device should have a unique identity; default credentials should be changed; access to communication interfaces should be restricted; and software updates should be supported. NIST's IoT baseline highlights device identification, configuration, data protection, interface access, software update, and security-state awareness as important capabilities [4].

Only necessary data should be collected. Sensor histories should have a clear purpose and retention period, and remote access should require authenticated users. Network segmentation can reduce exposure of IoT devices to personal computers and other sensitive systems. Security logs should avoid storing passwords or unnecessary personal information.

Electrical safety is separate from cybersecurity. Microcontroller GPIO pins are low-voltage signals and must never connect directly to mains wiring. Household loads require appropriately rated switching devices, isolation, circuit protection, suitable enclosures, and compliance with local electrical rules. Any mains installation must be handled by a qualified electrician. The prototype should initially be demonstrated using low-voltage loads.

XI. LIMITATIONS AND FUTURE SCOPE

The proposed model is limited by sensor coverage, network quality, device compatibility, power availability, and the accuracy of configured rules. A low-cost prototype cannot represent every residential installation. It does not establish quantified energy savings or long-term reliability without field testing. Future work may investigate Matter-compatible devices, Thread-based low-power sensors, voice interfaces, predictive energy scheduling, anomaly

detection, and local dashboards that preserve privacy. Future evaluation should include long-duration operation and user feedback.

XII. CONCLUSION

This paper presented a local-first IoT architecture for smart home automation using sensors, an ESP32-class controller, wireless communication, and suitable switching interfaces. Its main design emphasis is coordination of automatic rules with manual commands, explicit state feedback, and useful operation during internet outages. The literature review shows why controller capability, lightweight messaging, interoperability, and device cybersecurity must be considered together. The proposed workflow provides a basis for an academic prototype, while practical performance remains to be measured. A dependable smart home is not defined only by remote control; it must also be safe, secure, understandable, and recoverable when components fail.

ACKNOWLEDGMENT

The author would like to thank the faculty members of the Department of Computer Engineering for their academic guidance. This paper is prepared as a technical seminar study using the references listed below.

REFERENCES

[1] Espressif Systems, “ESP32 Series Datasheet,” Espressif Systems. [Online]. Available: https://documentation.espressif.com/esp32_datasheet_en.html.

[2] A. Banks, E. Briggs, K. Borgendale, and R. Gupta, Eds., “MQTT Version 5.0,” OASIS Standard, 2019. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.

[3] Connectivity Standards Alliance, “Matter Specification, Release 1.3,” Apr. 17, 2024. [Online]. Available: https://csa-iot.org/wp-content/uploads/2024/05/matter-1-3-core-specification.pdf.

[4] M. Fagan, K. N. Megas, K. Scarfone, and M. Smith, “IoT Device Cybersecurity Capability Core Baseline,” NISTIR 8259A, National Institute of Standards and Technology, May 2020, doi: 10.6028/NIST.IR.8259A.

[5] Connectivity Standards Alliance, “Matter: The Foundation for Connected Things,” Matter Overview, 2024. [Online]. Available: https://csa-iot.org/all-solutions/matter/.