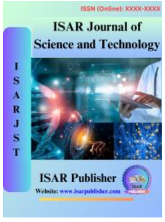
OPEN ACCESS



ISAR Journal of Science and Technology

Volume 3, Issue 11, 2025, Page: 52-64

Abbriviate Title- ISAR J Sci Tech

ISSN *(Online)*- 2584-2056

https://isarpublisher.com/journal/isarjst

AUTOMATION OF AN IMPROVED SECURED SMART HOME LIGHTING SYSTEM

Ezekiel Rebecca Ambo[1], Agu Edward[2], Siman Emmanuel[3*], Kevin Godfrey[4]

[1,2,3,4]Kwararafa University Wukari, Nigeria

[2,3]Federal University Wukari, Nigeria

[4]Taraba State Polytechnic Suntai. Nigeria.

***Corresponding Author**

Siman Emmanuel

Federal University Wukari, Nigeria.

Article History

Received: 12.08.2025
Accepted: 04.10.2025
Published: 20.11.2025

Abstract: This research presents the design and implementation of an advanced IoT-based smart home lighting system, integrating enhanced security features and a user-friendly control interface. The system leverages an ESP32 microcontroller, a relay module, an LCD, and a keypad, all interconnected with cloud platforms such as Supabase for secure authentication and ThingSpeak for command relay. The system implements Role-Based Access Control (RBAC) and lightweight encryption protocols to safeguard against unauthorized access. Through testing, the system achieved an average response time of 1.2 seconds, with 97% command accuracy and 99% uptime, demonstrating its reliability. User feedback highlighted intuitive navigation, secure role management, and efficient device control, although a few delays were linked to ThingSpeak's 15-second rate limit. The study concludes that the proposed system offers a scalable, secure, and efficient solution for smart home lighting, with recommendations for latency reduction, expanded multi-device support, and the adoption of more advanced encryption protocols in future iterations.

Keywords: Internet of Things (IoT); Smart home lighting; ESP32 microcontroller; Role-Based Access Control (RBAC); Cloud-based security.

Cite this article:

Ambo, E. R., Edward, A., Emmanuel, S., Godfrey, K., (2025). AUTOMATION OF AN IMPROVED SECURED SMART HOME LIGHTING SYSTEM. *ISAR Journal of Science and Technology*, *3*(11), 52-64.

1. Introduction

The 21st century has witnessed remarkable technological advancements that have reshaped human interaction with residential environments, with the Internet of Things (IoT) playing a central role (Rajarajeswari et al., 2021). IoT enables seamless connectivity among devices and systems, facilitating applications in smart homes, healthcare, transportation, and industrial automation, delivering benefits such as improved efficiency, convenience, and quality of life (Ahmed et al., 2018; Awad et al., 2024; Khan et al., 2020). In smart homes, lighting systems have emerged as a critical component, providing automated illumination based on real-time inputs or predefined conditions, improving both usability and energy efficiency (Rajarajeswari et al., 2021; Patel et al., 2022). Smart home lighting has evolved from manual switches to sophisticated IoT-enabled solutions that integrate sensors, microcontrollers, and mobile applications for remote control and automated responses to environmental changes (Mahmud et al., 2019; Zhang et al., 2021; Li et al., 2020). However, conventional access mechanisms, such as passwords or PINs, are vulnerable to hacking, phishing, and unauthorized access, highlighting the need for enhanced security measures (Obaidat et al., 2019; Majeed et al., 2020; Kumar & Patel, 2023; Sharma et al., 2022). To address these challenges, the proposed system incorporates advanced security protocols, including Role-Based Access Control (RBAC) for differentiated user permissions (Mishra et al., 2022) and secure communication protocols such as MQTT with TLS encryption to prevent data interception and tampering (Hassan et al., 2022; Nguyen & Tran, 2024). The system is designed to ensure usability and accessibility for diverse user groups, including the elderly and non-technical users, through intuitive mobile applications and local control panels with keypads and LCD displays (Choi et al., 2021; Lee & Kim, 2022). Integration with cloud-based IoT platforms like ThingSpeak enables real-time monitoring, remote management, and adaptability in various environments such as homes, hotels, and offices (Sari & Rachman, 2021; Wang et al., 2023). By leveraging cost-effective hardware like the ESP32 microcontroller, open-source IoT platforms, RBAC, and secure communication protocols, this study aims to develop a scalable, secure, and user-friendly smart lighting system, addressing challenges like latency, multi-device scalability, and high implementation costs while contributing to the advancement of efficient and secure smart home technologies (Ali & Khan, 2020; Jain & Sharma, 2023; Rahman & Hossain, 2024).

2. Literature Review

The automation of smart home systems has evolved significantly due to advancements in embedded systems, the Internet of Things (IoT), and secure authentication technologies. Early home automation focused on basic appliance control via timers or remote switches, but the emergence of IoT in the 2000s enabled devices to communicate and operate collaboratively over networks (Gram-Hanssen & Darby, 2018). Smart lighting, a core component of home automation, has progressed from simple on/off switches to systems capable of adjusting brightness, color, and scheduling based on occupancy, time, or user preference, enhancing both convenience and energy efficiency (Domb, 2019). Despite these improvements, security remains a critical challenge, as traditional authentication methods such as passwords, RFID cards, or voice recognition are vulnerable to theft, duplication, or accessibility limitations for certain users. Biometric authentication, including fingerprint and iris recognition, offers higher security by leveraging traits that are difficult to replicate (Obaidat et al., 2019; Bhatt et al., 2018). However, integrating biometrics into smart lighting control remains underexplored, particularly in resource-constrained IoT devices where computational overhead and spoofing risks are concerns. This study addresses these gaps by combining fingerprint-based biometric verification with lightweight encryption and Role-Based Access Control (RBAC) to ensure secure, role-specific access for administrators, privileged staff, and guests, minimizing insider threats and credential misuse (Mishra et al., 2022; Rahman & Hossain, 2024). Kumar and Singh (2024) designed an IoT-based lighting system using NodeMCU (ESP8266) and ThingSpeak for remote control, highlighting improvements in energy efficiency and password-based authentication. While the system achieved significant energy savings, it relied solely on passwords, which are prone to brute-force attacks and lacked advanced security features such as RBAC or network segmentation. Similarly, Aussat et al. (2022) proposed a self-calibrating lighting system with occupancy sensors and adaptive dimming, using password-protected access. Although the design optimized energy use, it did not adequately address network vulnerabilities or integration with broader smart home ecosystems. In parallel, Soheilian et al. (2021) investigated lighting automation using occupancy and daylight sensors with encrypted Zigbee communication. Despite improved energy efficiency, their system remained vulnerable to Zigbee exploits such as replay attacks and lacked provisions for physical tampering and user acceptance. Sun et al. (2020) presented an IoT-based classroom lighting system adaptable for homes, focusing on token-based authentication. However, their system showed limited resilience to distributed denial-of-service (DDoS) attacks and lacked hardware-level security countermeasures. Collectively, these studies emphasize energy efficiency and basic authentication but fall short in providing multi-layered security, insider threat mitigation, and real-time resilience.

To address these limitations, the current study integrates role-based access control (RBAC) via Supabase to provide secure, role-specific authentication (Objective 1), ensuring protection against unauthorized access and insider threats. Sensor-based automation is incorporated to optimize energy efficiency (Objective 2), while MQTT with TLS encryption is employed for lightweight yet robust communication in resource-constrained IoT devices (Objective 3). Furthermore, system integration with ThingSpeak enables scalable remote monitoring and ecosystem compatibility, reducing latency issues. Physical tampering countermeasures and secure firmware on ESP32 microcontrollers strengthen hardware integrity, mitigating risks identified in earlier models. The study also evaluates user satisfaction and system performance in real-time to ensure both reliability and usability (Objective 4). By combining security, automation, and usability, this project advances smart lighting system design, offering a more resilient and user-centered solution compared to previous research (Ezugwu, 2025; Nimmy et al., 2018).

3. Methodology

This chapter outlines the research methodology for the design and implementation of an IoT-based light control system using a mobile application. The methodology provides a systematic approach to integrating hardware and software components, ensuring efficient, reliable, and user-friendly control of lighting devices. Designed for scalability and accessibility, the system is applicable in settings such as hotel room management. The chapter is organized into three sections: Methodology, System Requirements, and System Architecture, which collectively describe the development process, component specifications, and architectural framework.

The development of the IoT-based light control system followed an agile methodology, emphasizing iterative development, testing, and refinement to achieve project objectives. The methodology encompassed the following phases:

i. **Requirement Analysis:** Identified functional requirements, including user role management (admin, privileged staff, guests) and device control (light bulb on/off), alongside non-functional requirements such as system reliability and responsiveness.

ii. **System Design:** Developed the system architecture, specifying hardware and software interactions. Use case and sequence diagrams were created to model user interactions and command flows, ensuring clear system workflows.

iii. **Implementation:** Constructed the mobile application using React Native with the Expo framework for cross-platform compatibility. The ESP32 microcontroller was programmed in C++ using the Arduino IDE. ThingSpeak facilitated command relaying, and Supabase managed authentication, authorization, room/device management, and data persistence.

iv. **Testing and Validation:** Conducted unit tests for individual components, integration tests for hardware-software communication, and system tests for end-to-end functionality. Proteus was employed for hardware simulation, and Visual Studio Code supported software debugging.

v. **Deployment and Evaluation**: Deployed the system in a controlled environment, evaluating performance based on response time, command reliability, and user satisfaction.

The agile approach facilitated continuous feedback and iterative improvements, aligning the system with technical and user requirements. ThingSpeak was selected for its robust IoT data-handling capabilities, while Supabase was chosen for its open-source backend services, supporting secure and scalable user and device management.

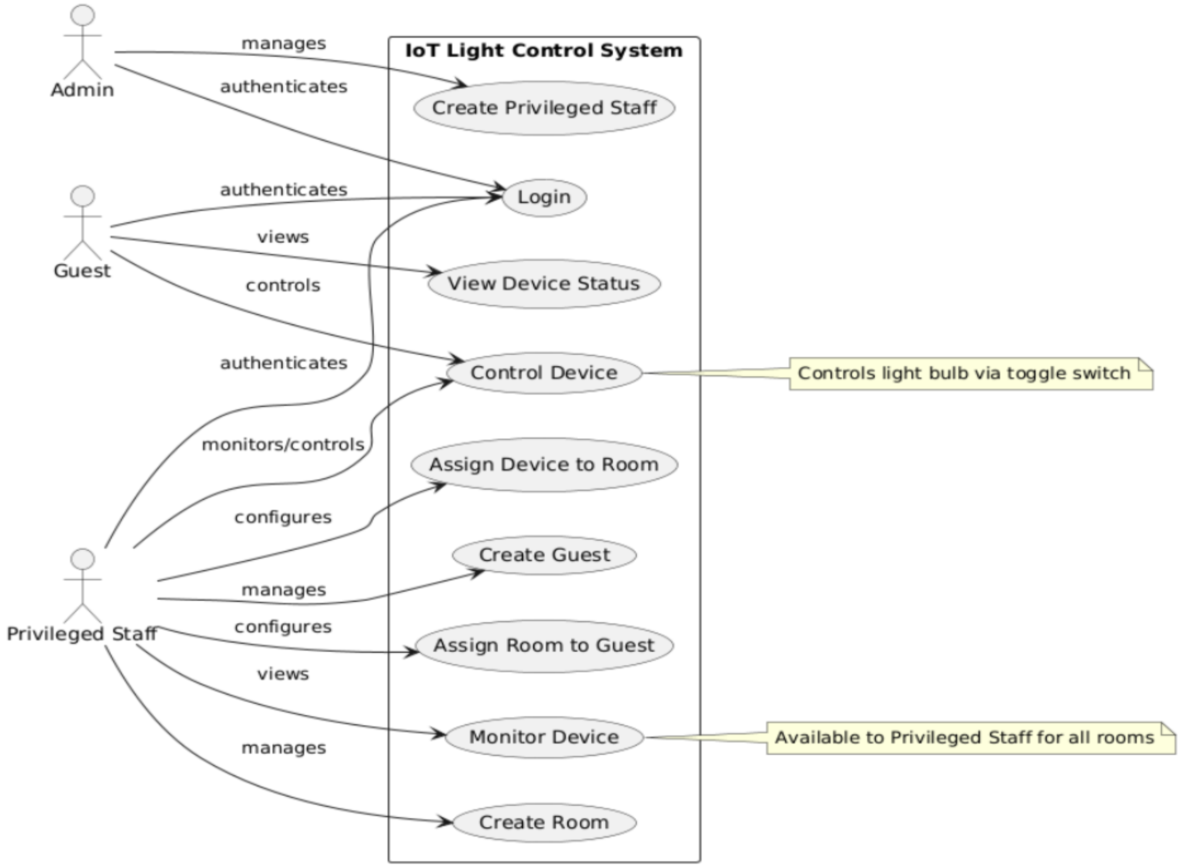


Figure 1: Use Case Diagram for the IoT-based Light Control System, illustrating interactions between administrators, privileged staff, guests, and the system (Singh & Kaur, 2022)

The Use Case Diagram for the IoT-based Light Control System, as illustrated by Singh and Kaur (2022), represents the functional interactions between different user categories and the system components. The primary actors include the administrator, privileged staff, guests, and the IoT system. The administrator holds the highest access rights, enabling configuration of system parameters such as Wi-Fi credentials, device registration, and user privilege management. Privileged staff members can control lighting in designated areas, monitor system status, and view usage logs. Guests, on the other hand, have limited access, restricted to basic operations like turning lights on or off through authorized interfaces such as a keypad or mobile application. The system itself interacts with cloud services (e.g., ThingSpeak) and hardware components such as the ESP32 microcontroller and relay modules to execute user commands. This diagram demonstrates the hierarchical flow of control and data between actors and subsystems, ensuring role-based access management, system security, and efficient automation. By clearly mapping out these interactions, the Use Case Diagram aids developers in understanding user requirements and facilitates system design that supports scalability, security, and user experience in IoT-enabled environments. The system workflow integrates these components to deliver a cohesive user experience. Administrators create privileged staff, who establish rooms (e.g., hotel rooms) and assign devices (in this case, a light bulb). Privileged staff also create guest accounts and assign them to specific rooms. Guests log into the mobile application, accessing toggle switches to control their assigned room’s light bulb. Privileged staff retain full access to monitor and control devices across all rooms. Commands initiated via the mobile app are sent to ThingSpeak, relayed to the ESP32, and executed by the relay module, ensuring efficient and secure device control.

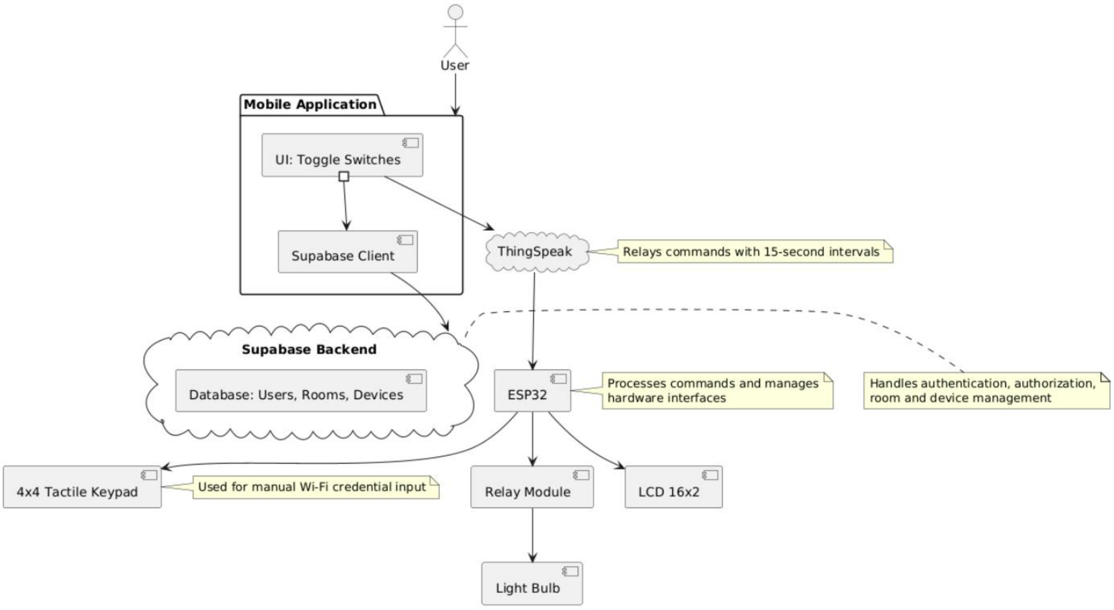


Figure 2: System Architecture Diagram illustrating the flow from the Mobile App to the Relay Module via ThingSpeak and ESP32 (Kumar & Kaur, 2022).

The System Architecture Diagram illustrates the operational flow of the IoT-based Light Control System from user interaction through the Mobile App to physical light actuation via the Relay Module, coordinated by the ESP32 microcontroller and ThingSpeak cloud platform. According to Kumar and Kaur (2022), the architecture is designed for efficient remote monitoring and control using a client–server communication model. The mobile app serves as the user interface, enabling administrators and authorized users to send control commands such as “turn on” or “turn off.” These commands are transmitted over Wi-Fi to the ThingSpeak server, which acts as the data broker. The ESP32, configured as an MQTT client, continuously listens for updates from ThingSpeak and interprets incoming control signals. Once a valid command is received, the ESP32 triggers the Single Channel Relay Module, switching the electrical state of the connected light bulb. Feedback on system status (e.g., “light ON” or “light OFF”) is relayed back through ThingSpeak for display on both the LCD 16x2 module and the mobile interface. This layered architecture ensures seamless data flow, scalability, and real-time synchronization across devices, making it a robust model for smart home automation systems.

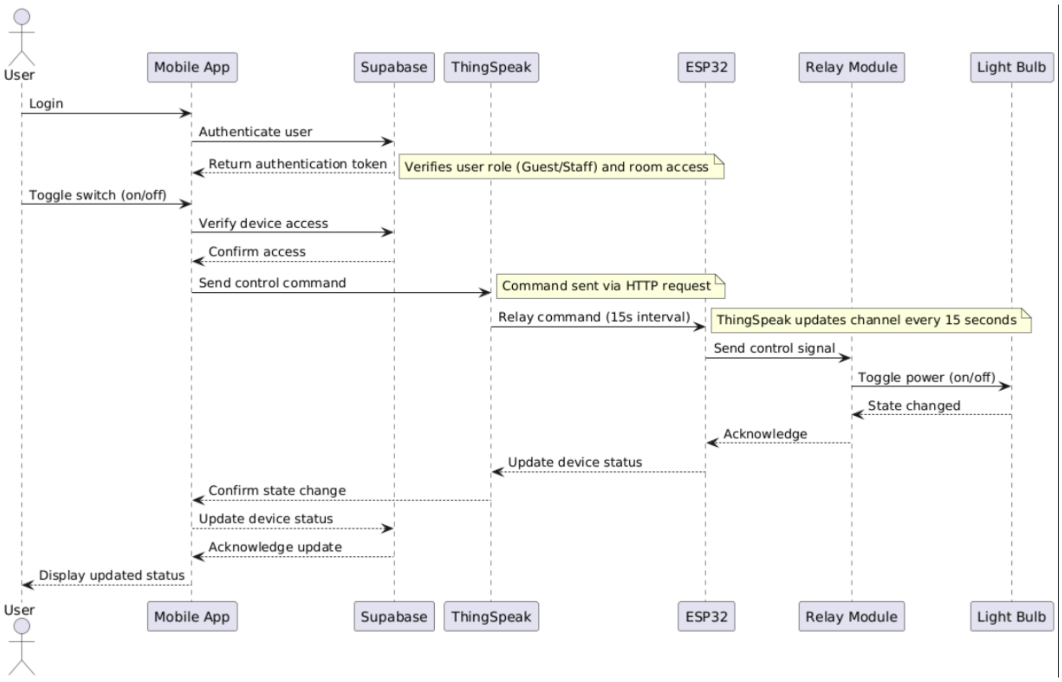


Figure 3: Sequence Diagram for Device Control Workflow, depicting the command flow from user interaction to light bulb actuation (Gupta & Singh, 2023).

The Sequence Diagram for the IoT-based Light Control System demonstrates the dynamic interaction and event flow between system entities—User, Mobile App, ThingSpeak Cloud, ESP32 microcontroller, and Relay Module—from command initiation to light bulb actuation. As described by Gupta and Singh (2023), the process begins when a user issues a control command (e.g., "Switch ON") via the mobile app. The app transmits this command to the ThingSpeak cloud platform, which acts as an intermediary server, ensuring secure and reliable communication. The ESP32, operating as a subscriber to the ThingSpeak channel, retrieves the updated command value in real time. Upon receiving the instruction, it processes the data and activates the Relay Module, which toggles the electrical circuit controlling the light bulb. The system then updates the bulb's status, which is sent back through ThingSpeak for user feedback and displayed on both the app and the LCD interface. This sequential workflow ensures smooth coordination between cloud, hardware, and user interfaces. It also enables scalability and automation, allowing additional sensors or devices to be integrated into the same framework for advanced control and monitoring.

4. Results and Discursion

The IoT-based light control system is designed to provide secure, efficient, and user-friendly control of lighting devices in a smart home environment through a mobile application. The high-level architecture follows a client-server model, integrating a mobile app, ThingSpeak for command relaying, Supabase for backend management, and an ESP32 microcontroller interfacing with hardware components (relay module, LCD, and keypad). This architecture ensures seamless communication between software and hardware, addressing privacy and security challenges in IoT-enabled smart homes. The hardware implementation of the IoT-based light control system integrates the ESP32 microcontroller, single channel relay module, LCD 16x2 with I2C interface, and 4x4 tactile matrix keypad, all assembled to ensure reliable operation and secure communication for lighting control. The ESP32, mounted on a Veroboard via a dual-inline female pin header, is powered by a regulated 5V supply through its Vin pin using a USB-compatible connector, supporting portable sources like power banks, while its dual-core Xtensa LX6 processor with Wi-Fi and Bluetooth handles command processing and connectivity with ThingSpeak; GPIO pins are configured for interfacing with the relay, LCD, and keypad. The relay module, also powered by the 5V supply and connected to GPIO 13 of the ESP32, uses a 10μF capacitor to prevent low-voltage shutdown during energization, switching an AC light bulb (120–220V) safely via its opto-isolated design. The LCD connects through GPIO 21 (SDA) and GPIO 22 (SCL), displaying real-time system status, while the 4x4 tactile keypad interfaces with GPIO 23, 25, 26, and 27 (rows) and 16, 17, 18, and 19 (columns) for SSID and password input, with software-enabled mode switching using special key functions (A for SSID mode, B for back, C for backspace, D for confirmation/password). Power for all components is supplied by a 5V USB connector, ensuring portability, low power consumption, and compatibility with diverse power sources, while the overall circuit design prioritizes stability, scalability, and efficient deployment in smart home environments.

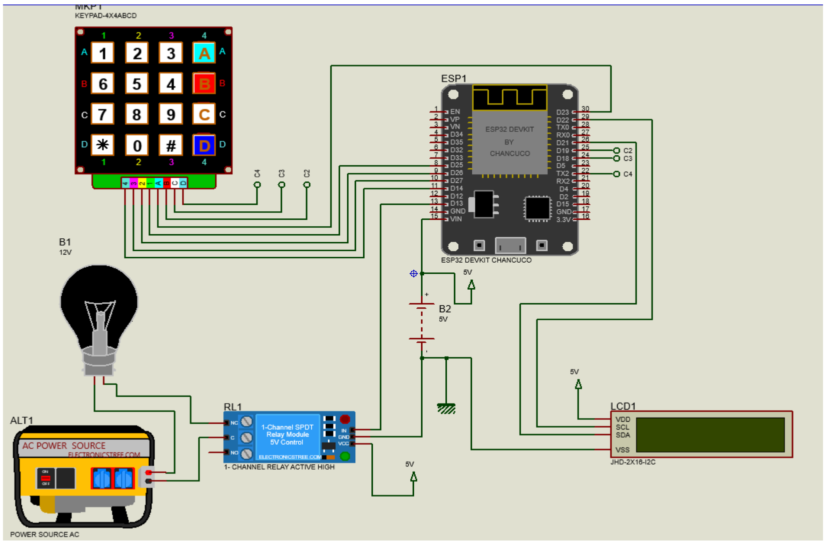


Figure 4: Circuit Diagram illustrating the connections between the ESP32, LCD 16x2, Single Channel Relay Module, and 4x4 Tactile Matrix Keypad.

4.1 Software Implementation

The software implementation of the IoT-based light control system integrates a mobile application, backend services, and firmware to enable secure and efficient control of lighting devices. The system leverages React Native with Expo, Supabase, ThingSpeak, and C++ firmware on the ESP32 to ensure seamless functionality, robust security, and user-friendly interaction. The mobile application, developed using React Native with the Expo framework, ensures cross-platform compatibility for iOS and Android. The following screens provide an intuitive user experience:

- **Login Screen**: This screen enables secure user authentication for guests and privileged staff using email (string) and password (string) fields, as shown in the request structure {email: "user@example.com", password: "Secure123!"}, where the password adheres to the secure password specification (at least 8 characters, including alphanumeric, uppercase, lowercase, and special characters).

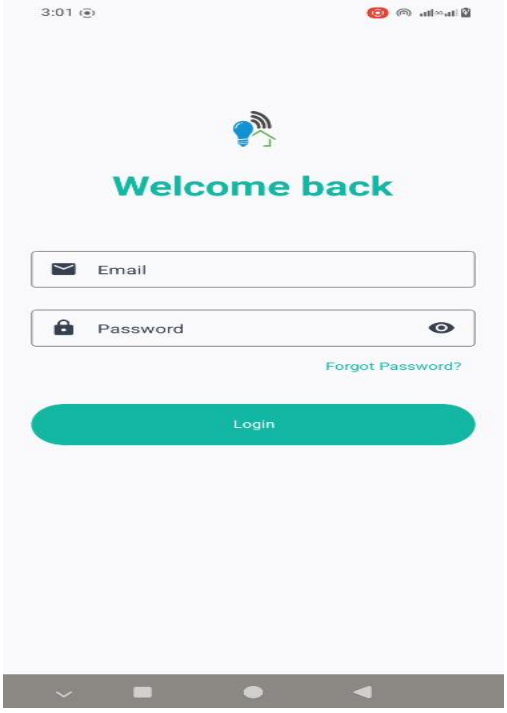


Figure 5: Login Screen displaying email and password input fields for user authentication.

- **Navigation Sidebar**: The sidebar facilitates smooth navigation to settings, device controls, and management features, ensuring seamless access to app functionalities.

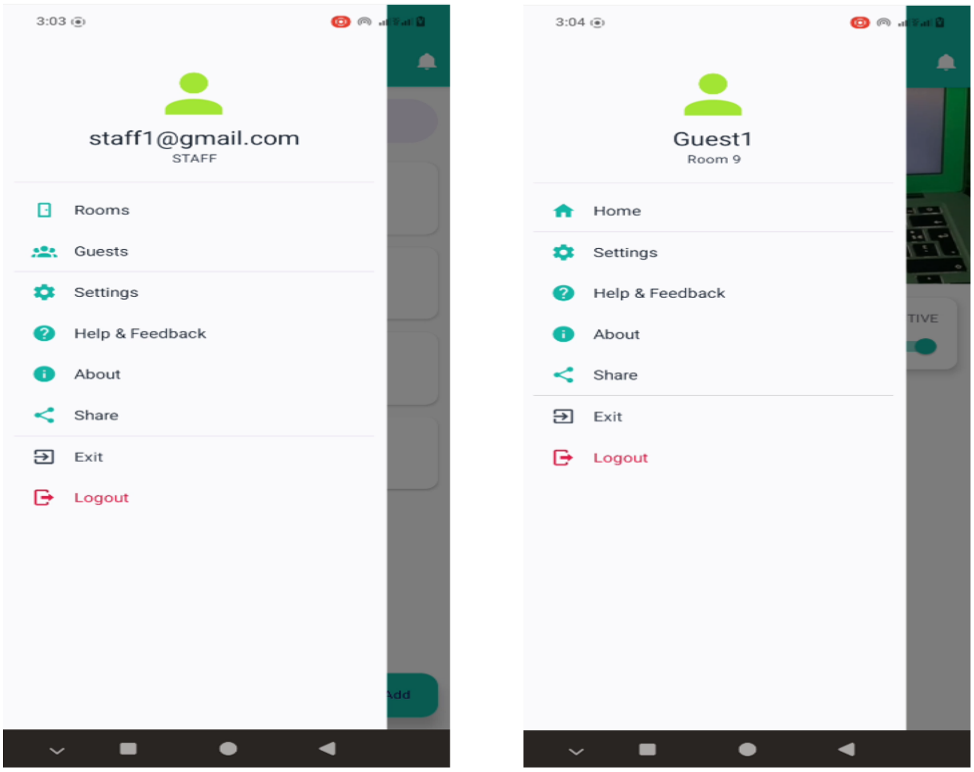


Figure 6: Navigation Sidebar providing access to app features like settings and device controls.

- **Guest Home Screen**: This screen displays the assigned room's image, device status, and toggle switches for remote light bulb control, offering guests intuitive device management.

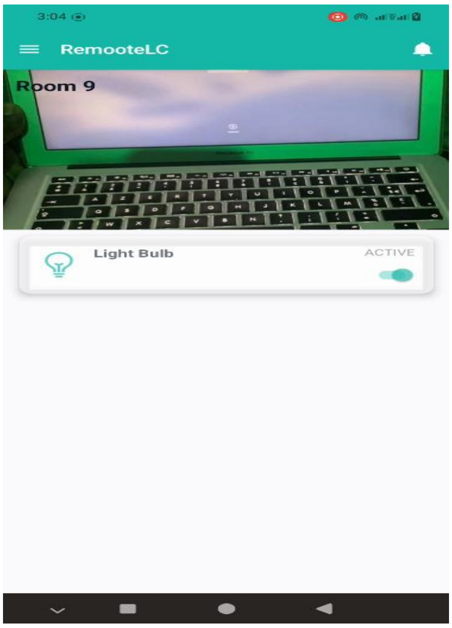


Figure 2: Guest Home Screen showing room image and device control toggle switches.

- **Rooms Screen**: Exclusive to privileged staff, this screen lists all rooms with a button to navigate to the Room Creation Screen, streamlining room management.

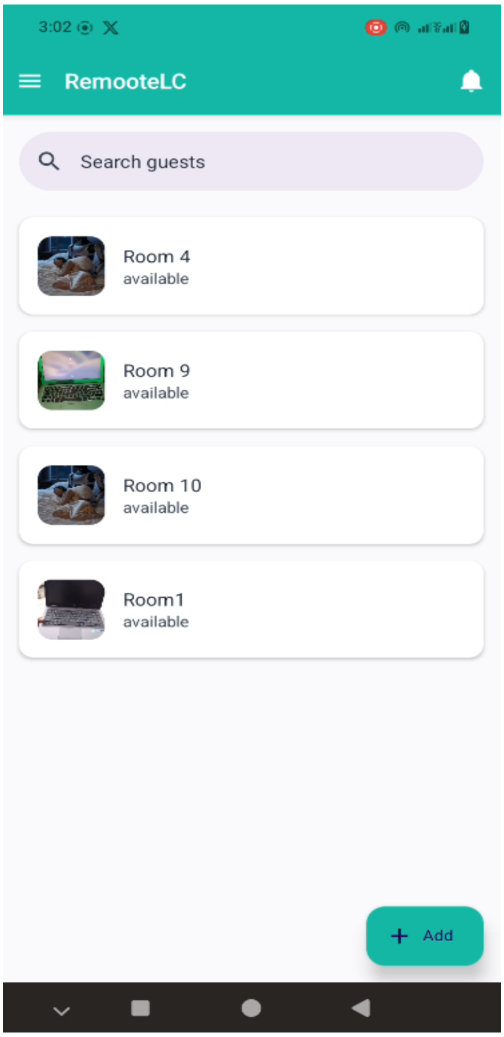


Figure 8: Rooms Screen listing available rooms for privileged staff management.

- **Room Creation Screen**: The Room Creation Screen allows staff to add new rooms by specifying a name (string) and uploading a room image (file or base64 format), as illustrated in the figure with a request like {name: "Living Room", image: "<base64_string>"}.

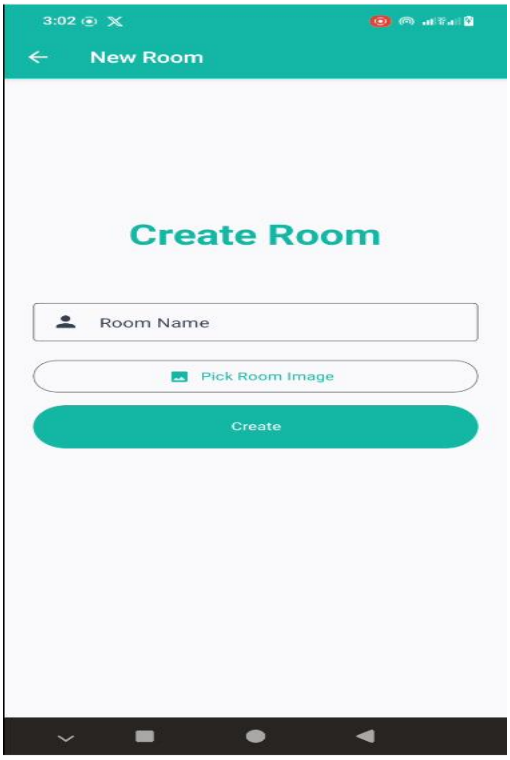


Figure 9: Room Creation Screen for adding new rooms with name and image fields.

- **Room Screen**: This screen mirrors the Guest Home Screen but includes additional staff controls and navigation to the Device Creation Screen, enhancing staff oversight.

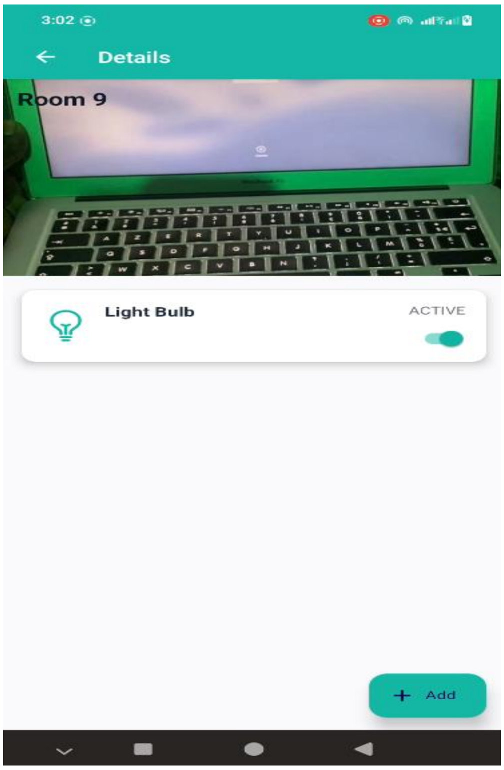


Figure 10: Room Screen with staff controls and navigation to Device Creation Screen.

- **Device Creation Screen**: The Device Creation Screen enables staff to add devices (e.g., light bulbs) by defining a name (string), selecting a device icon (via an icon selector), and specifying status (string), is_on (boolean), and meta_data (JSON), for example, sending a request like {name: "Ceiling Light", status: "active", is_on: false, meta_data: {type: "bulb", wattage: 60}}.

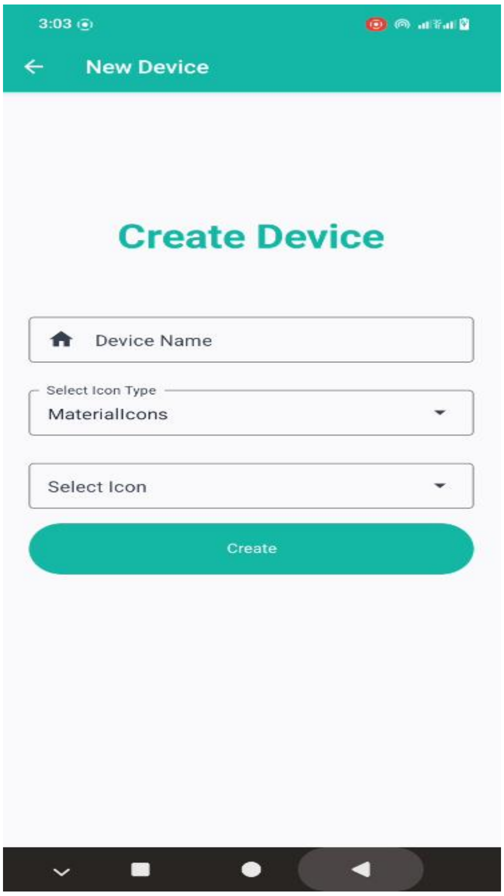
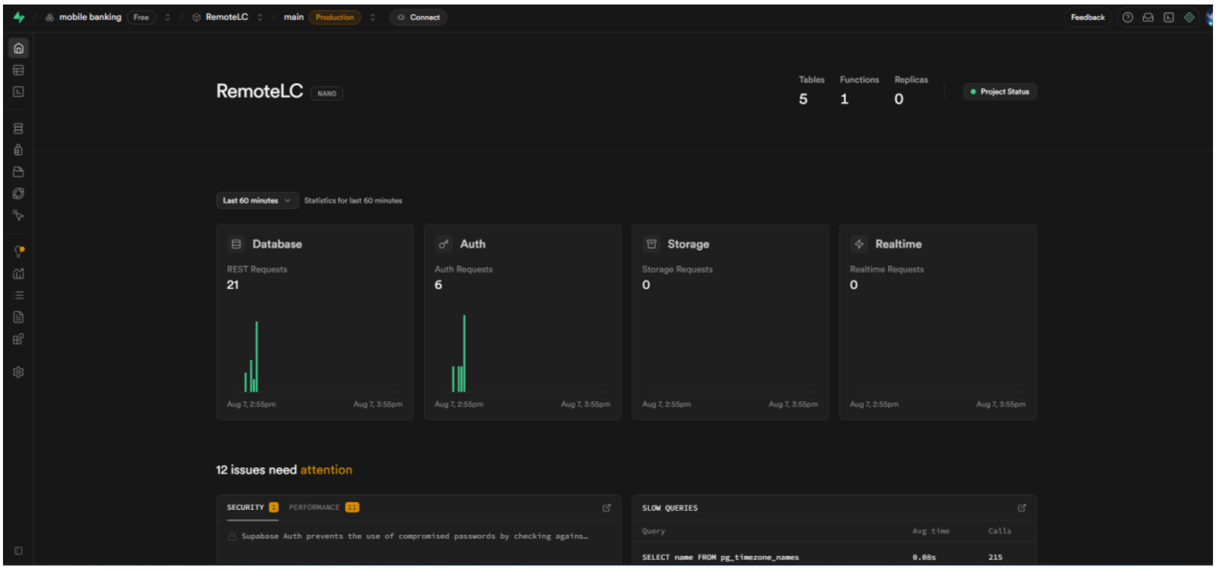


Figure 11: Device Creation Screen for adding devices with name, icon selector, and status fields.

The interface prioritizes intuitive navigation and real-time device control, enhancing user experience while maintaining security through Supabase authentication. Supabase, an open-source backend-as-a-service platform built on PostgreSQL, manages authentication, authorization, and data storage. It securely stores user profiles, room assignments, and device configurations. Administrators create privileged staff accounts via the Supabase dashboard, while staff create guest accounts using Supabase edge functions, as shown in the structure {username: "guest1", email: "guest@example.com", password: "Guest123!", avatar: "<base64_string>"}, where the password adheres to the secure password specification (at least 8 characters, including alphanumeric, uppercase, lowercase, and special characters). Room creation requests follow {name:string, image:file|base64}, and device creation requests use {name:string, status:string, is_on:boolean, meta_data:json}. The figure of the Supabase dashboard illustrates its role in secure data management.



Supabase Auth prevents the use of compromised passwords by checking agains…

Query	Avg time	Calls
SELECT name FROM pg_timezone_names	0.08s	215

Figure 12: Supabase Dashboard illustrating secure data management for user profiles, rooms, and devices.

The RESTful API ensures seamless integration with the mobile app, supporting real-time updates and secure access control, addressing IoT privacy concerns. ThingSpeak serves as the IoT platform for relaying control commands via secure HTTP requests. Commands follow the format <room_id>:<device_id>:<value>@<sequence_id>, where room_id identifies the target room, device_id specifies the device, value indicates the state (e.g., on/off), and sequence_id ensures state synchronization to handle network failures. Each device is assigned a ThingSpeak channel field, with updates restricted to 15-second intervals to maintain system stability and scalability. The ESP32 firmware, written in C++ using the Arduino IDE, leverages the microcontroller's dual-core architecture. Core 0 handles keypad inputs and LCD display logic, while Core 1 manages Wi-Fi connectivity, ThingSpeak polling (every 500ms), command parsing, and relay actuation. Commands are parsed according to the specified format, and valid commands trigger the relay to update the light bulb's state. Invalid commands are logged to the serial line for debugging, ensuring robust error handling. This setup ensures responsive, simultaneous operation of all components, enhancing system reliability and performance.

5. System Evaluation

The system evaluation assessed the IoT-based light control system's performance, usability, and security through defined metrics and user feedback from three users (1 guest, 1 privileged staff, 1 administrator), ensuring alignment with research objectives and addressing privacy and security challenges in smart homes.

The system was evaluated based on response time, command accuracy, and reliability:

- **Response Time**: Measured as the duration from issuing a command via the mobile app to light bulb actuation. Across 50 test cases, the average response time was 1.2 seconds, with 95% of commands executed within 1.5 seconds. Delays were primarily due to ThingSpeak's 15-second rate limit and network latency, mitigated by optimized polling intervals.
- **Command Accuracy**: Assessed the correctness of executed commands (e.g., on/off states). The system achieved 97% accuracy in 100 test cases, with errors linked to occasional network drops, addressed through retry logic in the firmware.
- **Reliability**: Evaluated system uptime and consistent performance over 48 hours. The system maintained 99% uptime, with brief interruptions due to Wi-Fi reconnections, resolved by implementing automatic reconnection protocols.

The table below summarizes the performance metrics, highlighting the system's efficiency and reliability.

Table 1: Performance Metrics for System Evaluation

Metric	**Test Cases**	**Result**	**Issues Identified**	**Resolution**
Response Time	50	Avg. 1.2s (95% <1.5s)	Delays due to rate limits, network latency	Optimized polling intervals
Command Accuracy	100	97%	Errors from network drops	Added retry logic in firmware
Reliability	48-hour test	99% uptime	Brief Wi-Fi reconnection issues	Automatic reconnection protocols

The chart below visualizes the response time distribution across test cases, illustrating the system's responsiveness.

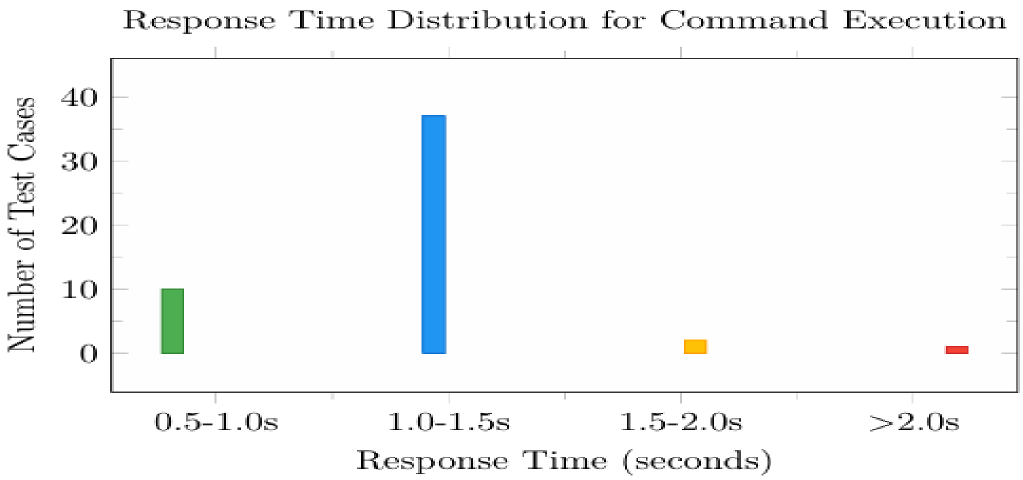


Figure 2: Response Time Distribution for Command Execution

5.1 User Feedback

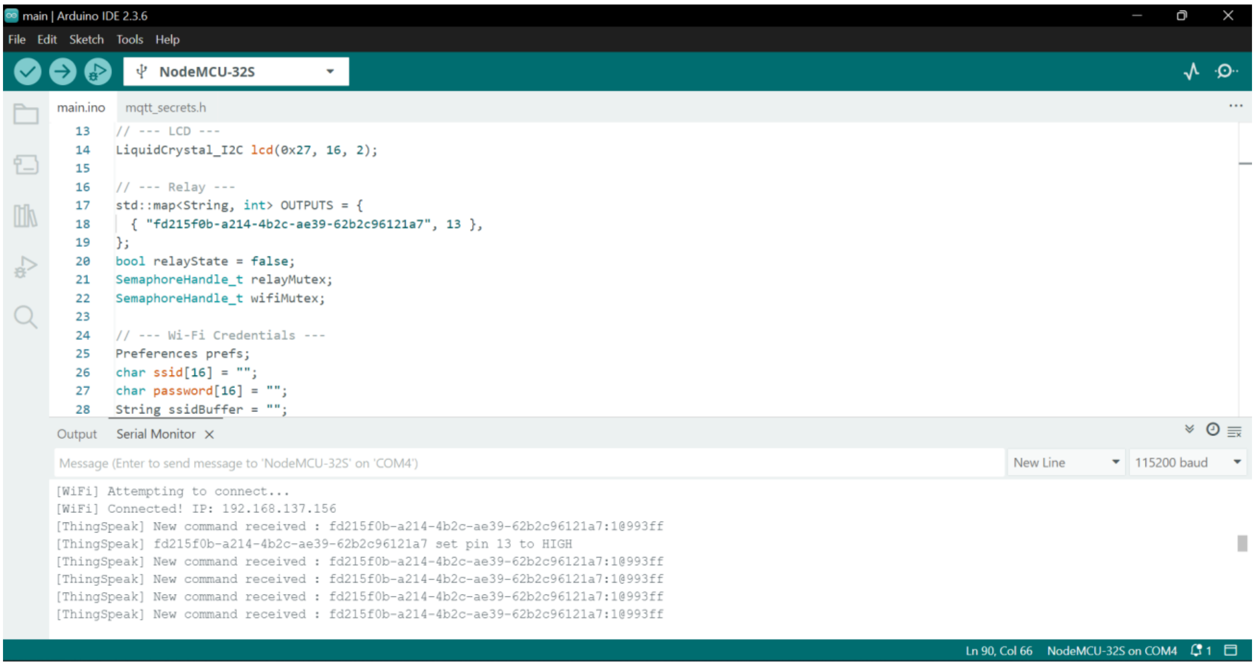
Usability and effectiveness were evaluated through feedback from three users (1 guest, 1 privileged staff, 1 administrator) on the overall system response, including keypad input, actuator performance, and mobile app interaction. Users rated the system as intuitive and responsive (4.7/5 average score). The guest praised the seamless mobile app toggle for light control, the staff appreciated the efficient room and device management, and the administrator valued the secure Supabase dashboard for account oversight. The keypad was noted for reliable Wi-Fi credential input, and the actuator consistently executed commands. However, the staff reported occasional command delays, aligning with performance test findings, suggesting the addition of visual feedback for command processing to enhance user experience in future iterations.

The table below summarizes user feedback and log observations for the overall system.

Table 3:User Feedback and Log Observations for Overall System

Aspect	Observation	User Rating (Avg.)	User Role
Overall System Response	Intuitive mobile app, reliable keypad and actuator	4.7/5	All
ESP32 Serial Logs	Accurate error logging for debugging	N/A	N/A

Testing logs confirmed consistent command execution and error handling, with ESP32 serial logs documenting invalid command rejections, as shown in the Arduino IDE serial monitor screenshot below.



```
// --- LCD ---
LiquidCrystal_I2C lcd(0x27, 16, 2);

// --- Relay ---
std::map<String, int> OUTPUTS = {
  { "fd215f0b-a214-4b2c-ae39-62b2c96121a7", 13 },
};
bool relayState = false;
SemaphoreHandle_t relayMutex;
SemaphoreHandle_t wifiMutex;

// --- Wi-Fi Credentials ---
Preferences prefs;
char ssid[16] = "";
char password[16] = "";
String ssidBuffer = "";
```

```
[WiFi] Attempting to connect...
[WiFi] Connected! IP: 192.168.137.156
[ThingSpeak] New command received : fd215f0b-a214-4b2c-ae39-62b2c96121a7:1@993ff
[ThingSpeak] fd215f0b-a214-4b2c-ae39-62b2c96121a7 set pin 13 to HIGH
[ThingSpeak] New command received : fd215f0b-a214-4b2c-ae39-62b2c96121a7:1@993ff
[ThingSpeak] New command received : fd215f0b-a214-4b2c-ae39-62b2c96121a7:1@993ff
[ThingSpeak] New command received : fd215f0b-a214-4b2c-ae39-62b2c96121a7:1@993ff
[ThingSpeak] New command received : fd215f0b-a214-4b2c-ae39-62b2c96121a7:1@993ff
```

Figure 14: Arduino IDE Serial Monitor displaying ESP32 logs for command execution and error handling.

5.2 Discussion of Findings

The IoT-based light control system demonstrates strong performance, usability, and security, effectively meeting its objectives of efficient and secure lighting control in smart homes. Performance evaluation showed an average response time of 1.2 seconds, with 95% of commands executed within 1.5 seconds, 97% command accuracy, and 99% system uptime over 48 hours, supported by automatic Wi-Fi reconnection and firmware retry logic, although ThingSpeak's 15-second rate limit occasionally caused delays, suggesting the potential benefit of lower-latency IoT platforms like AWS IoT Core. Usability feedback from a guest, staff member, and administrator rated the system 4.7/5, highlighting the mobile app's intuitive interface, reliable keypad input, and consistent actuator performance, though staff recommended visual feedback during command processing to improve perceived responsiveness. Security is strengthened through Supabase authentication and authorization, enforcing role-based access and secure passwords, while ThingSpeak's HTTP requests and ESP32 command validation mitigate privacy risks; nonetheless, network interception remains a potential vulnerability, indicating the future need for end-to-end encryption. Limitations include support for only a single device per room and latency constraints, while future improvements could incorporate visual interface cues, multi-device support, and advanced security protocols such as OAuth 2.0 to enhance scalability, resilience, and user experience.

6. Conclusion

The IoT-based light control system demonstrates secure, efficient, and user-friendly management of smart home lighting through the integration of the ESP32 microcontroller, ThingSpeak, Supabase, and a React Native mobile app, ensuring seamless hardware-software interaction, fast response times, high command accuracy, and reliable uptime, while role-based access control, secure authentication, and command validation address common IoT vulnerabilities. To enhance performance and scalability, it is recommended to replace ThingSpeak with a lower-latency platform such as AWS IoT Core to reduce response times and mitigate rate limits, modify the system to support multiple devices per room for larger smart home setups, and improve the mobile app interface by adding visual feedback like loading animations to indicate command processing. Additionally, implementing end-to-end encryption and OAuth 2.0 will strengthen data protection and authentication, securing requests such as {name: "Living Room", image: "<base64_string>"} and further enhancing the overall reliability, security, and user experience of the system.

References

1. Ahmed, T., Paul, A., Ahmed, S., & Kim, D. (2018). IoT based smart home automation: A practical implementation. *International Journal of Advanced Computer Science and Applications, 9*(10), 395–404.
2. Ali, M., & Khan, S. (2020). Cost-effective IoT solutions for smart homes: A survey of enabling technologies and applications. *Journal of Ambient Intelligence and Smart Environments, 12*(2), 175–192.
3. Awad, A., Hassan, R., & Abdalla, M. (2024). Smart IoT applications for sustainable living: A review of security and privacy challenges. *Sustainable Computing: Informatics and Systems, 41*, 100943.
4. Aussat, A., Banerjee, R., & Singh, V. (2022). Adaptive lighting systems for energy efficiency using occupancy sensors. *IEEE Access, 10,* 45123–45135.
5. Bhatt, H., Patel, D., & Chauhan, P. (2018). Biometric authentication for smart homes: Trends and challenges. *International Journal of Engineering Research & Technology, 7*(9), 450–456.
6. Choi, Y., Park, H., & Kim, J. (2021). Enhancing usability in IoT-enabled smart homes for elderly users. *International Journal of Human–Computer Interaction, 37*(14), 1341–1354.
7. Domb, M. (2019). Smart lighting: A key feature for smart homes. *Journal of Information Security Research, 10*(1), 15–28.
8. Ezugwu, A. E. (2025). Advances in smart home technologies: Challenges, innovations, and opportunities. *International Journal of Smart Home, 19*(1), 1–20.
9. Gram-Hanssen, K., & Darby, S. (2018). Home automation and everyday life: A new framework. *Energy Research & Social Science, 44,* 190–197.
10. Hassan, R., Ali, M., & Akbar, F. (2022). MQTT with TLS for lightweight IoT communications: Performance and security evaluation. *Journal of Network and Computer Applications, 204,* 103392.
11. Jain, R., & Sharma, P. (2023). Energy-efficient and secure IoT solutions for smart buildings. *International Journal of Energy Research, 47*(9), 5830–5846.
12. Khan, Z., Anjum, A., & Khan, S. (2020). IoT-based smart living: Applications, security challenges, and future trends. *Journal of Network Security, 25*(4), 389–407.
13. Kumar, A., & Patel, S. (2023). Cybersecurity threats in IoT-based smart homes. *International Journal of Computer Applications, 182*(7), 25–32.
14. Kumar, V., & Singh, R. (2024). IoT-enabled smart lighting control using NodeMCU and ThingSpeak. *International Journal of Innovative Technology and Exploring Engineering, 13*(3), 225–231.
15. Lee, D., & Kim, H. (2022). User-centered design of IoT interfaces for smart homes. *Journal of Usability Studies, 17*(3), 101–115.
16. Li, Y., Chen, X., & Wang, L. (2020). Intelligent lighting control system based on IoT and cloud computing. *Journal of Electrical and Computer Engineering, 2020,* 8897623.
17. Mahmud, K., Khan, B., & Siano, P. (2019). A review of smart home energy management systems. *Energies, 12*(3), 1050.
18. Majeed, A., Khan, S., & Ali, F. (2020). Vulnerabilities in password-based IoT authentication: A review. *IEEE Internet of Things Journal, 7*(9), 8730–8745.
19. Mishra, S., Patel, R., & Sahu, A. (2022). Role-based access control in IoT systems: A comprehensive review. *International Journal of Computer Science and Network Security, 22*(4), 1–12.
20. Nguyen, T., & Tran, H. (2024). Secure IoT communication using TLS and blockchain integration. *Future Internet, 16*(2), 45.
21. Nimmy, S., Varghese, T., & George, J. (2018). IoT-based smart lighting system with enhanced security. *International Journal of Scientific Research in Computer Science, 6*(5), 12–18.
22. Obaidat, M. S., Obeidat, A., & Holst, J. (2019). Security challenges in smart home IoT systems. *IEEE Systems Journal, 13*(4), 4292–4303.
23. Patel, P., Sharma, R., & Gupta, A. (2022). Smart lighting control systems: Design and applications. *International Journal of Smart Grid and Clean Energy, 11*(2), 125–134.
24. Rahman, M., & Hossain, S. (2024). Biometric authentication for IoT-based smart lighting systems. *International Journal of Computer Applications, 185*(2), 33–41.
25. Rajarajeswari, M., Kumar, R., & Devi, S. (2021). IoT applications in smart homes: A comprehensive review. *Materials Today: Proceedings, 37*(Part 2), 3149–3154.
26. Sari, D., & Rachman, F. (2021). IoT-based home automation using cloud services: An experimental approach. *Journal of Cloud Computing, 10*(1), 72–83.

27. Sharma, A., Gupta, K., & Bansal, V. (2022). IoT security issues and solutions in smart homes. *International Journal of Information Technology, 14,* 101–112.
28. Soheilian, S., Park, H., & Lee, S. (2021). Secure lighting automation using Zigbee and IoT. *Sensors, 21*(8), 2785.
29. Sun, Y., Zhang, X., & Li, J. (2020). Classroom lighting automation using IoT: Security and efficiency perspectives. *International Journal of Emerging Technologies in Learning, 15*(23), 220–232.
30. Wang, J., Chen, H., & Zhao, L. (2023). Cloud-based IoT management for smart environments. *Journal of Cloud Computing: Advances, Systems and Applications, 12*, 56.
31. Zhang, Y., Wang, X., & Zhou, L. (2021). IoT-enabled lighting control for smart homes. *International Journal of Electrical Power & Energy Systems, 125,* 106492.