

Implementation of Internet of Things Devices in Website-Based Smart Homes

M. Naja Annajafi Asyfhani^{1*}, Fara Triadi², Agus Triyono³

^{1,2,3}Department of Information Technology, State Polytechnic of Samarinda, Indonesia
mnaja7742@gmail.com^{1*}, fara_triadi@polnes.ac.id², triyono@polnes.ac.id³

Abstract

The rise of the Internet of Things (IoT) has accelerated the implementation of smart home systems, enabling household devices to be monitored and controlled remotely. This study focuses on developing a web-based smart home system using web technology, integrating Django, Supabase (PostgreSQL), Raspberry Pi, and Message Queuing Telemetry Transport (MQTT) protocol. The proposed system is implemented with a database-driven control approach where user commands are stored in the Supabase (PostgreSQL) cloud database and executed by Raspberry Pi via MQTT, while sensor data are synchronized and displayed on the web dashboard. The system also stores the historical activity of the devices, which is then analyzed in Google Colab to estimate the energy consumption. The results of the implementation show that the proposed system works successfully according to the designed architecture. Sensor data are monitored in real-time, actuators respond correctly to commands from users, and historical data are stored consistently for later analysis. The combination of Django, Supabase, Raspberry Pi, and MQTT shows that it is possible to have a unified, web-based smart home platform that provides real-time monitoring, remote control of devices, historical data management, and energy consumption analysis.

Keywords: *Django; Internet of Things; MQTT; Raspberry Pi; Smart Home; Supabase*

1. Introduction

The rapid development of the Internet of Things (IoT) has accelerated the adoption of smart home technology, enabling household devices to be monitored and controlled remotely through internet-connected systems. Smart home technology integrates sensors, actuators, and communication networks to improve convenience, security, and energy efficiency by automating various household functions [1], [5]. In addition to remote control, smart home systems provide real-time monitoring of environmental conditions, allowing users to respond more quickly to potentially hazardous situations such as gas leakage and fire [1], [8], [9], while also supporting more efficient household energy management [1].

But even with these benefits, household management still faces a number of practical challenges. Often electrical appliances are left running by accident, causing wasteful energy use. Dangerous situations, such as a gas leak or fire can arise without being noticed immediately. Additionally, traditional control systems are frequently tied to particular mobile applications or closed platforms, potentially restricting access and flexibility. These challenges emphasize the importance of a smart home platform that enables continuous monitoring, remote control of devices and structured management of historical data, through a web-based interface.

Several previous studies have developed IoT-based smart home systems using different architectures and technologies. An Android-based smart home application integrated with Firebase has been proposed for real-time monitoring and device control [6]. A web-based smart home system with RFID authentication has also been developed [3]. Another study implemented a Raspberry Pi-based system for remote control of household electrical devices [4], while a NodeMCU- and Blynk-based smart home system has been developed for device control and temperature monitoring [5]. In addition, IoT-based smart home applications have been proposed for fire detection [8] and gas leakage detection [9]. Although these studies successfully demonstrated monitoring, remote control, and hazard detection, most of them primarily focus on individual smart home functions and provide limited support for integrated historical data management and subsequent energy consumption analysis.

Considering these limitations, this study proposes a web-based smart home system by integrating Django, Raspberry Pi, Supabase (PostgreSQL), and Message Queuing Telemetry Transport (MQTT) protocol. In the proposed system, a database driven control mechanism

is used, where user commands are stored in the cloud database and executed by Raspberry Pi via MQTT communication. Besides, sensor readings and actuator states are continuously stored as historical data for further analysis of the energy consumption with the help of Google Colab. This integrated architecture combines real-time monitoring, remote device control, historical data management and energy consumption analysis into one web-based platform. Therefore, the purpose of this study is to design and implement a web-based smart home system that can provide real-time monitoring, remote controlling of devices and historical data management for energy consumption analysis by integrating Django, Raspberry Pi, Supabase and MQTT.

2. Methods

2.1. Research object and equipment

The research object is a web-based smart home prototype designed to support real-time environmental monitoring, remote device control, and historical data management. The prototype simulates a residential environment by integrating multiple sensing devices, actuators, and cloud-based services within an Internet of Things (IoT) architecture. The hardware platform consists of a Raspberry Pi 4 as the central controller, environmental sensors including the AHT20, MQ-2, flame sensor, LDR sensor, reed switch, and RFID MFRC522 module, as well as actuator devices comprising a servo door lock, a lamp, and a DC fan. The software platform comprises the Django framework for web application development, Supabase with PostgreSQL as the cloud database, and the MQTT protocol for data communication within the system.

2.2. System architecture

The proposed smart home system adopts an Internet of Things (IoT) architecture that integrates sensing devices, a central controller, cloud-based data storage, and a web application to enable real-time monitoring and remote device control. Raspberry Pi functions as the central controller and communicates with the cloud platform through the MQTT protocol, receiving data from multiple sensors and controlling connected actuators based on user commands. The collected sensor data are transmitted to the Django application and synchronized with the Supabase cloud database, which is built on PostgreSQL. The web application retrieves the stored data from the database to provide users with real-time monitoring and remote device control through an internet connection. The system consists of six sensing components, namely the AHT20 sensor for measuring temperature and humidity, the MQ-2 sensor for gas detection, the flame sensor for fire detection, the reed switch for door status detection, the RFID reader for access authentication, and the LDR sensor for light intensity detection. The actuator components include a servo door lock, a lamp, and a fan, which are controlled by Raspberry Pi according to the commands received from the web application.

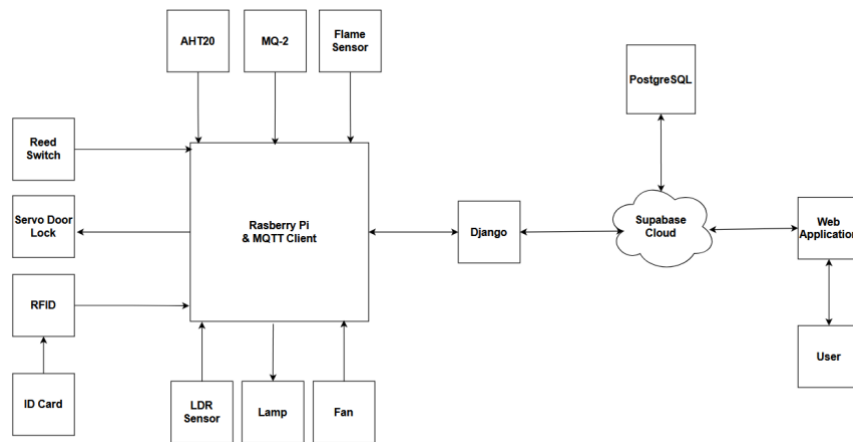


Fig. 1: Architecture of the proposed IoT-based smart home system

2.3. Communication and control mechanism

The proposed smart home system employs a database-driven control mechanism to establish bidirectional communication between the web application and IoT devices. The communication architecture integrates Django, Supabase (PostgreSQL), Raspberry Pi, and the MQTT protocol to synchronize sensor data and actuator commands. User commands issued through the web dashboard are processed by Django and stored in the cloud database before being executed by Raspberry Pi. Conversely, Raspberry Pi continuously collects data from environmental sensors and transmits the readings to the cloud database through MQTT. The stored data are retrieved by the Django application and displayed on the web dashboard, enabling users to monitor environmental conditions and device status remotely. Historical data, including sensor readings and actuator states, are maintained in the database to support further analysis of system performance and energy consumption.

2.4. Energy consumption estimation

The proposed smart home system records the operating history of actuator devices, including the lamp, DC fan, and servo door lock. The recorded operating duration of each actuator was exported from the system database and processed using Google Colab to estimate daily energy consumption.

The energy consumption of each actuator was estimated using Equation (1), where the rated power of the electrical device was multiplied by its operating duration. The rated power values were obtained from the specifications of the actuator devices, and the resulting energy values were aggregated to determine the total daily energy consumption.

$$E = \frac{P \times t}{1000} \quad (1)$$

where E is the energy consumption (kWh), P is the rated power of the electrical device (W), and t is the operating duration (h).

2.5. Development process

The proposed smart home system was developed using the Agile software development methodology with the Scrum framework [9]. Agile was selected because it supports iterative development, continuous evaluation, and adaptability to changing requirements, making it suitable for Internet of Things (IoT) applications that involve the integration of hardware and software components [9]. Scrum divides the development process into a series of short iterations (sprints), where each sprint produces a functional increment of the system through planning, implementation, testing, and evaluation activities. In this research, the development process began with requirement analysis to identify system objectives and functional requirements, followed by system design to define the hardware architecture, software architecture, and communication mechanism. The implementation phase was then carried out iteratively through multiple sprints. Each sprint consisted of sprint planning, incremental development, sprint testing, and sprint review and evaluation. The evaluation results determined whether additional iterations were required before proceeding to the final system implementation. This iterative approach enabled continuous improvement throughout the development process while ensuring that each system component functioned correctly before integration into the complete smart home system.

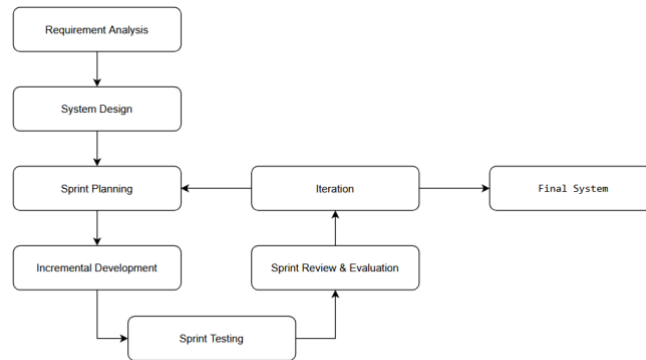


Fig. 2: Agile Scrum development process adopted in this research

3. Results and discussion

3.1. System implementation

The proposed smart home system was successfully implemented by integrating environmental sensors, actuator devices, Raspberry Pi, the Django web application, Supabase (PostgreSQL), and the MQTT communication protocol into a single IoT platform. The implementation followed the architecture described in Section 2, where Raspberry Pi functioned as the central controller to collect sensor readings, execute actuator commands, and synchronize sensor data and actuator states with the cloud database through MQTT communication.



Fig. 3: Implemented smart home prototype

As shown in Fig. 3, the developed prototype consists of multiple rooms equipped with environmental sensors and actuator devices. The AHT20, MQ-2, flame sensor, LDR sensor, reed switch, and RFID module are installed to monitor environmental conditions and user access, while the lamp, DC fan, and servo door lock provide automated operation and remote control capabilities. The prototype served as the experimental environment for evaluating the proposed smart home system.

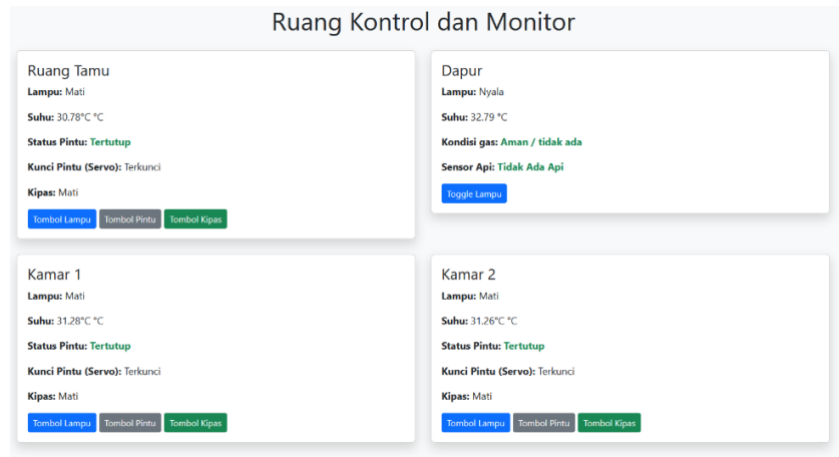


Fig. 4: Web dashboard for real-time monitoring and remote device control

Figure 4 presents the web dashboard developed for real-time monitoring and remote device control. The dashboard displays real-time environmental information, including room temperature, gas condition, fire detection status, door status, lighting status, and fan operation. Users can remotely control connected devices through the web interface, while sensor readings and actuator states are continuously synchronized with the cloud database to provide near real-time system updates.

3.2. Performance evaluation of sensors and actuators

3.2.1. Environmental sensor performance

After integration in the proposed smart home system, the performance of the environmental sensors such as AHT20, MQ-2, flame sensor, and LDR sensor was evaluated. Each sensor was tested under representative operating conditions to verify its functionality and to make sure that the acquired sensor data could be sent to the cloud database and displayed on the web dashboard in near real time. The assessment included temperature measurement, gas detection, flame detection, and light intensity detection. The testing scenarios and the results are summarized in Table 1.

Table 1: Summary of environmental sensor performance

Sensor	Function	Test scenario	Result
AHT20	Temperature monitoring	Compared with an analog thermometer in four rooms	Successful
MQ-2	Gas detection	Gas exposure using a lighter	Gas detected successfully
Flame Sensor	Fire detection	Flame exposure using a lighter	Fire detected successfully
LDR	Light intensity detection	Bright and dark conditions	Successful automatic lighting control



Fig. 5: Experimental setup for evaluating the AHT20, MQ-2, flame, and LDR sensors

Figure 5 presents the evaluation setup of the environmental sensors integrated into the proposed smart home system. The AHT20 sensor was tested against an analog thermometer for temperature measurement accuracy. The MQ-2 sensor, flame sensor and LDR sensor were tested against representative operating conditions for gas detection, flame detection and light intensity detection, respectively. The AHT20 sensor showed only slight differences compared to the analog thermometer, meaning that the accuracy was good enough for indoor environmental monitoring. Also, the obtained data from the sensors was synchronized with the cloud database successfully and visualized in the web dashboard, thus enabling continuous monitoring of the room conditions.

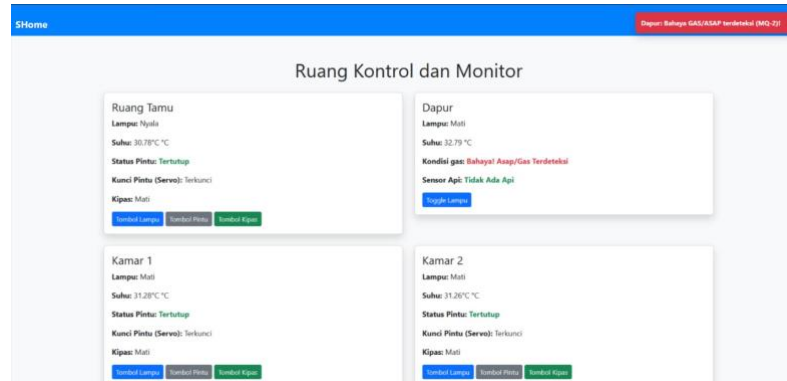


Fig. 6: Hazard detection and warning notifications on the web dashboard

Figure 6 shows the hazard detection and warning notifications on the web dashboard. The MQ-2 gas sensor and flame sensor were tested by exposing them to gas and flame sources in a controlled manner. When a hazardous condition was detected, the respective gas and fire status was updated on the web dashboard in near real time and the sensor data were synchronized with the cloud database simultaneously. It also generated warning alerts to warn users of potential hazards. The results show that the proposed system provides reliable real-time hazard detection and timely notification to the user for smart home safety.

3.2.2. Access control and actuator performance

The access control and actuator subsystem has been tested after the integration of RFID authentication module, reed switch, servo door lock, lamp, and DC fan to the proposed smart home system. The evaluation considered each component's ability to perform its intended function, such as user authentication, door status detection, remote device control, and automatic actuator operation. Table 2 summarizes the evaluation results, showing that all components worked successfully within the defined testing scenarios.

Table 2: Summary of access control and actuator performance

Component	Function	Test scenario	Result
RFID MFRC522	User authentication	RFID card verification	Authentication successful
Reed switch	Door status detection	Door opened and closed	Door status detected successfully
Servo door lock	Door locking and unlocking	RFID authentication and remote control via the web dashboard	Operated successfully
Lamp	Remote and automatic lighting control	Web dashboard and LDR-based control	Operated successfully
DC fan	Remote and automatic ventilation control	Web dashboard and temperature-based control	Operated successfully

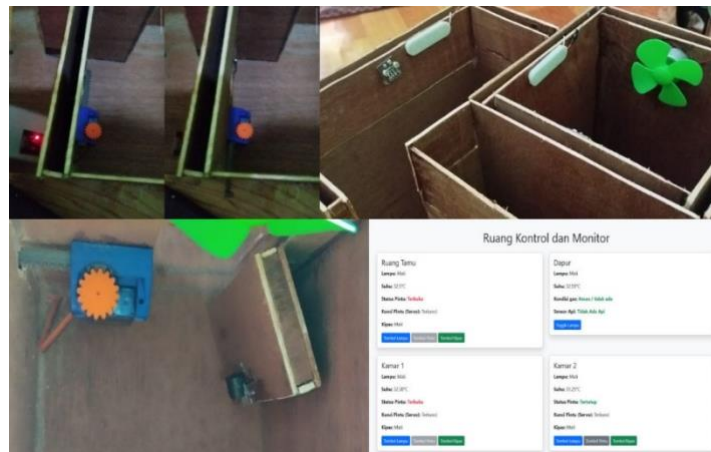


Fig. 7: Experimental evaluation of RFID, reed switch, and actuator operation

The evaluation results for access control and the actuator subsystem are shown in Figure 7. The RFID authentication mechanism was tested by scanning the registered RFID cards to allow Raspberry Pi to send the corresponding command to the servo door lock to grant access to the prototype. The door opening and closing conditions were successfully sensed by the reed switch; and the updated door status was synchronized with the cloud database and displayed on the web dashboard in real time.

The actuator subsystem had remote and automatic operating modes. The web dashboard allowed users to remotely control the lamp, DC fan and door lock and monitor the current status of each device in real time. In automatic mode, the light system was controlled according to the light intensity received by the LDR sensor, and the DC fan was automatically switched on according to the room temperature. The experimental results demonstrated correct operation of the access control mechanism and all actuator devices under both remote commands and sensor-based control logic, indicating reliable smart home automation and real-time device control.

3.3. Historical data management and energy consumption analysis

The proposed smart home system automatically recorded sensor readings and actuator state changes in the PostgreSQL database through the Supabase cloud platform. The historical data management feature allows users to review previous system activities and serves as the primary data source for evaluating system performance and estimating household energy consumption. Figure 8 presents the historical data interface developed for the proposed system.

Ruangan	Alat	Mulai	Selesai	Cap
Ruang Tamu	Lampu	30 Sep 2025 02:35:49		Reset
Ruang Tamu	Pinza	30 Sep 2025 02:35:15		Reset
Ruang Tamu	Pinza	30 Sep 2025 02:35:13		Reset
Ruang Tamu	Suhu	30.83°C → 30.87°C	26 Sep 2025 07:36:17	Reset
Ruang Tamu	Suhu	33.25 → 30.83°C	26 Sep 2025 07:36:11	Reset
Kamar 1	Suhu	31.43°C → 31.33°C	26 Sep 2025 07:35:30	Reset
Kamar 1	Suhu	31.62°C → 31.43°C	26 Sep 2025 07:35:25	Reset
Kamar 1	Suhu	31.26°C → 31.62°C	26 Sep 2025 07:35:19	Reset
Kamar 1	Suhu	31.27°C → 31.28°C	26 Sep 2025 07:35:13	Reset
Kamar 1	Suhu	31.29°C → 31.27°C	26 Sep 2025 07:35:07	Reset
Ruang Tamu	Reed_Switch	True → False	26 Sep 2025 07:35:01	Reset
Kamar 1	Suhu	31.33°C → 31.29°C	26 Sep 2025 07:34:55	Reset
Ruang Tamu	Reed_Switch	False → True	26 Sep 2025 07:34:55	Reset
Kamar 1	Suhu	31.32°C → 31.33°C	26 Sep 2025 07:34:49	Reset
Kamar 1	Suhu	31.33°C → 31.32°C	26 Sep 2025 07:34:43	Reset

Fig. 8: Historical data management interface

The historical data interface provides filtering options based on room, device, and date range, allowing users to retrieve specific records efficiently. Each record contains the room name, device name, state transition, and timestamp, enabling comprehensive monitoring of system activities. In addition to supporting historical monitoring, the stored data can be exported for further analysis of device usage patterns and energy consumption. The exported historical data were processed using Google Colab to estimate the daily energy consumption of the connected electrical devices based on the energy consumption estimation method described in Section 2.4. The calculated energy values were aggregated to obtain the total daily energy consumption and subsequently visualized using Google Colab, as presented in Fig. 9.

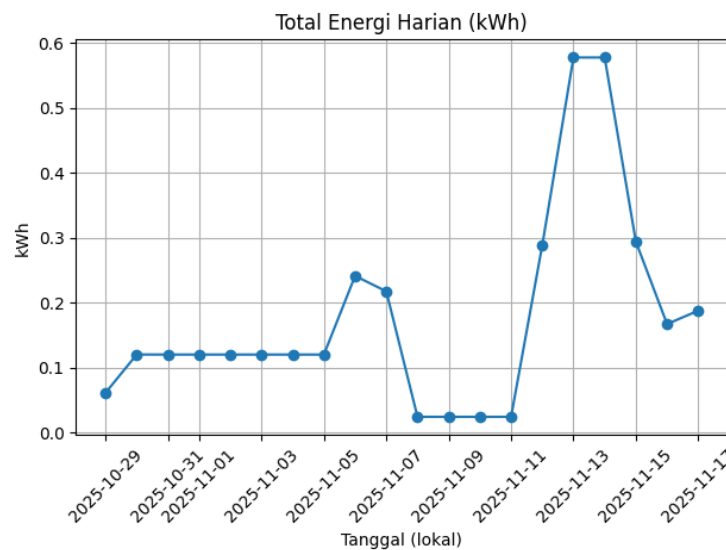


Fig. 9: Daily energy consumption analysis generated using Google Colab

The daily energy consumption profile shows variations in energy usage according to the operating duration of the connected electrical devices. Days with longer operating durations resulted in higher estimated energy consumption, whereas shorter operating durations produced lower energy usage. The observed fluctuations indicate that the daily energy consumption was primarily influenced by the

operating duration of the actuator devices recorded by the system. These results demonstrate that the proposed smart home system not only provides real-time monitoring and remote device control but also supports historical data analysis and energy consumption estimation, enabling users to better understand household energy usage and identify opportunities for improving energy efficiency.

3.4. Discussion

The experimental results demonstrate that the proposed smart home system successfully integrates environmental monitoring, hazard detection, access control, remote device control, historical data management, and energy consumption analysis within a single web-based platform. The integration of Raspberry Pi, Django, Supabase (PostgreSQL), and the MQTT protocol enabled reliable communication among sensors, actuators, and the web application, allowing users to monitor environmental conditions, authenticate user access, and control connected devices through a standard web browser.

Previous studies have demonstrated IoT-based smart home systems for real-time monitoring, remote device control, RFID-based access control, and Raspberry Pi-based home automation [2]–[6]. Other studies have focused on specific applications such as fire detection and gas leakage monitoring [8], [9]. In contrast, the proposed system integrates these functionalities with structured historical data management and daily energy consumption estimation within a single web-based platform. By continuously recording actuator operating history in the cloud database, the proposed system enables users to analyze device usage patterns using Google Colab, extending the system functionality beyond conventional monitoring and control.

Although the proposed architecture has been successfully implemented and evaluated on a smart home prototype, the current implementation has several limitations. The experimental evaluation was performed in a prototype environment instead of a real residential building, and the communication mechanism uses the standard MQTT protocol without any additional security mechanisms such as encryption. Moreover, the energy consumption analysis is based on the operating time and rated power of actuator devices instead of direct electrical power measurement, which may lead to estimation errors under different working conditions.

Future research should validate the proposed architecture in real residential environments, improve communication security by means of MQTT encryption and incorporate intelligent energy management techniques to predict energy consumption and optimize the household electricity usage.

4. Conclusion

This study successfully developed a web-based smart home system by integrating Raspberry Pi, Django, Supabase (PostgreSQL), and the MQTT communication protocol into a unified Internet of Things (IoT) platform. The developed system enables real-time environmental monitoring, remote device control, historical data management, and energy consumption estimation through a web-based interface. Experimental results demonstrated that the proposed system operated reliably in monitoring environmental conditions, controlling connected actuator devices, and managing user access through RFID authentication. Furthermore, the historical data recorded by the system were successfully utilized to estimate daily energy consumption, providing additional insights into device usage patterns and household energy utilization. Overall, the proposed system provides an integrated solution for smart home monitoring and automation by combining cloud-based data storage, web technologies, and IoT communication. Future work should focus on validating the proposed architecture in real residential environments, enhancing communication security through MQTT encryption, and extending the system with intelligent energy management capabilities.

References

- [1] S. P. Lesmana, A. P. S. B. Merah, D. Hermawati, and N. Puspitasari, "Dampak Implementasi IoT pada Sistem Smart Home untuk Efisiensi Energi dan Keamanan di Kota Berkembang," in *Proc. Seminar Nasional AMIKOM Surakarta (SEMNAS)*, 2024.
- [2] N. R. Ahsy, A. Bhawiyuga, and D. P. Kartikasari, "Implementasi Sistem Kontrol dan Monitoring Smart Home Menggunakan Integrasi Protokol Websocket dan MQTT," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 4, pp. 3709–3718, 2019.
- [3] A. B. Aldiansyah, M. Hakimah, and T. Tukadi, "Sistem Monitoring dan Kontrol Rumah Berbasis Internet of Things (IoT)," in *Seminar Nasional Sains dan Teknologi Terapan X*, Institut Teknologi Adhi Tama Surabaya, 2022.
- [4] K. U. Ariawan, "Penerapan IoT untuk Sistem Kendali Jarak Jauh Peralatan Listrik Rumah Tangga Berbasis Raspberry Pi," *Jurnal Nasional Pendidikan Teknik Informatika*, vol. 9, no. 3, pp. 292–293, 2020.
- [5] A. R. Mukti, Widyanto, M. Chairul, and E. R. Kasih, "Perancangan Smart Home Menggunakan Konsep Internet of Things (IoT) Berbasis Microcontroller," *Jurnal JUPITER*, vol. 14, no. 2, pp. 516–522, 2022.
- [6] I. Ibrahim and Y. Saragih, "Implementasi Aplikasi Styins Home pada Smart Home Security Menggunakan Real-Time Database Firebase," *Jurnal Ecotipe*, vol. 7, no. 2, 2020.
- [7] W. Kuncoro, J. Maulindar, and R. P. Indah, "Monitoring Peringatan Dini Kebakaran Pada Sistem Smart Home Menggunakan NodeMCU Berbasis IoT," *Generation Journal*, vol. 7, no. 2, pp. 105–115, 2023.
- [8] N. Nurhapsari, S. Paembonan, and R. Suppa, "Rancang Bangun Sistem Pendeteksi Kebocoran Gas Berbasis IoT," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 13, no. 1, 2025.
- [9] N. Hikmah, A. Suradika, and R. A. A. Gunadi, "Metode Agile untuk Meningkatkan Kreativitas Guru Melalui Berbagi Pengetahuan (Knowledge Sharing)," *Jurnal Instruksional*, vol. 3, no. 1, pp. 30–39, 2021.
- [10] L. D. Samsumar, Z. Zaenudin, A. Akbar, E. Suryadi, and B. A. Hidayatullah, "Sistem Monitoring dan Kontrol Rumah Pintar Berbasis Internet of Things untuk Peningkatan Efisiensi Energi," *Jurnal Teknik Informatika dan Teknologi Informasi (JUTITI)*, vol. 3, no. 2, pp. 33–50, 2023.