Integration of Multiple Sensors and Actuators in Smart Home Automation System Based on IoT Concept

Ni Luh Putu Wulandarini[1], I Gusti Ngurah Agung Ananda Putra[2], Made Liandana[3], Gede Herdian Setiawan[4], and Roy Rudolf Huizen[5]

[1,2,3,4,5]Institut Teknologi dan Bisnis (ITB) STIKOM Bali, Bali, Indonesia
niluhputuwulandarini@gmail.com

Abstract. Modern times have created various devices to help human lives in homes, offices, and other buildings. However, most of these devices still rely on manual or conventional operation which creates some inconveniences, including limitations in monitoring, remote control, safety, and inefficient energy usage. Smart home is an IoT-based automation that allows owners to control, and monitor electrical devices and home security systems automatically as desired. However, devices in a smart home often operate with heterogeneous communication protocols, resulting in integration challenges. Adding new modules to an existing system require complex initial configuration to be properly integrated. For this reason, an integration and coordination system in a smart home is needed to ease the addition of new modules to the existing system without disrupting the performance of old modules, monitors, and control devices in the smart home. This research was carried out by applying the waterfall software development method to achieve the integration of new modules on smart home devices without the need to carry out initial configuration manually. This system uses a website as an interface to monitor and control devices. Using ESP8266 as a microcontroller, this system collects data from PIR, IR Flame, MQ-2, DHT22, LDR, and RFID sensors. In this way, a smart home control and monitoring system has been produced that uses multi-sensors and actuators with dynamic module configurations.

Keywords: Internet of Things, Integration, Smart Home

1 Introduction

Modern times have created various devices to help human lives in homes, offices, and other buildings. Examples of commonly used electronic devices include: lamps to provide lighting in each room, fans for air conditioning, CCTV for home security, and speakers for listening to songs. In general, these devices are controlled manually or conventionally, for example lights using single-pole switches (Adedoyin et al., 2020). The manual use of these devices can present some obstacles, including difficulties in monitoring, limitations of remote control, lack of security, and wastage of electricity For example, users forget to turn off lights after use or when traveling, which can result in unnecessary expense of electricity and heightened electricity bills. The

A. A. N. G. Sapteka et al. (eds.), *Proceedings of the International Conference on Sustainable Green Tourism Applied Science - Engineering Applied Science 2024 (ICoSTAS-EAS 2024)*, Advances in Engineering Research 249,
https://doi.org/10.2991/978-94-6463-587-4_76

negative impacts that can be caused by increased electricity bills are household economic instability and sacrifices on other needs such as education and health (Rajarajeswari et.al., 2021).

A smart home system is the solution for overcoming the manual use of electronic devices (Prasetyo et. al., 2019). The smart home is one of the IoT-based automation that allows owners to control, and monitor electrical devices, and home security systems automatically as desired (Abdulraheem et al., 2020). However, many smart home devices currently work differently due to the heterogeneity of their communication protocols. Heterogeneity in the context of IoT refers to the diversity of hardware, software, communication protocols, and standards used by different IoT devices. Based on interviews with laypeople about smart home technology, the use of electronic devices with manual controls such as lights that must be turned off physically can cause anxiety, especially when away from home, and cause loss of time and inconvenience. Meanwhile, users who understand smart home technology experience obstacles in controlling and monitoring due to differences in protocols on devices, such as Wi-Fi, Zigbee, or Z-Wave on smart lights and door locks. Integration is usually smoother between devices from the same manufacturer; however, the integration of different brands often causes problems. In addition, the existing smart home system must be configured when adding new modules to ensure proper integration without disrupting the performance of old modules, as well as monitoring and controlling smart home devices (Stolojescu-Crisan et.al., 2021).

In this regard, a study conducted in 2020 by Uma Pujari, and friends who succeeded in this research system produces an android-based multi-sensor integration using the ESP32 module with Internet connectivity that allows remote device control (Pujaria et.al., 2020). Another study conducted by Ghozali et al. (2022), produced the results of a website-based AC control and monitoring system using an ESP8266 (Ghozali et al., 2022). Apart from that, in 2020 Ayu Anki and Deswarto conducted another study that produced Smart Home system modules that were integrated and configured manually or hardcoded (Anki & Deswarto, 2020). Further research conducted by Reski Damayanti et.al. in 2020 produced a smart home application that can control and monitor remotely with configurations implemented via hardcode (Anki & Deswarto, 2020).

Many studies on smart homes have been conducted previously but from these studies, the integration of new modules to be added to smart home devices has not been the focus of resolution. Therefore, the study that will be conducted will focus on integrating new modules into smart home devices more easily and effectively without the need to hardcode the initial configuration. It is hoped that this system will make it easier and more cost-effective to add new modules to smart home control and monitoring devices.

2 Methodology

The Waterfall Model, the first process model introduced, is also known as the linear sequential lifecycle model. It is straightforward to use, with each phase needing to be completed before the next begins, and phases do not overlap. This model represents the software development process in a linear sequence, ensuring that each phase starts only after the previous one is finished. It is widely utilized in Software Engineering to promote project success. The Waterfall approach divides the development process into distinct phases, where the output of one phase becomes the input for the next. Below is a diagram of the phases of the Waterfall Model (Senarath, 2021).

2.1 Analysis

Often called Software Requirements Specification (SRS), this document provides a comprehensive description of the behavior of the software to be developed. It includes system and business analyses to determine functional requirements through use cases and non-functional requirements such as performance, scalability, and performance. Functional requirements include the purpose, scope, functions, software attributes, and interface specifications, whereas non-functional requirements focus on design and operational criteria.

2.2 Design

This is the process of planning and solving problems related to software solutions. This process involves software developers and designers in determining the solution plan that includes algorithm design, software architecture design, basic data conceptual schema, logistics diagram design, concept design, graphical user interface design, and data structure definition.

2.3 Implementation

This term describes the process of translating business requirements and design specifications into an executable program, database, website, or software component through programming and implementation. In this phase, code is written and compiled into a working application, and databases and text files are created. In other words, it is the process of turning all the requirements and design plans into a final product that is ready to use.

2.4 Testing

Known as verification and validation, these processes ensure that the software meets the intended initial requirements and functions. Verification checks whether the output of the development phase meets the specific conditions of the phase, whereas validation assesses whether the software meets the requirements during or after development. In the testing phase is where bugs and system issues are identified and resolved.

2.5 Maintenance

Maintenance involves updating software after deployment to refine output, fix errors, and enhance performance and quality. It also includes adapting the software to its environment, addressing new user needs, and improving reliability.

3 Result and Discussion

3.1 System Analysis

Analysis of the working system. Analysis of the working system in IoT-based smart home automation allows users to control and monitor home electronic devices through the website. This system uses ESP8266 (NodeMCU), an MQTT broker, and a 2-channel relay for device connection from a near or far distance. Various sensors are used, such as a fire sensor (IR Flame sensor), motion sensor (PIR HCSR501), gas sensor (MQ-2), temperature and humidity sensor (DHT22), light sensor (LDR), and RFID for access. The system provides on/off switches to activate or deactivate lights, fans, doors, and motion detection, as well as alarm features for fire, gas leak, and movement detection. In addition, there is a feature to control the light intensity of the lights.

The Functional Requirement Analysis. The Functional Requirement Analysis covers the services that the system should provide, such as providing up-to-date temperature, humidity, light intensity, gas, fire, and movement detection information. In addition, the system must be able to control the on-off of lights, disable motion sensors, control the opening and closing of gates, and perform configuration and integration of new modules.

Non-Functional Requirement Analysis. Non-Functional Requirement Analysis involves the devices required to support the system. For hardware, a computer or laptop, miniature smart home, and various sensors such as fire sensor (IR Flame sensor), motion sensor (PIR HCSR501), gas sensor (MQ-2), temperature and humidity sensor (DHT22), light sensor (LDR), and RFID are required. As for software, the system uses Windows 10, XAMPP for database and web server (PHP), Mosquitto as an MQTT broker, NodeJs as a front-end web server, Arduino IDE for hardware module development, and VSCode for website creation.

3.2 System Design

Use Case Diagram. A use case diagram is used to provide an overview of the behavior or functions that a system can perform. This diagram illustrates the interactions between actors, such as users or other systems, and the various functions provided by the system. It helps clarify how different users will engage with the system and what specific actions they can take. In this system, there are six main functions available (see Figure 1).

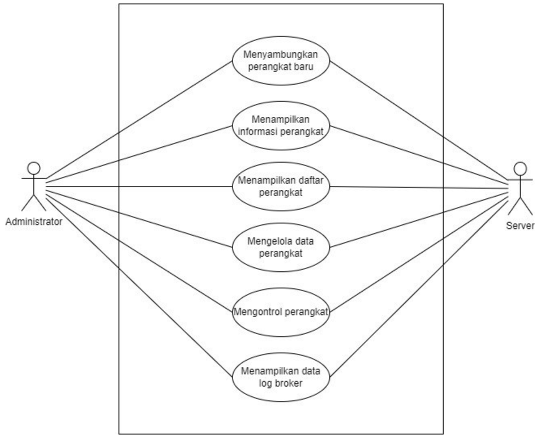

Figure 1. Use case diagram of smart home integration (in Indonesia language)

Class Diagram. In this system, the class diagram is used to illustrate the structure and relationships within the smart home integration system. The diagram displays various classes, attributes, methods, and how these classes are interconnected. The class diagram helps developers understand how the system components interact, aiding in the effective design of an integrated system (see Figure 2).

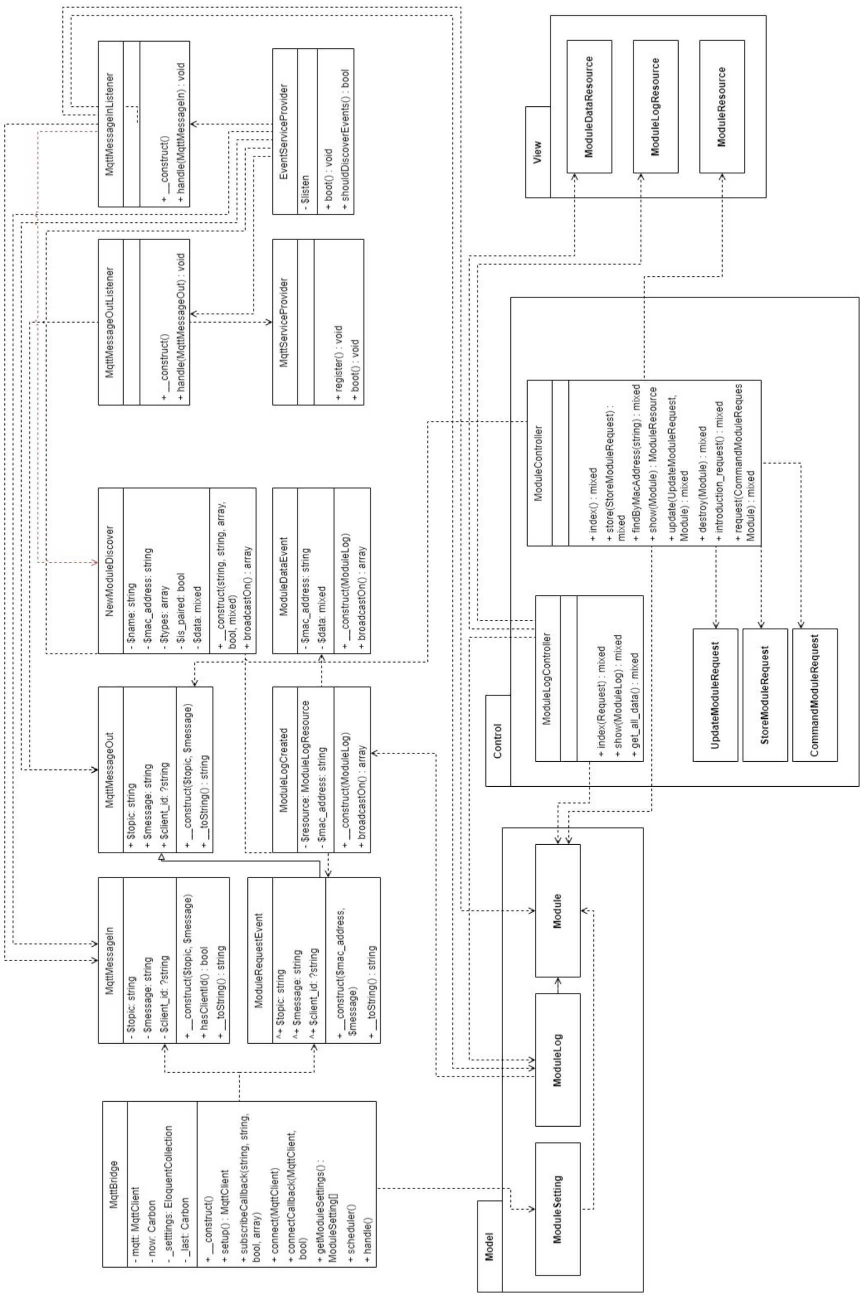

Figure 2. Class diagram of smart home integration

3.3 Implementation

Hardware implementation. The mechanical design results of the smart home integration system can be seen in the smart home miniature. It includes six sensors placed in various rooms: a gas sensor in the kitchen to detect gas leaks, a temperature and humidity sensor in Room 1, a fire sensor in the living room to detect fire, a light sensor in Room 2 to detect light, a motion sensor in the guest room to detect the presence, and an RFID sensor at the gate for opening or closing. The fan is the output from the temperature and humidity sensor, while the light is the output from the light sensor. Miniature household items are used to support the smart home miniature display (see Figure 3).

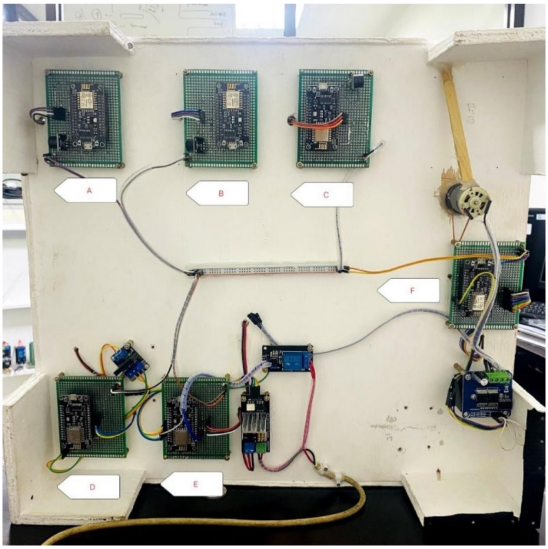

Figure 3. Hardware implementation

The components in each module of the smart home integration system are as follows: Gas Sensor Module includes an MQ-2 sensor, ESP8266 for control, PCB for electrical connections, a buzzer as an alarm, and a pin header for external connections, Fire Sensor Module consists of an IR Frame sensor, ESP8266 for control, PCB, buzzer, and pin header, Motion Sensor Module comprises a motion sensor, ESP8266, PCB, buzzer, and pin header, Temperature, and Humidity Sensor Module features a DHT22 sensor, ESP8266, PCB, relay, fan, battery, and pin header, Light Sensor Module contains an LDR sensor, ESP8266, PCB, relay, dimmer, plug, and pin header, RFID Module includes RFID, ESP8266, PCB, motor driver, battery, dynamo, and pin header.

Data collected by the sensors is transmitted via the MQTT protocol to the backend, where it is processed, stored, and analyzed. The results are then displayed on the front end in the form of graphs, information, and control buttons.

Software Implementation. To access the system, users must first log in. The login process begins with the user entering their username and password, then pressing the login button. The system will verify the entered information against the data stored in the database. If the information matches, the user will be granted access to the smart home integration system.

Dashboard Menu. The main page of the smart home integration system is shown in the image above. This page displays information about the integrated modules and provides access to additional menu options. The profile popup includes a logout function to exit the system. Once logged out, users must log in again to access the smart home integration system (see Figure 4).

Figure 4. Dashboard menu (in Indonesia language)

Module Menu. The module page displays the modules integrated into the system. The "Details" button enables users to view detailed information about each module. Additionally, the "Connect Module" button facilitates the integration of new modules into the system. This setup allows for a comprehensive view and management of the system's components (see Figure 5).

Figure 5. Module menu (in Indonesia language)

Log Broker Menu. The log broker page provides a detailed view of data transmitted by each integrated module over a specified period. It captures and records module activity in real-time, offering up-to-date logs and insights into system operations. This feature allows users to monitor and review the performance and interactions of each module effectively (see Figure 6).

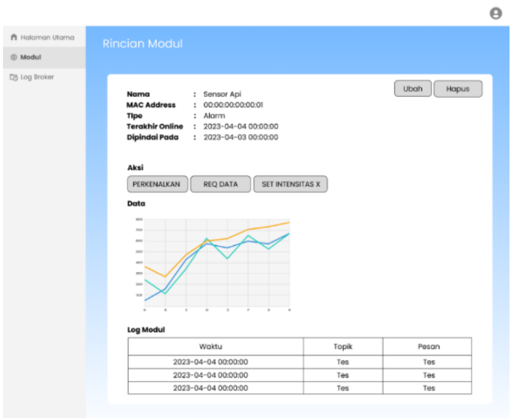

Figure 6. Log broker menu (in Indonesia language)

3.4 Testing

After the system is implemented and the devices are properly integrated, various tests are conducted to ensure the system and equipment operate effectively. The testing process includes four main functions: system functionality testing, response time testing, and user testing.

Blackbox Testing. In this testing phase, Blackbox Testing is used, specifically Scenario-based Testing. This method involves evaluating various scenarios that may occur during smart home system use, without examining the internal code structure. The testing scenarios focus on system responses to different situations, including login processes, device settings, alarm functions, and interactions with integrated modules. Blackbox Testing for this system is performed by potential users. The results show that all functions of the smart home integration system operate effectively and as intended.

Response Time Testing. Response time testing is conducted using a stopwatch to precisely measure the time it takes for the device to respond, with tests executed on a local host server. This approach ensures accurate measurement of how quickly the system processes and reacts to inputs. The results reveal that the device delivers a swift response, meeting the performance expectations and demonstrating its efficiency in real-world scenarios. This effective response time is crucial for maintaining smooth operation and user satisfaction in the smart home system. The average response times of the modules were 0.56 second.

User Testing. User testing of the system assesses user satisfaction and experience with the product, evaluating how well its functionality meets user needs. The results showed that the smart home system met or exceeded the expectations for each tested feature. The user interface (UI/UX) is both aesthetically pleasing and clear, with strong security and privacy measures. Adding new modules is intuitive, smart home controls are easy to understand, and system responsiveness is highly efficient. The creative design of the house miniature enhances the user experience. In conclusion, the waterfall development method successfully produced an efficient and satisfying smart home solution.

4 Conclusion

Based on the implementation results of multi-sensor and actuator integration in a smart home automation system based on the IoT concept, the author can provide conclusions namely; ease of addition of device to the system has been implemented, sensors and actuators that have been tested can function well according to the functionality of the system, black box testing has results that are following the functionality of the system and integration functions the dynamic module has been successfully implemented and a system has been produced smart home control and monitoring using multiple sensors and actuators with dynamic module configuration.

References

Abdulraheem, A. S., Salih, A. A., Abdulla, A. I., Sadeeq, M. A. M., Salim, N. O. M., Abdullah, H., Khalifa, F. M., & Saeed, R. A. (2020). Home automation system based on IoT. *Technology Reports of Kansai University*, *62*(05), 2453–2464.

Adedoyin, M. A., Shoewu, O. O., Adenowo, A. A., Yussuff, A. I. O., & Senapon, M. F. (2020). Development of a smart IoT-based home automation system. *Engineering and Technology Research Journal*, *5*(2), 25–37. https://doi.org/10.47545/etrj.2020.5.2.062

Anki, A., & Deswarto. (2020). *Integrasi perangkat-perangkat pada smart*.

Ghozali, T., Ilena, G., Mulyadi, M., & Wijayanti, L. (2022). Web-based air conditioner monitoring and control. *Saudi Journal of Engineering and Technology*, *7*(8), 461–471. https://doi.org/10.36348/sjet.2022.v07i08.006.

Prasetyo, N. A., Prabawati, A. G., & Suyoto. (2019). Smart home: Power electric monitoring and control in Indonesia. *International Journal of Interactive Mobile Technologies*, *13*(3), 143–151. https://doi.org/10.3991/ijim.v13i03.10070.

Pujaria, U., Patil, P., Bahadure, N., & Asnodkar, M. (2020). Internet of things based integrated smart home automation system. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3645458.

Rajarajeswari, S., Rani, S. U., Kataria, A., & Dutta, S. (2021). Home Automation through smart lighting, smart security and other appliances. *ITM Web of Conferences*, *37*, 01024. https://doi.org/10.1051/itmconf/20213701024

Senarath, U. S. (2021). Waterfall methodology, prototyping and agile development. *Tech. Rep.*, *June*, 1–16.

Stolojescu-Crisan, C., Crisan, C., & Butunoi, B. (2021). *Sensors-21-03784-V2.Pdf*. 1–23.