Modular Design for a Flexible IoT Monitoring and Control Application using Flutter

Fajri Profesio Putra[1,a)], Muhammad Asep Subandri[1,b)]

[1]*Department of Informatics, Bengkalis State Polytechnic, Bengkalis, Indonesia*

[a)]Corresponding author: fajri@polbeng.ac.id
[b)]msubandri@polbeng.ac.id

Abstract. The Internet of Things (IoT) has emerged as a transformative technology that enables smart environments through extensive sensor networks and connected devices. However, the diversity and heterogeneity of IoT devices pose significant challenges in developing versatile applications capable of monitoring and controlling a broad spectrum of sensors and actuators. This paper proposes a novel modular design for a flexible IoT monitoring and control application using Flutter, a popular framework for developing cross-platform solutions. Our design leverages Flutter's robust UI capabilities and its efficient communication with backend services to offer a scalable and customizable interface that can adapt to various IoT deployments, from domestic to industrial settings. The modular architecture allows users to specify and configure the application to dynamically include any combination of devices and functionalities, catering to specific needs without additional programming.

Keywords: *Internet of Things (IoT), Modular Design, Flutter, Cross-platform, Sensor Networks*

INTRODUCTION

In the rapidly evolving digital era, the Internet of Things (IoT) technology has become an integral part of daily life [1]. IoT enables various physical devices to connect and communicate over the internet, creating a network that allows for automatic data collection and exchange. This technology has brought significant changes across various sectors, such as industry, healthcare, smart homes, and agriculture, where real-time data management and remote-control capabilities are crucial [2].

As IoT adoption expands, the need for flexible and customizable monitoring and control applications to meet diverse user requirements is growing as well [3]. One of the primary challenges IoT application developers face is creating solutions that can adapt to various devices and scenarios without requiring significant code adjustments. A flexible IoT application must be designed to perform optimally across diverse environments, from household to more complex industrial settings [4], [5].

Modular design offers a relevant solution to this challenge. With modular design, each application component can be configured independently, allowing features to be added or removed as needed to meet user requirements or environmental conditions [6], [7]. This approach enables quick adaptation and eases maintenance for various use cases.

Flutter, an open-source framework developed by Google, provides a relevant solution in cross-platform application development with a consistent interface across multiple devices (Android, iOS, web, and desktop) [8]. Flutter supports modular component development, enabling an IoT application design with high flexibility and easy expansion. Additionally, Flutter supports common IoT communication protocols such as MQTT (Message Queuing Telemetry Transport) and HTTP, facilitating seamless integration with existing IoT systems.

This study aims to design and evaluate a modular architecture based on Flutter that allows IoT monitoring and control applications to dynamically adapt to various scenarios, from household settings to industrial applications. With

a modular design, this application offers ease in configuration adjustments, cross-platform compatibility, and strong scalability to meet diverse user needs. This study hopes to significantly contribute to the development of more flexible and user-centered IoT applications, while also providing a solid foundation for future development.

METHODS

The proposed IoT application architecture in this study is designed with a modular approach to ensure optimal flexibility and scalability across different IoT implementations. This modular approach is designed to ensure each architectural component functions independently, thus enabling quick adaptation to various user needs, both in small-scale (household) and large-scale (industrial) settings. The architecture structure is divided into several key layers, each with a specific role and designed as an independent component to facilitate further customization and development.

User Interface Layer

a. Modular Widgets
   Flutter's widget-based UI structure is highly suited to a modular approach. Each widget in this application is designed as an independent functional component (e.g., sensor, actuator, data graph) that can be configured according to user needs. This design enables UI components to be modified or updated without affecting the entire user interface.
b. Customizable Dashboard
   The application allows users to create a personalized dashboard by adding or removing widgets as needed. For example, users can display IoT data in the form of charts, tables, or status indicators depending on the IoT context in use.

Application Logic Layer

a. Modular Controllers
   Each IoT device in this application has a dedicated control module. For instance, a temperature sensor has an independent control module for data acquisition and configuration, separate from other devices. This approach allows new devices to be added to the application without affecting the overall application logic.
b. State Management
   Flutter provides several state management options (such as Provider, Bloc, or Riverpod) that allow each UI module to be updated in real-time according to the latest data from IoT devices. Each control module connects to the backend, ensuring that the user interface always displays accurate data.

Backend Communication Layer

a. Service Module
   This service module is responsible for handling various communication protocols (such as MQTT, HTTP, and WebSocket), enabling the application to reliably interact with IoT devices or backend servers. For example, the MQTT module allows communication with IoT brokers, while HTTP allows data access from the cloud.
b. Data Processing Module
   Data from IoT devices is pre-processed before display, such as through filtering or averaging, so that the UI only presents data ready for use.
c. API Management
   This module manages backend API endpoints, allowing new endpoints to be easily added as the application interacts with different IoT devices or platforms.

Configuration and Settings Layer

a. Device Configuration Module
   Users can dynamically add or configure devices through the UI, including sensor reading frequency and communication methods. This module facilitates device configuration according to specific user needs.
b. Dynamic Configuration
   The application supports dynamic configuration changes without requiring reprogramming. Users can add new sensors, change protocols, or adjust data visualization settings as needed, making the application versatile for various IoT implementations.

Cloud Integration Layer

a. Cloud Synchronization Module

This module is responsible for data synchronization between the application and the cloud, including data storage and integration with IoT platforms such as AWS IoT, Google Cloud IoT, or Microsoft Azure IoT.

b. Notification and Remote Control Module
The application can send real-time notifications when device errors or data anomalies occur and allows users to control devices remotely through cloud integration.

Security Layer

a. Authentication and Authorization Module
This module ensures that only authorized users can access the application, using OAuth or token-based authentication methods.

b. Data Encryption
Data exchanged between the application and devices is encrypted with standards such as TLS, and cloud communication is protected with similar encryption methods.

This modular design aims to ensure that each component in the system has a clear function and is well-organized within a directory structure, as shown below:

```
/lib
├── /ui               # Contains all user interface components, modular widgets, main screen
├── /controllers      # Controls IoT device logic, each device has an independent controller
├── /services         # Handles backend communication modules using MQTT, HTTP, or WebSocket
├── /models           # Data models for devices and sensors
├── /config           # Configuration settings for devices and dynamic configurations
├── /cloud            # Modules for cloud synchronization and remote device control
└── /security         # Modules for authentication and data encryption
```

With this architecture, the proposed IoT application can be designed to provide flexibility and scalability across various implementation scenarios without requiring extensive code adjustments. This modular structure allows developers to modify or add components as needed for operational requirements without disrupting the entire system.

RESULTS AND DISCUSSION

The proposed modular architecture was designed to support the flexibility and scalability of IoT monitoring and control applications, allowing adaptation to various user needs and scenarios. To evaluate its potential, a simulation was conducted using a smart home scenario to demonstrate the modular architecture's flexibility in handling dynamic configuration, device addition, and MQTT communication integration.

In a smart home scenario, users are expected to monitor and control smart home devices, such as lights, air conditioners, and security systems, from a single integrated application. The application must be able to add locations, devices, and configure communication settings without requiring developers to make significant code changes. This simulation evaluates how this architecture's modularity supports various user requirements, including:

1. Dynamic Location Addition: Users can add new locations (e.g., living room, bedroom, or kitchen) in the application through a modular user interface. The Device Configuration Module in the Configuration and Settings Layer allows users to dynamically specify new locations and customize the dashboard display to show data from relevant devices in each location.
2. Device Addition and Configuration: The application supports the addition of new devices, such as temperature sensors, motion sensors, and actuators. Users can add these devices through the Device Configuration interface, which allows users to set sensor reading frequency, configuration parameters, and the type of communication protocol (e.g., MQTT or HTTP) used.
3. MQTT Communication Integration: The MQTT protocol, commonly used in IoT applications, allows real-time communication between the application and IoT devices. In this simulation, MQTT transmits data from temperature and motion sensors, ensuring that the application can display accurate and up-to-date data.

The simulation results show that the proposed modular architecture effectively supports dynamic configuration adjustments and protocol integration, this is shown in Figure 1 below. This validates that Flutter-based modular architecture enables quick adaptations across diverse IoT scenarios.

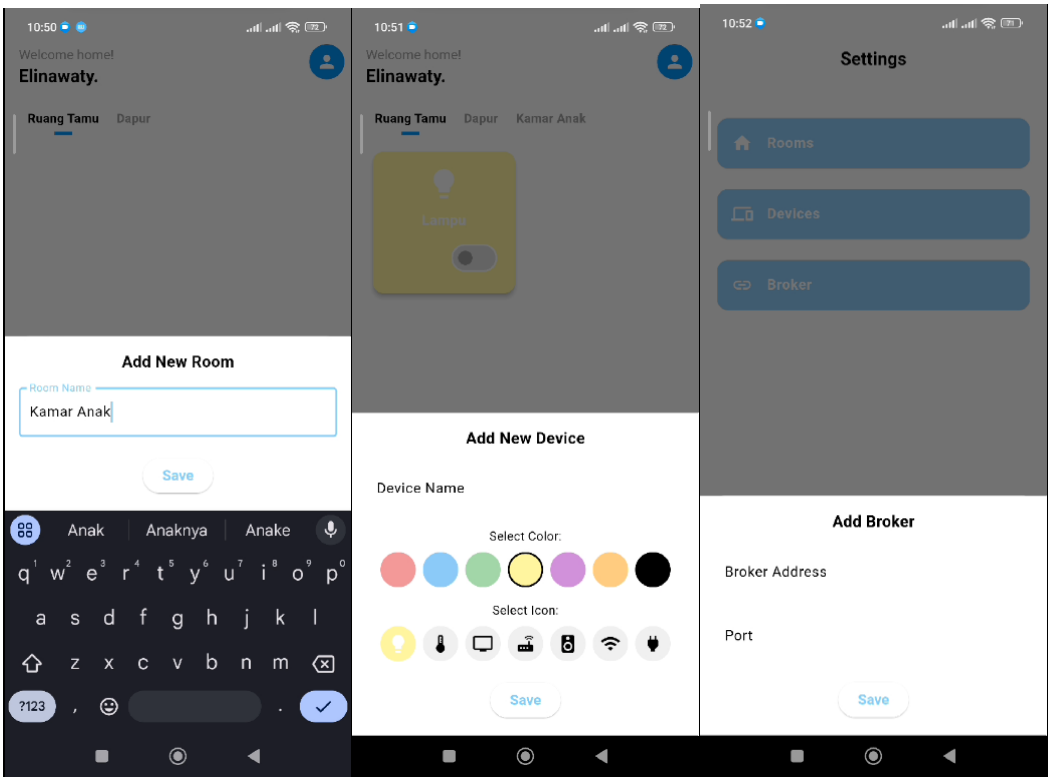

FIGURE 1. Description

CONCLUSIONS

This research presents a modular architecture design for a Flutter-based IoT monitoring and control application, aimed at providing high flexibility and scalability across various implementation scenarios, from smart homes to industrial settings. Through this modular approach, each component within the architecture can operate independently, allowing the addition or removal of features without affecting the entire application. Simulation results in a smart home scenario demonstrate that this design allows users to add locations, devices, and communication configurations dynamically as needed, proving the effectiveness of modular architecture in supporting diverse IoT adaptation and configuration.

The success of this modular design is primarily supported by Flutter's capability in cross-platform interface development and its compatibility with common communication protocols such as MQTT and HTTP. Additionally, the layered design, which includes User Interface, Application Logic, Backend Communication, Configuration and Settings, Cloud Integration, and Security layers, ensures that the application can evolve and adapt easily to future technological and operational needs.

By proposing a flexible and customizable architecture, this research contributes to the development of a more user-centered and adaptable IoT application, providing a strong foundation for the development of more complex IoT applications. In the future, this architecture holds the potential for implementation in large-scale scenarios requiring advanced data management and cloud integration, such as industrial environments or smart cities. With this modular design, developers have the flexibility to expand application functionality without requiring major modifications, making it a future-proof solution for various IoT needs.

ACKNOWLEDGMENTS

This research was funded and supported by the Research and Community Service Center of the Bengkalis State Polytechnic.

REFERENCES

[1] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review," *Journal of Computer and Communications*, vol. 03, no. 05, pp. 164–173, 2015, doi: 10.4236/jcc.2015.35021.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, Dec. 2015, doi: 10.1109/COMST.2015.2444095.

[3] N. Van Co, H. L. Le, N. T. Tran, and H. N. Le, "Research and propose flexible IoT solutions with radio communication networks and sensors for measurement, control," *Ministry of Science and Technology, Vietnam*, vol. 65, no. 1, pp. 32–37, Jan. 2023, doi: 10.31276/VJST.65(1).32-37.

[4] C. Martín, J. Hoebeke, J. Rossey, M. Díaz, B. Rubio, and F. Van den Abeele, "Appdaptivity: An Internet of Things Device-Decoupled System for Portable Applications in Changing Contexts," *Sensors*, vol. 18, no. 5, p. 1345, Apr. 2018, doi: 10.3390/s18051345.

[5] M. De Sanctis, H. Muccini, and K. Vaidhyanathan, "Data-driven Adaptation in Microservice-based IoT Architectures," in *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*, IEEE, Mar. 2020, pp. 59–62. doi: 10.1109/ICSA-C50368.2020.00019.

[6] chao zhang, G. yao Cheng, A. ju Shao, D. dong Wang, and T. Zhang, "Modular design of aircraft system software," in *Third International Conference on Artificial Intelligence and Computer Engineering (ICAICE 2022)*, X. Li, Ed., SPIE, Apr. 2023, p. 221. doi: 10.1117/12.2671511.

[7] K. Yelamarthi, M. S. Aman, and A. Abdelgawad, "An Application-Driven Modular IoT Architecture," *Wirel Commun Mob Comput*, vol. 2017, pp. 1–16, 2017, doi: 10.1155/2017/1350929.

[8] S. Sharma, S. Khare, V. Unival, and S. Verma, "Hybrid Development in Flutter and its Widgits," in *2022 International Conference on Cyber Resilience (ICCR)*, IEEE, Oct. 2022, pp. 1–4. doi: 10.1109/ICCR56254.2022.9995973.