Matrix: Jurnal Manajemen Teknologi dan Informatika
Volume 15 Issue 2 Year 2025 Pages 87-101
DOI: http://dx.doi.org/10.31940/matrix.v15i2.87-101
URL: http://ojs2.pnb.ac.id/index.php/matrix

Android-based multi-IoT fish feeding system: An end-to-end information system approach

Putu Nanda Arya Adyatma [1]*, Putu Wira Buana [2], I Made Agus Dwi Suarjaya [3]

[1,2,3] *Department of Information Technology, Universitas Udayana, Indonesia*

*Corresponding Author: *putunanda928@gmail.com*

Abstract: The ornamental fish industry in Indonesia has experienced significant growth, positioning the country as the second-largest global exporter of ornamental fish in 2020. However, fish shop owners still face operational challenges, especially in managing consistent and timely feeding across multiple aquariums. Manual feeding practices often lead to inefficiencies and can compromise fish health and water quality. This study presents an end-to-end fish feeding information system integrated with an Android mobile application, designed to address these challenges. System development in this study employs waterfall method. The system supports automated fish feeding routines, device management, and multi-user access with token-based authentication, enabling fish shop owners to operate multiple feeders under a single account. Communication between IoT devices and the backend server utilizes MQTT, ensuring independent control of each feeder through unique topics. The system introduces a novel architecture that supports multi-user, multi-device operations in an end-to-end feeding workflow, improving scalability and efficiency compared to existing single-device systems. System testing, including black box and load testing, demonstrated robust performance, with all test scenarios passing successfully and an error rate of 0.00% during high-load simulations involving up to 100 virtual users. These results indicate that the system effectively addresses existing limitations in fish feeding management and is capable of supporting multiple users and fish feeder devices simultaneously. Further development is recommended to enhance infrastructure, security, and scalability for real-world deployment.

Keywords: automated fish feeding, android, information system, IoT, mobile application

History Article: Submitted 4 June 2025 | Revised 30 July 2025 | Accepted 4 August 2025

How to Cite: P. N. A. Adyatma, P. W. Buana, and I. M. A. D. Suarjaya, "Android-based multi-IoT fish feeding system: an end-to-end information system approach", *Matrix: Jurnal Manajemen Teknologi dan Informatika*, vol. 15, no. 2, pp. 87–101, 2025. doi.org/10.31940/matrix.v15i2.87-101.

Introduction

Ornamental fish are among the most popular types of pets across various demographic in Indonesia [1]. The hobby of keeping ornamental fish continues to grow, supported by both community development and the increasing scale of aquaculture industry. According to the International Trade Center, Indonesia ranked third globally in ornamental fish exports in 2019, following Japan and Singapore. That year, the export value reached USD 7.8 million for marine ornamental fish and USD 25.2 million for freshwater ornamental fish [2]. By 2020, Indonesia had captured 11.35% of the global market share, becoming the world's second-largest ornamental fish exporter [3]. These figures underscore the economic potential of ornamental fish as a leading commodity in the country.

This growth has not only increased public interest in ornamental fish as pets but has also driven the expansion of fish shops across Indonesia. These fish shops typically manage multiple aquariums with various freshwater species. However, many of these businesses still rely on manual operations, which introduces significant challenges in aquarium and fish management. One of the most pressing issues for fish shops owner is feeding management, which is also one of fish welfare issues of ornamental fish trade [4].

Manual feeding practices often lead to inefficiencies, such as delayed feeding and inconsistent portions [5]. These problems are compounded when managing a large number of aquariums, making it difficult for fish keeper to remember and follow proper feeding and maintenance schedules. Inaccurate or inconsistent feeding does not only result in material losses,

but also degrades water quality and jeopardizes fish health [6]. Overfeeding contributes to excess waste and poor water conditions, while underfeeding inhibits growth and lowers immunity [7]. This problem shows how important it is to maintain the quality and quantity of feeding to maintain the quality of fish products for ornamental fish shops.

To address these operational challenges, an automated fish feeding system is needed to support consistent and timely feeding routines. Automated fish feeding technology has been shown to improve both operational efficiency and feed consumption [8]. Several studies on IoT-integrated fish feeding systems have been conducted previously, including by Izak Habel et al. in 2023 [9], Komang Martadana Wijaya et al. in 2023 [10], Rafly Fernanda et al. in 2022 [11], and Husnul Khatimi et al. in 2022 [12]. These studies offered systems that enable remote monitoring and control of fish feeding. However, these previous systems have not provided a mechanism for managing multi-user authentication and are typically limited to single-IoT-device communication. Moreover, most existing systems do not provide features for fish feeder device registration and management independently, making them unsuitable for scaling to users who operate multiple fish feeders on their aquariums.

These limitations have significant implications in real-world applications, particularly in fish stores on hatcheries that operate multiple aquariums. Without multi-user support, the systems cannot be collaboratively accessed or supervised by more than one employee, reducing flexibility and increasing the risk of mismanagement. Similarly, the inability to register and manage multiple feeder devices under a single account makes the system impractical for environments where different tanks require separate feeding schedules. These constraints highlight the need for a scalable, multi-user, multi-device management system, a gap this study intends to address.

Unlike previous systems that primarily support single-user or single-device interaction, this study proposes an end-to-end multi-IoT-device fish feeding information system integrated with an android mobile application. The system is uniquely designed to handle the entire process from device installation and scheduling to real-time feeding execution and monitoring via mobile. It also introduces support for multi-user authentication and multiple device management under a single account, enabling fish shop owners to control several fish feeders independently. This approach reduces operational complexity and addresses scalability issues commonly faced by shops managing multiple aquariums. By integrating feeding automation, device management, and mobile control, the proposed solution offers a novel architecture that enhances efficiency and sustains optimal aquarium conditions.

Methodology

The research in this study employs the waterfall method, a structured and step-by-step (sequential) approach commonly used in system development [13]. The model of waterfall method is illustrated in Figure 1. The waterfall model begins with requirement analysis. This initial phase aimed at analysing and clearly defining the user's software requirements and specifications [14]. System design is the phase where system requirements are allocated to both hardware and software components, forming the overall system architecture. Implementation is the stage where the system is developed and implemented by writing software code. Integration and system testing is the phase where individual program units are integrated and tested as a completed system to verify whether it meets the defined software requirements. At this stage, the integration between the API, the mobile app, and the existing prototype IoT-based fish feeder device is carried out. In addition, comprehensive system testing is performed using black box testing and load testing. Operation and maintenance are the final phase, where the system and application are fully deployed and used by end users, while ongoing maintenance is carried out.

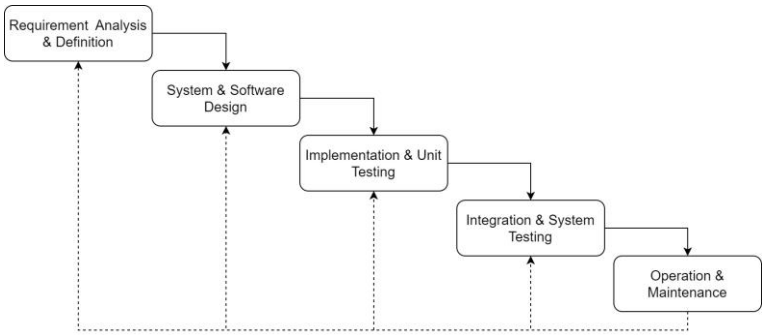


Figure 1. Waterfall model

System Design

System design describes the design of the overall fish feeding information system based on the results of the system requirement analysis that have been carried out. In this study, the system design includes system overview, use case diagram, database design, user and IoT fish feeder communication scheme.

System Overview

System overview illustrates the general workflow and architecture of the system in a visual format. The overview of end-to-end multi-IoT-device fish feeding information system based on Android mobile application is depicted in Figure 2.

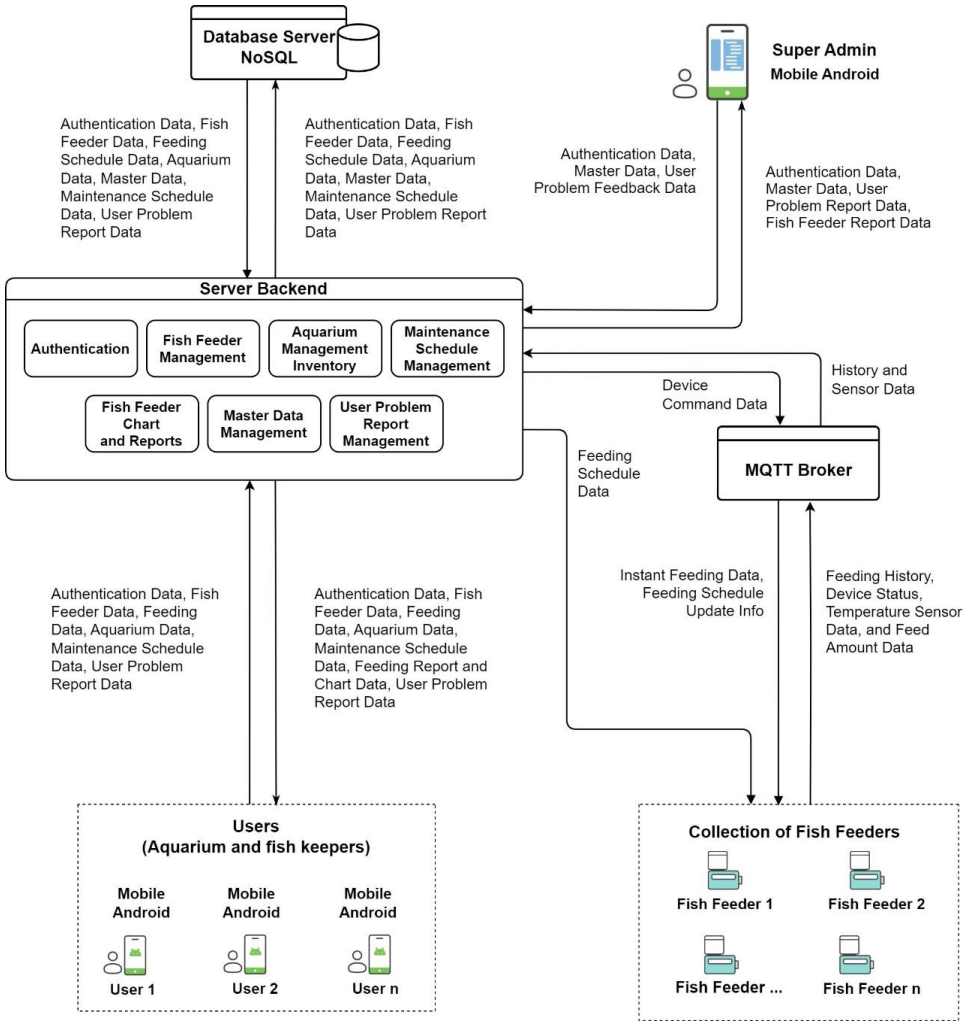


Figure 2. System overview

Several key entities are involved in the system, including a database server, backend server, MQTT broker, IoT fish feeder, and Android mobile frontend (user and super admin). The database server stores and provides access to data used by the system. The backend server processes data received from the frontend before storing it in the database and vice versa. It also communicates with the IoT fish feeder both directly and through the MQTT broker. The backend handles various service modules such as authentication management, fish feeder device management, aquarium inventory management, scheduled maintenance management, feeder reports and charts, master data management, and user problem report management. The frontend provides the user interface, displaying data retrieved from backend and enabling users to input data that will be processed and stored. There are two frontend applications, one for general user role and one for super admin role, both developed for Android mobile platforms. The super admin manages master data, monitors and responds to user issue reports, and oversees fish feeder usage reports. The general user responsible for caring for fishes and aquariums. They input authentication data, fish feeder information, feeding schedules, aquarium data, maintenance schedules, and problem report. The fish feeder device executes fish feeding commands and sends sensor data and feeding history to the database through the backend. The MQTT broker facilitates real-time communication between the backend and the fish feeder, especially for immediate actions such as feeding or updates to the feeding schedule. The system is designed to support multiple Android mobile users and multiple fish feeder devices, allowing each user to control more than one feeder unit simultaneously.

Use Case Diagram

A Use Case Diagram illustrates the interactions between users and the system. This diagram also be defined as a functional description of a system from perspective of its users [15]. The use case diagram of end-to-end multi-IoT-device fish feeding information system is presented in Figure 3. The use case diagram identifies two primary actors, the super admin and the user. The super admin is responsible for managing the entire fish feeding information system. This role has full access to master data management, fish feeder management, fish feeder report, problem report handling, and also information and library. The user can manage fish feeder and feeding data, aquarium inventory, maintenance schedules, view feeder reports, submit user problem report, and access information and library.

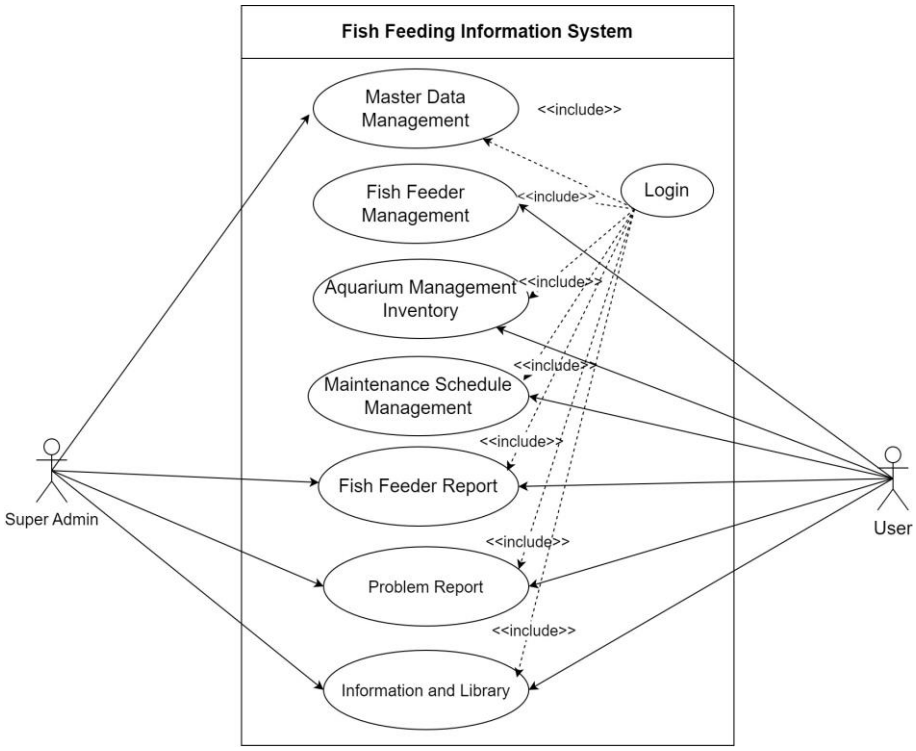


Figure 3. Use case diagram

Database Design

The database for the fish feeding system is designed using a NoSQL database approach. This NoSQL database allow us to store data in a JSON-like format, making it easy to represent and query nested hierarchical structures. This structure and format allow NoSQL database to excel in terms of speed performance compared to RDBMS [16]. The database is built using MongoDB, which organizes data into collections and documents [17]. The design of database is illustrated through a Physical Data Model, as shown in Figure 4. This database design adopts a NoSQL model, which enables embedded relationships by storying multiple data entries within a single field. The database consists of several collections, including user collection, OTP collection, device collection, deviceOrigin collection, feedingHistory collection, information collection, aquarium collection, fish collection, component collection, problemReport collection, sensorData collection and maintenanceSchedule collection.

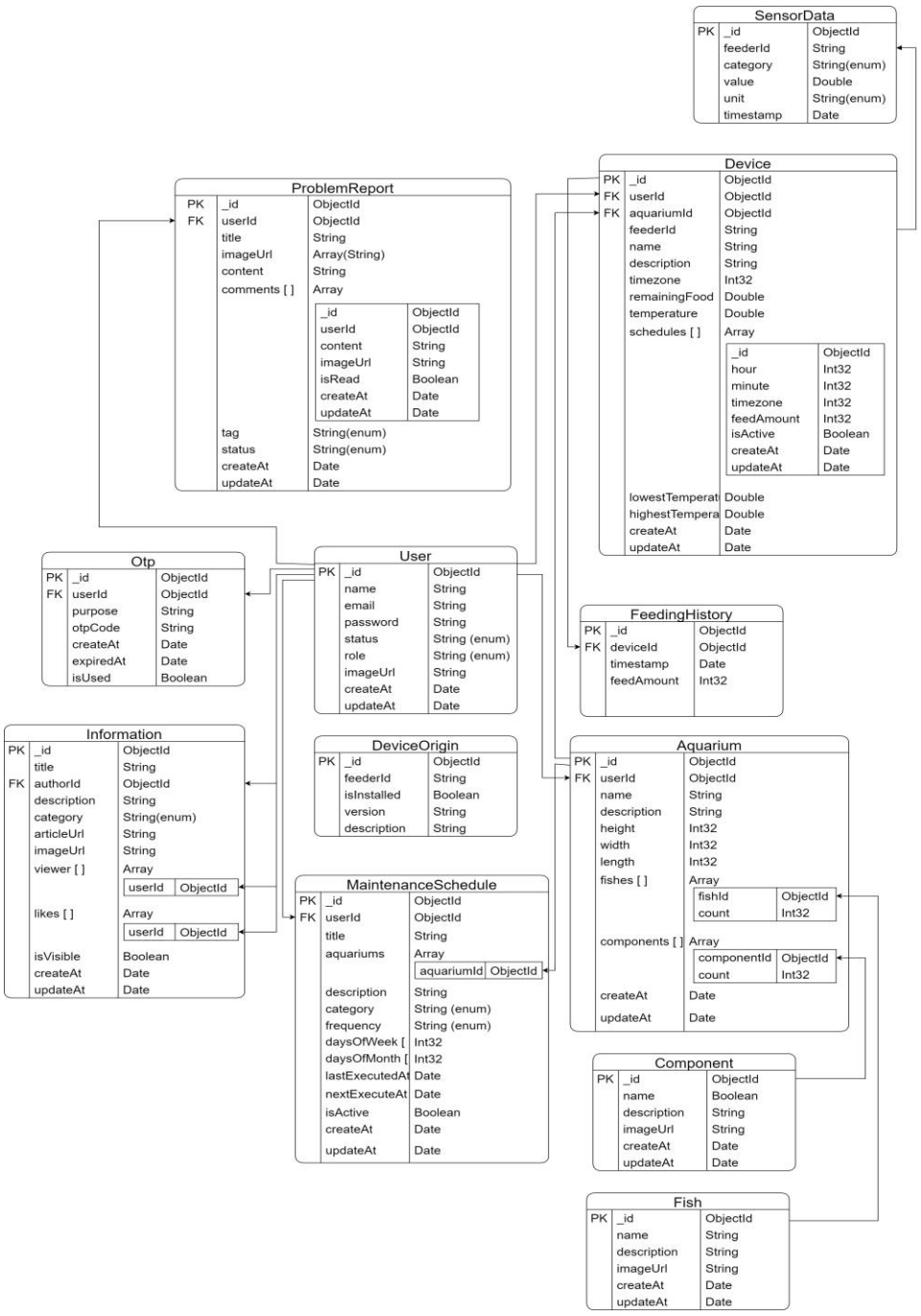


Figure 4. Physical data model

System Communication

The communication in the fish feeding system utilizes two types of protocols, HTTPS and MQTT. HTTPS (Hypertext Transfer Protocol Secure) is a communication protocol between the client and the server via the internet that is protected by encryption. It is used for interactions between the Android user and the server. MQTT, on the other hand, is used for communication between the IoT device (the fish feeder) and the server. This protocol is lightweight and follows a publish (PUB) and subscribe (SUB) communication [18]. MQTT allows the communication with low latency, enabling high responsiveness for features such as real-time feeding, schedule synchronization commands, and sensor data transmission.

The backend services implement comprehensive input data validation, including mandatory value check, data type validation, unique email verification, and ID validation. For secure information access, all REST API requests are protected using JSON Web Token (JWT) authorization, which authenticates and enforces role-based access control. A new token is generated upon user login and is set to expire after seven days. MQTT communication between device and backend is secured using CA (Certificate Authority) certificate, TLS encryption, and unique feeder IDs, ensuring confidentiality and integrity of the fish feeder communication.

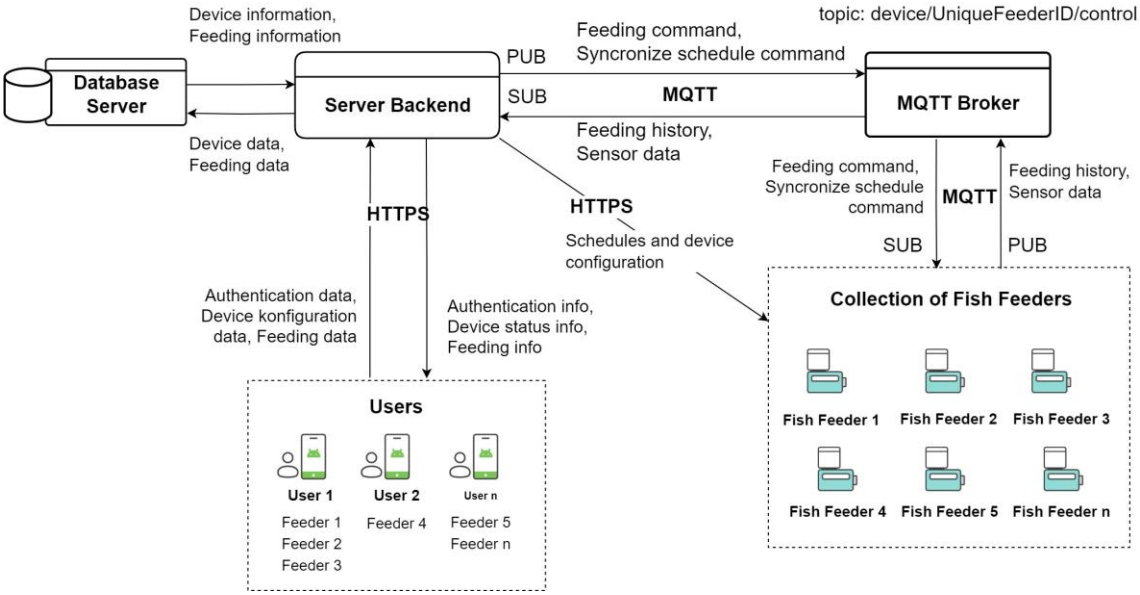


Figure 5. System communication diagram

Figure 5 illustrates the communication flow within the system. Users interact with the system via APIs on the backend server, sending authentication data, device configuration data, and feeding data, and receiving authentication information, device status, and feeding information. These API communications use the HTTPS protocol. Each user must register their fish feeder so the device data can be linked to their account. Fish feeders communicate with the backend through an MQTT broker, enabling real-time messaging, but can also communicate directly via HTTPS API. Both the fish feeder and the backend use publish and subscribe communication on the topic device/UniqueFeederID/control. Each fish feeder has a unique feeder ID, ensuring independent communication per device. The backend publishes feeding commands and schedule synchronization data, while the fish feeder publishes feeding history and sensors data. Due to limited payload resources, schedule data synchronization is requested over HTTPS rather than MQTT. All data is stored in the server database for future use.

IoT Fish Feeder

The IoT fish feeder is a device designed to ensure automatic, consistent, and accurate fish feeding according to a predefined schedule. The IoT fish feeder device used in this study is a prototype previously developed, as depicted in Figure 6. This device was built using an ESP8266 microcontroller and is equipped with several sensors, including a DS18B20 sensor for measuring

the water temperature in the aquarium and a load cell sensor for detecting the remaining fish food in the container. The feeding system is powered by an SG90 servo to push the grain feed with a size of 1mm. This device communicates via MQTT and HTTPS, as explained in the system communication section. This study does not cover the wiring details or component circuitry within the IoT device, as its focus is on developing the information system to support integration between the IoT device and the mobile application.

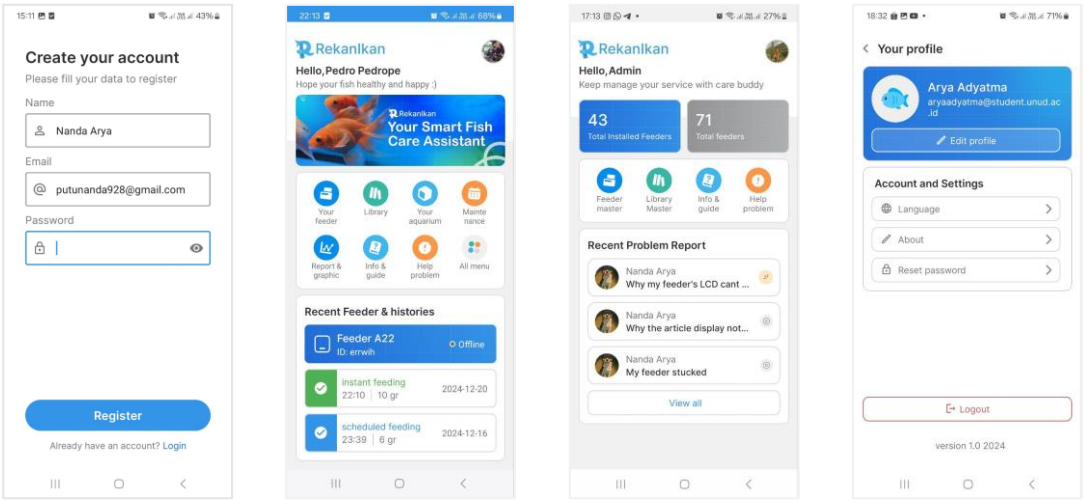
Figure 6. Fish feeder prototype

Results and Discussions

Results

App User Interface

The user interface (UI) is a crucial component in the information system development. It serves as the bridge between users and the system to ensure seamless interaction [19]. The entire user interfaces were built on the Android mobile platform using the Android Studio IDE in Kotlin programming language. The user interface of fish feeding information system is divided into two based on its role, namely the user interface for user role and the user interface for the super admin role.

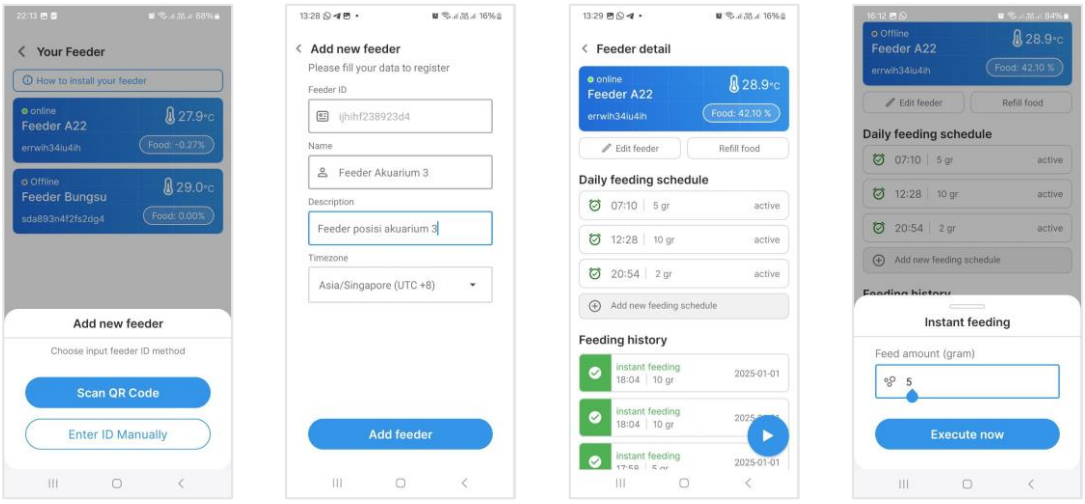


(a) (b) (c) (d)

Figure 7. User interface of register page, home page, and profile page

Figure 7 shows the user interface of the fish feeding system. Every user has to create an account first to enjoy the features provided in register page as shown in Figure 7 (a). User who already have an account and successfully login will be directed to the home page. The home page is divided into two, first for general user role as shown in Figure 7 (b) and second for super admin

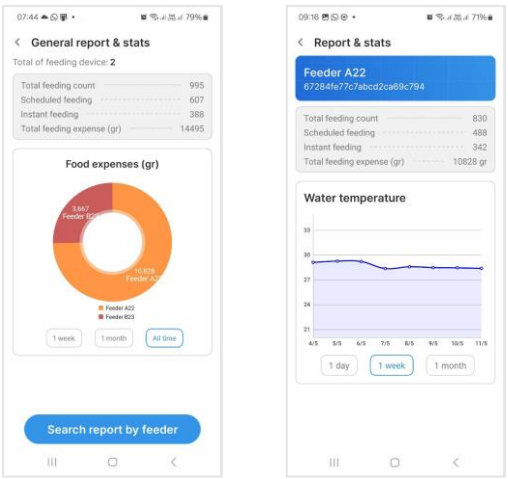
as show in Figure 7(c). The user home page provides several features such as feeder management, library, aquarium management, maintenance schedule management, report and graphic, info and guide, and user problem report. Meanwhile, the super admin home page provides feeder master data management, library master management, guide info management, and user report handling. Users can view their profile details on the profile page shown in Figure 7 (d).



(a) (b) (c) (d)

Figure 8. User Interface of Fish Feeder Management

Figure 8 illustrates the user interface for the fish feeder management menu. Users can independently install and register their fish feeder using two methods, by scanning the device's QR code or manually entering the device ID, as shown in Figure 8 (a). The registration process requires input such as the device ID, name, description, and the time zone where the feeder is located (Figure 8 (b)). Once the data is verified, the device will be successfully added to the system. Figure 8 (c) displays the interface for a registered and active device. It shows information such as the device's status, feeding schedules, and feeding histories. Users can also trigger a real-time feeding action by pressing the blue floating play button and specifying the desired amount in grams, as illustrated in Figure 8 (d). Every executed feeding will be followed by push notification from the server that indicated successful feeding.

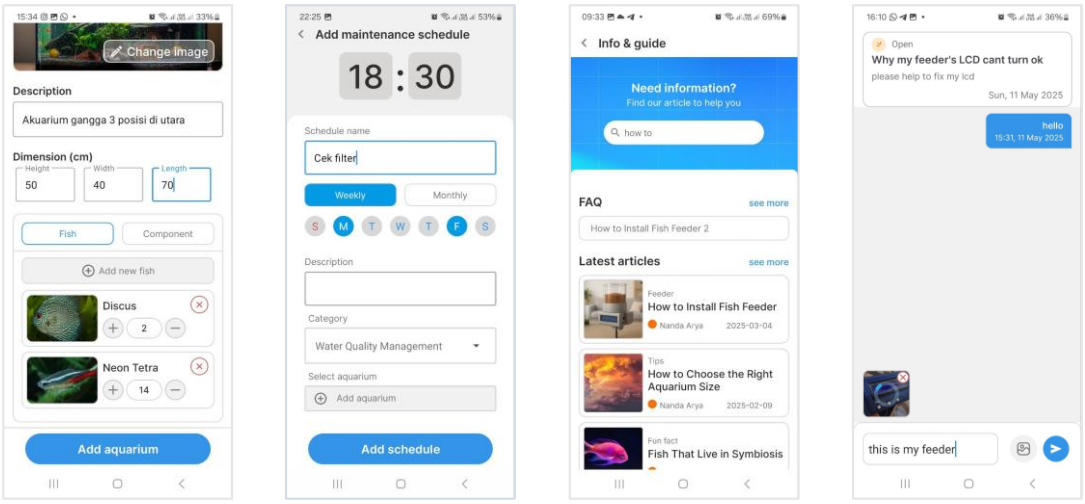


(a) (b)

Figure 9. User Interface of Fish Feeder Report and Stats

Figure 9 shows the user interface of the fish feeder report and statistics feature. Users with multiple devices can view an overall report of all their feeders. Figure 9 (a) presents the general

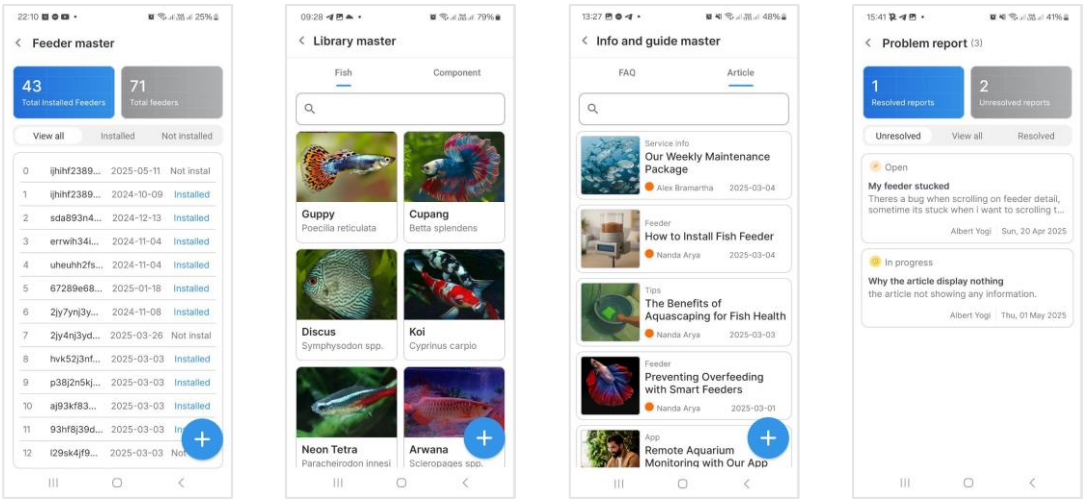
report and statistics view, which includes feeding statistics across all fish feeder devices. Users can also access more detailed device-specific reports, as shown in Figure 9 (b). This page displays feeding statistics and a temperature trend graph based on data captured by feeder's sensor.



(a) (b) (c) (d)

Figure 10. User Interfaces of Additional User Role App Features

The application offers additional features to support users in maintaining their aquariums, such as aquarium management, maintenance scheduling, guide information, and problem report chat as shown in Figure 10. Figure 10 (a) displays the aquarium management feature, which helps users keep track of their aquariums. To add a new aquarium, users need to provide an image, name, description, dimension, and details of the fish and components inside the aquarium. Figure 10 (b) shows the page for adding aquarium maintenance schedules. Users can set schedules on a weekly or monthly basis and link them to a specific aquarium. Active schedules will trigger automatic reminder notifications. User can visit the info and guide page, which contains a collection of FAQs and articles related to fish care and the app's services, as shown in Figure 10 (c). Additionally, users can report issues or problems related to the app or the fish feeder device through the problem report feature. All reports will be sent to the super admin for the resolution (Figure 10 (d)).



(a) (b) (c) (d)

Figure 11. User Interfaces of Super Admin Role App Features

The super admin side of the application provides several features to assist in managing service-related data. There are four main features namely feeder master management, library

master management, info and guide master management, and problem report management, as shown in Figure 11. Super admin can manage master data used for registering fish feeders by general users. Figure 11 (a) displays the dashboard of the feeder master page, which shows the total number of installed devices. Super admins can add multiple feeder master entries at once using the blue floating button. They can also manage master data for fish and aquarium components via the library master page, as seen in Figure 11 (b). This data is utilized by users when managing aquariums and accessing the library feature. All FAQ entries and articles can be managed on the info and guide master page, shown in Figure 11 (c). Figure 11 (d) illustrates the problem report management page, where super admins can monitor and respond to incoming reports. For each report, they are required to engage in a chat discussion with the user and update the report's processing status accordingly.

Backend and Cloud Service

The backend is a critical component of the fish feeding information system. Backend is responsible for behind-the-scenes processes such as database interactions and executing server-side logic [20]. Additionally, the backend functions as the central integration hub for the other services required by the system, such as FCM (Firebase Cloud Messaging) for push notifications, HiveMQ as the MQTT broker, and MinIO as the cloud object storage. The entire backend and API on this information system was built using the Express JS framework using the JavaScript language.

The backend system requires a server that can operate continuously to host and execute the backend codebase. In this research, the entire backend is deployed on the Railway serverless platform. Railway is a cloud-based Platform as a Service (PaaS) that provides an environment for managing container-based backend applications. The backend is deployed in the Singapore region, aiming to minimize communication latency with users, the majority of whom are located in Indonesia. The allocated resources include up to 8 vCPUs and 8GB of RAM.

Testing

The testing of the fish feeding information system was carried out using blackbox testing and load testing techniques. Black box testing evaluates the software without examining its internal code. This technique is focusing solely on the provided inputs and the expected outputs [21]. This black box testing was carried out both independently by author at home and by an employee of the Gangga Fish ornamental fish shop.

Table 1. Summary of black box testing results by test group on mobile app

Test Group	**Number of Test Cases**
Register	6
Login	4
Home	3
Profile	5
Feeder	5
Library	5
Aquarium	6
Maintenance	5
Report and graphic	3
Info and guide	8
Problem report	8
Feeder master	7
Library master	11
Info and guide master	11
Problem report admin	7
Notification	6
Total	**100**
All test cases passed successfully (100%)	

Black box testing was applied to all features on both the super admin and user sides. This technique was also used to ensure that the integration between the feeding device and the system functioned properly. Table 1 presents the list and results of black box testing conducted on the mobile application. The test cases are grouped based on main pages and features, with the number of scenarios varying according to the complexity of each function. Based on the results, all 100 test cases were successfully executed without any failures and met the predefined functional expectations. This mobile application was tested on multiple devices including Vivo Y16 Android version 12 with 8GB RAM and Samsung A54 Android version 14 with 8GB RAM to ensure compatibility and performance consistency. All features on these devices work properly without any significant issues.

A preliminary usability test was conducted with one employee from a local ornamental fish shop, Gangga Fish. The employee was asked to perform 27 general scenarios in user role features. This basic task such as user authentication, device installation, feeding schedule creation, feeding execution, aquarium management, maintenance schedule creation, and user report creation through the mobile app. The user was able to complete all test scenarios successfully and provided positive feedback regarding the service clarity and ease to use.

Load testing is a method used to evaluate system performance under varying levels of load request [22]. This testing ensures that the APIs in the developed fish feeding information system can handle the expected load as planned, providing insights into the system's performance and behavior. The tests were performed on several API endpoints using the Postman application. The endpoints tested included login, get all fish feeders, get fish feeder by ID, and get all feeding history. Figure 12 shows the load testing of the login endpoint, executed using Postman. The load testing was conducted using a ramp-up load profile over a 20-minute duration with 100 virtual users (VUs).

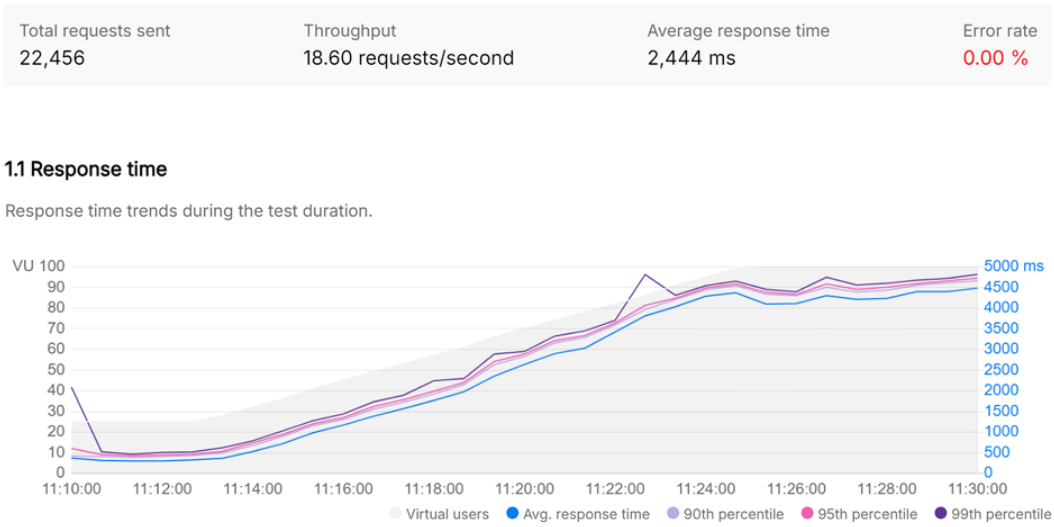


Figure 12. Load testing on login endpoint using Postman

Similar testing procedures were applied to the remaining three endpoints which are get all fish feeders, get fish feeder by ID, and get all feeding history. All the load testing data are summarized in Table 2. This table presents the total number of requests sent, throughput, average response time, and error rate for the four tested endpoints. The results demonstrate that all tests were successful, with an error rate of 0.00%. The highest total number of requests sent was recorded in the load test for the "get feeder by ID" endpoint, while the lowest number was recorded in the test for the "login" endpoint.

Table 2. Summary of load testing results

Endpoints	Total Request Sent	Troughput (request/s)	Average response time (ms)	Error rate (%)
Login	22,456	18.60	2,444	0.00
Get all feeders	62,382	51.66	61	0.00
Get feeder by ID	66,078	54.68	53	0.00
Get all history	27,924	23.12	1,787	0.00

Black box testing also applied to the fish feeding and fish feeder device functionality, the results are described in Table 3. This testing ensures that functionality of the feeding device works consistently and as expected. This testing consists of several scenarios and each scenario run 20 times repetition. All scenarios were successfully carried out with their respective results in the table. These results show that the system integration of the fish feeder has met the functional needs as well as reliability in emergency situations such as internet disconnection.

Table 3. Summary of fish feeder and feeding testing results

Testing Scenario	Results
Duration of fish feeder connection	2 minutes 12 secs 47 ms (average)
Duration of instant feeding delay	1 sec 34 ms (average)
Accuracy of scheduled feeding time	100% (on schedule)
Execution of instant feeding	100% (executed)
Notification of executed feeding	100% (executed)
Notification of delayed executed feeding	100% (executed)
Notification of disconnected feeder device	100% (executed)

Discussions

The end-to-end fish feeding information system developed in this study is designed to support automated fish feeding, complete with features that assist in fish and aquarium care. The application is built comprehensively, from device registration to autonomous feeding. It incorporates token-based authentication using JWT to support multi-user access with different roles, and enables operation of multiple feeders under a single user account. This is made possible through MQTT communication with unique topics, allowing each feeder to operate independently. The android application includes two main roles, super admin (responsible for monitoring and managing operational data) and user (who uses the app to care for their fish).

System testing was conducted to ensure performance met the standards, using both black box testing and load testing techniques. The black box testing was carried out both independently by the author and an employee from the Gangga Fish. The black box testing covered 100 scenarios across all super admin and user features, all of which passed successfully without errors. Additionally, a preliminary usability testing involving 27 general user scenarios was conducted by the fish shop employee, who completed all task without difficulty and provided positive feedback. For performance evaluation, load testing was conducted using a 20-minute ramp-up simulation with 100 virtual users. Result showed that all four tested endpoints passed with an error rate of 0.00%. The "get feeder by ID" endpoint recorded the highest total requests (66,078) with an average response time of 53 ms, as it only retrieves a single record based on an ObjectID in MongoDB. The login endpoint had the lowest total request (22,456) with an average response time of 2,444 ms, due to authentication processes involving email matching, password decryption, and simultaneous token creation for 100 clients.

This research demonstrates that the end-to-end fish feeding information system can handle multiple users and fish feeder devices simultaneously, filling a gap in previous research. However, further improvement is needed to optimize system security and cloud infrastructure as well as increase user and device testing to simulate large-scale market usage. This system is also open to potential to be further integrated with another cloud platforms to enable real-time

synchronization, remote device monitoring, and scalable data analytics. Such integration would support larger deployment in smart aquaculture farming ecosystem.

Limitations

While this proposed system demonstrates effective integration of multi-user and multi-IoT-device functionality in a controlled environment, several limitations remain. First, the system has not yet been tested with large-scale condition involving up to 100 fish feeder devices simultaneously and up to 100 real users. This may affect the reliability of the backend performance in field conditions. Second, the test only uses two android devices, namely Android version 14 with 8 GB RAM and Android version 12 with RAM 12 GB. These testing devices are not representative of the overall the overall specifications available in the community. Third, data encryption on this system communication is limited to HTTPS and basic MQTT SSL/TLS. Additional security layers such as end-to-end encryption are not yet implemented. Lastly, the end-user test involved only a single participant due to limited device availability. These finding are not sufficient to generalize usability across broader user groups in real-world scenario.

Conclusion

The end-to-end multi-IoT-device fish feeding information system based on Android mobile application developed in this study successfully demonstrates its ability to automate fish feeding, handle multiple devices and users, and perform reliably under load. System testing covering both black box and load testing showed strong and reliable performance. All test scenarios passed without issue, and high-load simulations with up to 100 virtual users resulted in a 0.00% error rate. Further development is recommended to optimize system performance, security, and infrastructure, as well as to expand testing to simulate large-scale usage scenarios, ensuring the system is robust for real-world deployment.

Acknowledgments

The author would like to express heartfelt gratitude for the support and contributions from everyone involved in completing this research. Special thanks go to Ir. Putu Wira Buana, S.Kom., MT and Dr. Eng. I Made Agus Dwi Suarjaya, ST., M.Eng. for their guidance and insights throughout the development of this end-to-end fish feeding information system. Gangga Fish Ornamental Fish Shop, which has kindly provided opportunities for conductiong this research.

References

[1] A. Fu'adi, D. A. F. Yuniarti, A. Prianggono, and B. J. M. Putra, "Pembangunan Aplikasi Katalog Online Berbasis Mobile Sebagai Fasilitas Pemasaran Bagi Pembudidaya Ikan Hias," *Journal of Electrical, Electronic, Mechanical, Informatic and Social Applied Science*, vol. 1, no. 2, pp. 25–31, Dec. 2022, [Online]. Available: https://eemisas.aknpacitan.ac.id/index.php/eemisas/article/view/15/9

[2] Neneng Tita Amalya, Yhonanda Harsono, and Tri Sulistyani, "Manajemen Usaha Budidaya Ikan Hias Dalam Upaya Meningkatkan Penjualan Pada Kelompok Budidaya Ikan Hias," *Abdimas Awang Long*, vol. 6, no. 1, pp. 1–6, Jan. 2023, doi: 10.56301/awal.v6i1.659.

[3] A. R. Prihikmat, "Terapkan Standardisasi, Maksimalkan Potensi Indonesia jadi Pusat Ikan Hias Dunia." [Online]. Available: https://bsn.go.id/main/berita/detail/19025/terapkan-standardisasi-maksimalkan-potensi-indonesia-jadi-pusat-ikan-hias-dunia#:~:text=Melalui%20peningkatan%20mutu%20budi%20daya,%2C70%25%20pada%20tahun%202021.

[4] C. Maia, A. Gauy, and E. Gonçalves-de-Freitas, "Fish Welfare in the Ornamental Trade: Stress Factors, Legislation, and Emerging Initiatives," *Fishes*, vol. 10, p. 224, May 2025, doi: 10.3390/fishes10050224.

[5] A. Gunawan, Affandi, and M. A. Sekamdo, "Inovasi Teknologi Budidaya Ikan Yang Berkelanjutan Di Provinsi Sumatera Utara," *ABDI SABHA (Jurnal Pengabdian kepada Masyarakat)*, vol. 3, no. 2, pp. 301–308, Jun. 2022.

[6] E. Rajagukguk, Mulyadi, and U. MT, "Pengaruh Waktu Pemberian Pakan Terhadap Pertumbuhan dan Kelulushidupan Ikan Nila Merah (Oreochromis niloticus) Dengan Sistem

Resirkulasi," *Jurnal Perikanan dan Keluatan*, pp. 45–52, 2018, [Online]. Available: https://jom.unri.ac.id/index.php/JOMFAPERIKA/article/viewFile/21231/20542?form=MG0AV3

[7] D. P. A. N. Aini and A. Rohman, "Strategi Aspek Manajemen dalam Mengembangkan Usaha Budi Daya Ikan Hias Di Lamongan dalam Persepektif Studi Kelayakan Bisnis," *Jurnal Media Akademik (JMA)*, vol. 2, no. 6, pp. 1–15, 2024.

[8] S. Razali, A. Rahman, and A. Damora, "Penerapan Sistem IoT Berbasis Energi Surya untuk Pemberian Pakan Otomatis dan Pemantauan Kualitas Air pada Budidaya Udang Vaname," *Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi)*, vol. 9, no. 1, pp. 389–398, Jan. 2025.

[9] I. Habel Wayangkau, D. Abdianto Nggego, C. Chotimah, and N. Patawaran, "Internet of Things Implementation in Automatic Fish Feeding," 2023. [Online]. Available: www.techniumscience.com

[10] K. M. Wijaya, I. N. Piarsa, and P. W. Buana, "Design and Development Automatic Fish Feed Measuring and Feeding System Based on Internet of Things," *Jurnal Ilmiah Merpati*, vol. 11, no. 2, pp. 115–126, Aug. 2023.

[11] R. Fernanda and T. Wellen, "Perancangan Dan Implementasi Sistem Pemberi Pakan Ikan Otomatis Berbasis IoT," *Jurnal Teknik Informatika dan Sistem informasi*, vol. 9, no. 2, pp. 1261–1273, 2022, [Online]. Available: https://doi.org/10.35957/jatisi.v9i2.2030

[12] H. Khatimi, M. Maulida, N. F. Mustamin, and A. F. Zulkarnai, "Pengembangan Automatic Fish Feeder untuk Meningkatkan Produksi Keramba Apung Kelompok Budidaya Ikan," *ILUNG: Jurnal Pengabdian Inovasi Lahan Basah Unggul*, vol. 1, no. 3, pp. 34–41, Mar. 2022.

[13] P. G. K. Mahadiputra, I. M. A. D. Suarjaya, and K. S. Wibawa, "Automatic Pet Feeder Rotational Model Using MQTT and Mobile Application," *Jurnal Ilmiah Merpati*, vol. 12, no. 2, Aug. 2024.

[14] A. Taqwiym and Nurasiah, "Aplikasi Pencatatan Perhitungan Laba Rugi berbasis Desktop pada PT. Fachri Syafii Akbar," *SISFOKOM (Sistem Informasi dan Komputer)*, vol. 9, no. 1, pp. 69–76, 2020.

[15] L. Setiyani, "Desain Sistem : Use Case Diagram," in *Prosiding Seminar Nasional Inovasi dan Adopsi Teknologi (INOTEK)*, Nov. 2021, pp. 246–260.

[16] I. M. Sukarsa, I. K. A. M. Antara, P. W. Buana, I. P. A. Bayupati, N. W. Wisswani, and D. W. Puteri, "Data Storage Model in Low-cost Mobile Applications," *Indonesia Journal of Electrical Engineering and Computer Science*, vol. 28, no. 1, pp. 1128–1138, Nov. 2022.

[17] C. A. Győrödi, D. V Dumşe-Burescu, D. R. Zmaranda, and R. Ş. Győrödi, "A Comparative Study of MongoDB and Document-Based MySQL for Big Data Application Data Management," *Big Data and Cognitive Computing*, vol. 6, no. 2, 2022, doi: 10.3390/bdcc6020049.

[18] M. Bender, E. Kirdan, M.-O. Pahl, and G. Carle, "Open-Source MQTT Evaluation," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, Las Vegas, NV, USA, 2021, pp. 1–4.

[19] I. Darmawan, M. S. Anwar, A. Rahmatulloh, and H. Sulastri, "Design Thinking Approach for User Interface Design and User Experience on Campus Academic Information Systems," *JOIV International Journal on Informatics Visualization*, vol. 6, no. 2, pp. 327–334, Jun. 2022.

[20] E. Nurhayati and Agussalim, "Rancang Bangun Back-end API pada Aplikasi Mobile AyamHub Menggunakan Framework Node JS Express," *JUSTIN (Jurnal Sistem dan Teknologi Informasi)*, vol. 11, no. 3, pp. 524–531, Jul. 2023.

[21] S. Nidhra and J. Dondeti, "Black Box and White Box Testing Techniques –A Literature Review," *International Journal of Embedded Systems and Applications (IJESA)*, vol. 2, no. 2, pp. 29–50, 2021.

[22] M. Hendayun, A. Ginanjar, and Y. Ihsan, "Analysis of Application Performance Testing Using Load Testing and Stress Testing Methods in API Service," *Jurnal Sisfofek Global*, vol. 13, no. 1, pp. 28–34, Mar. 2023.