Volume XI Issue 3 Year 2026 | Page 864-875 | ISSN: 2527-9866
Received: ***30-06-2026*** | *Revised:* ***25-07-2025*** | *Accepted:* ***11-08-2026***

Development of a Layered Architecture for a Single-Farm Fish Farming Management System Using Prototype Method

Danuri[1], Jaroji[2]
[1,2] Politeknik Negeri Bengkalis, Bengkalis, Riau, Indonesia
e-mail: danuri@polbeng.ac.id[1], jaroji@polbeng.ac.id
*Correspondence: danuri@polbeng.ac.id

Abstract: *Small-scale fish farming businesses often experience challenges in managing the fragmented operational data which includes cultivation activities, inventory, transactions and reporting processes. Lack of an integrated information management structure restricts the traceability of data and reduces the efficiency of operations. Unlike prior digital aquaculture systems, which mainly target environmental monitoring, this study designs a layered architecture for the web-based single-farm fish farming management system using the Prototype method, with novelty in the architecture-level integration of five system layers with four business domains rather than in either method individually. The research methodology consists of requirement analysis, Software Requirements Specification development, layered architecture design, prototype development, system implementation, and functionality verification. The proposed architecture separates system components into five logical layers: presentation, application, data access, data and cache layers, and decomposes business functions into production, inventory, financial and reporting domains. The developed system comprises eight functional modules that enable the fish farming management processes from end to end. The functional verification was done by black-box testing with nine test scenarios, which had 100% success rate. The results indicate that the proposed layered architecture provides a structured basis for integrating cultivation and business processes in a single-farm fish farming environment; maintainability and scalability follow from the design but were not empirically measured. Future development may extend the architecture with multi-farm management, Internet of Things integration and quantitative evaluation of software quality attributes.*

Keywords: *layered architecture, prototype method, fish farming management system.*

1. Introduction

Digital transformation has increasingly influenced the various sectors including aquaculture through the adoption of information systems, Internet of Things (IoT), cloud computing and data-driven technologies. These technologies enable operational efficiency by collecting data, monitoring and making decisions in a systematic way. Aquaculture activities are complex and interrelated processes, such as pond management, growing cycles, feeding, monitoring, recording mortality and growth, harvesting and sales [1], [2]. Therefore, aquaculture management systems should not only provide data recording capabilities, but also an architecture to integrate the different operational and business processes.

At the farm level, operational data are often managed manually or through separate applications, causing fragmentation among pond records, cultivation cycles, feed usage, inventory, harvest data, sales transactions, and reporting [3]. This condition limits traceability of data and increases the risk of inconsistent information. In previous studies, digital aquaculture systems have been developed, especially IoT-based solutions for water quality monitoring, environmental control and intelligent decision support [4]. However, most of the current research is on environmental monitoring, and little is on the integrated management systems that combine cultivation activities with business processes.

The development of integrated information systems requires a software architecture that supports modularity, maintainability, and future system evolution. One way to organise software components into independent layers according to their responsibilities is the layered architecture that allows better separation of concerns and scalability of the system [5], [6]. Moreover, the Prototype method offers an iterative development approach that allows incremental refinement of complex information systems with changing requirements [7], [8].

Therefore, this study aims to develop a layered architecture for a web-based single-farm fish farming management system using the Prototype method. The proposed system integrates cultivation management, daily monitoring, inventory, procurement, harvesting, sales, receivables, and reporting processes into a centralized platform. Unlike prior digital aquaculture systems, which mainly target environmental monitoring and automation [4], [9], [10], [11], [12], the main contribution of this study is an architecture model that explicitly maps single-farm aquaculture business processes onto structured software layers, together with the validation of the implemented system by functional testing.

2. Literature Review

2.1 Digital Fish Farm Management System

Digital fish farm management systems use information technology to integrate aquaculture production activities, operational data and business processes into a single centralized platform. They turn fragmented manual records into structured digital information, which enables more efficient data management, operational monitoring and decision making [13]. Digital technologies, and in particular Internet of Things (IoT)-based solutions have been previously shown to improve aquaculture performance by monitoring environmental parameters including temperature, dissolved oxygen, pH and water quality [9].

However, existing digital aquaculture systems predominantly focus on environmental monitoring and automated control, while integrated management of cultivation cycles, inventory, harvesting, sales, and financial processes remains limited [14]. A comprehensive fish farm management system should provide not only monitoring capabilities but also operational traceability across the aquaculture lifecycle [15]. Therefore, integrating production activities with business processes is essential to establish a more complete management framework that supports efficient and data-driven aquaculture operations. However, existing integrated aquaculture systems [3], [13], [14], [15] target enterprise, group, or cage-based contexts and lack an explicit layered architecture for combined cultivation and business operations at a single farm, a gap this study addresses.

2.2 Software Architecture for Integrated Information Systems

Software architecture defines software components, their interactions and the separation of responsibilities and is of fundamental importance for guaranteeing the scalability, maintainability and extensibility of information systems [16]. The layered architecture is one of the most popular architectural patterns where systems are built in layers such as presentation, business logic, and data access [17]. This pattern increases the modularity and decreases the dependency among components.

An agricultural and aquaculture information system needs a well-defined architecture to integrate production, inventory, financial and reporting processes into one platform [1]. However, the existing studies mainly focus on the enterprise-scale farm management or IoT-based monitoring, and little attention is paid to the architectural designs of the farm-tailored aquaculture management for a single farm. Thus, there is still a need for a layered architecture adapted to the operational characteristics and business processes of the single-farm fish farming.

2.3 Prototype Method for Software Development

Software development methodologies are important for converting system requirements into working applications. Among them, the Prototype method is widely used in the development of information systems with evolving requirements, because it can be refined through iterative processes of requirement analysis, prototype construction, evaluation and implementation. This allows early validation of the system functionality and reduces the risk of requirement mismatches [18].

Previous studies have demonstrated the effectiveness of the Prototype method in developing information systems with complex business processes [19]. However, most studies employ Prototype primarily as a software development process, with limited attention to its integration with software architecture design. This research applied the Prototype development method. The main contribution of this research is to design and develop a layered architecture for a web-based single-farm fish farming management system.

2.4 Research Gap and Contribution

Previous studies about digital aquaculture mainly focus on IoT-based monitoring and automation [4], [10], [11], [12], while integrated management of cultivation and business processes is less investigated. Moreover, the Prototype method has been widely applied to software development but has not yet been applied to the design of a structured layered architecture for single-farm aquaculture management systems. To fill this gap, this study develops a layered architecture for web-based single-farm fish farming management system using Prototype method. The proposed architecture integrates the major functional modules such as cultivation management, inventory, procurement, harvesting, sales, receivables and reporting into an integrated platform. Unlike prior systems targeting enterprise or multi-farm operations, it is intended as a maintainable and scalable foundation tailored to single-farm management, validated here through functional testing rather than empirical quality measurement.

3. Methods

A. Research Design

This study employed a Design and Development Research (DDR) approach using the Prototype method to develop a web-based single-farm fish farming management system. The research focused on designing a layered software architecture and implementing it into a functional information system. The development process consisted of six sequential stages: Requirement Analysis, Software Requirements Specification (SRS), Layered Architecture Design, Prototype Development, System Implementation, and Functional Testing, as illustrated in Figure 1. This approach was selected because it enables systematic software development while supporting iterative refinement throughout the implementation process.

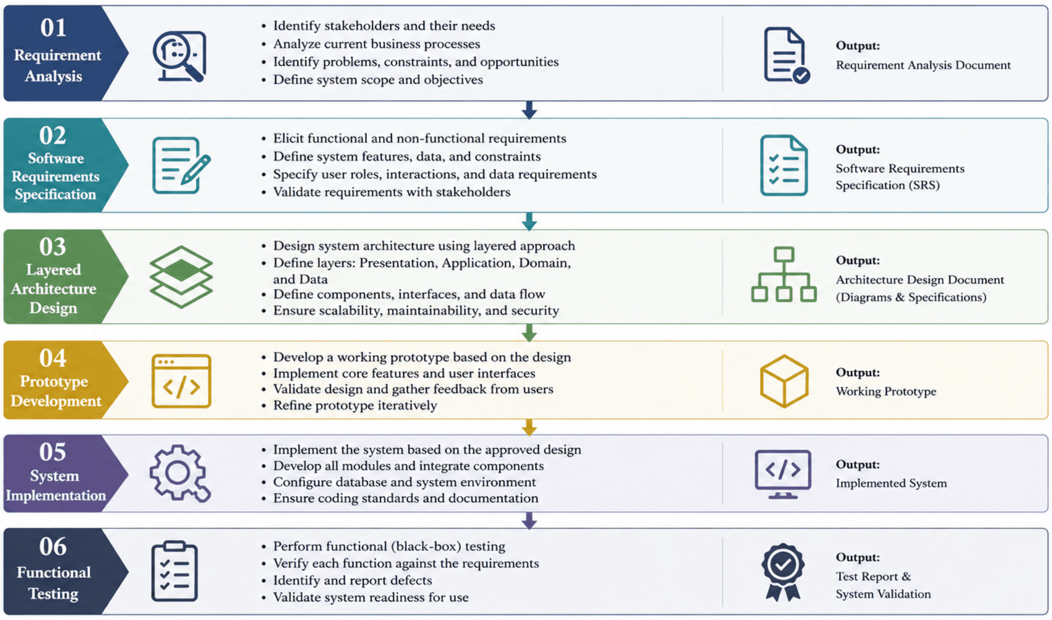

Figure 1. Research Framework

B. Requirement Analysis

The requirement analysis phase was performed to identify and define the system requirements based on the operational processes of single-farm fish farming management. In this phase we analyzed current business workflows and challenges, reviewed information requirements and defined functional needs for supporting cultivation and business activities. The identified requirements were systematically analyzed to establish the system boundaries, user interactions, and relationships among operational processes. The analysis process focused on the understanding of the way the cultivation activities and business transactions are performed in a single-farm environment to ensure that the developed system components appropriately represent the required operational workflows.

The results of this stage were used as the foundation for defining the system specification, designing the software architecture, and developing the prototype. This ensures consistency of the proposed system with the operating characteristics and management requirements of a single-farm fish farming operation.

C.Software Requirements Specification (SRS)

The identified prerequisites were systematically documented in a Software Requirements Specification (SRS). This led to the development of an orderly structure for the software development process. The fish farming management system version 3.1 SRS document describes functional and non-functional requirements, system stakeholders, business regulations, module interaction, use case narratives, system architecture, and API specifications.

The designated criteria were subsequently evaluated and modified to conform to the research parameters through the application of a single-farm operational model (Solo Mode), indicating a singular business entity managed by two user roles, namely Owner and Staff, without multi-tier approval mechanisms. The refined SRS served as the foundation for the subsequent multi-tiered architectural design and prototype development to verify that the implemented system met the specified operational standards and research objectives.

D. Layered Architecture Design

The software architecture was designed according to the Layered Architecture pattern to provide a structured separation of the system's responsibilities. The design process was focused on the definition of architectural boundaries, the assignment of responsibilities to components and the definition of interactions among layers of the system based on the identified requirements.

The architecture design considered software quality attributes, including modularity, maintainability, scalability, and separation of concerns. The proposed structure consists of presentation, application, data access, data, and cache layers, with business functionalities organized according to the operational domains of the fish farming management system. The resulting architecture model was subsequently used as the foundation for prototype implementation.

E. Prototype Development

Subsequent to the architectural design, the software development was executed utilizing the Prototype methodology. The development procedure has been executed in six iterative phases: requirement validation, rapid design, prototype construction, prototype assessment, prototype enhancement, and ultimately implementation. Each functional module was developed incrementally following the proposed layered architecture and then tested before being integrated to the overall system. Through this iterative approach, system performance was continuously validated, requirements were refined, and potential implementation risks were mitigated throughout the software development life cycle.

F. System Implementation

The last prototype was implemented as a web-based fish farming management system based on the proposed Layered Architecture. We developed the Presentation Layer with Next.js for the web and Flutter for mobile and Application Layer with NestJS for business logic and RESTful API services. Operational data were stored using PostgreSQL as the main database, while Redis was used to support session management and caching mechanisms. The technology implementation for each architectural layer is summarized in Table 1.

Table 1. Layer Technology Mapping

No	Layer	Technology	Responsibility
1	Presentation	Next.js, Flutter	User Interface
2	Application	NestJS	Business Services
3	Data Access	TypeORM	Database Communication
4	Data	PostgreSQL	Data Storage
5	Cache	Redis	Cache and Session

The implemented system integrates eight functional modules derived from the Software Requirements Specification (SRS), including Pond and Cultivation Cycle Management, Daily Monitoring, Inventory, Procurement, Harvest, Sales, Accounts Receivable, and Reporting. These modules were organized according to the defined business domains and integrated within the layered architecture to provide a centralized platform for managing single-farm fish farming operations.

G. Functional Testing

The functional validation was performed by applying Black-box Testing to verify if the functionalities of the developed system were compliant with the requirements defined in the Software Requirements Specification (SRS). The test scenarios have been systematically derived from the functional requirements of each module without considering the internal implementation details of the software.

Each test case consisted of the tested module, test scenario, expected output, actual output, and validation result. The functional success rate was calculated using Equation (1):

$$Success\ Rate\ (\%) = \frac{Passed\ Test\ Cases}{Total\ Test\ Cases} \times 100 \tag{1}$$

A functional module was considered successfully validated when all predefined test scenarios generated outputs consistent with the expected system behavior. This evaluation helps to ensure that the developed system satisfies the functional requirements specifications and operates as per the designed architecture.

4. Results and Discussion

A. Requirement Analysis Results

The system scope was defined based on the requirement analysis as the operational characteristics of a single-farm fish farming environment. The derived specification identifies two main actors of the system: the Owner and the Staff. The system interacts with them according to their operational responsibilities.

The identified requirements were organized into functional modules and grouped into four business domains: Production, Inventory, Financial, and Reporting. The Production domain refers to activities such as pond management, cultivation cycles, monitoring and harvesting. The Inventory domain covers procurement of feed resources and the financial domain merges sales and accounts receivable. The Reporting domain provides summarized information from operational and transactional data.

The relationship between operational problems, system requirements, software modules, and business domains is summarized in Table 2.

Table 2. Requirement Identification

No	Identified Problems	System Requirements	Software Modules	Business Domain
1	Fragmented management of pond and cultivation cycle data	Centralized management of pond and cultivation cycle information	Pond & Cultivation Cycle	Production
2	Manual recording of daily operational activities	Digital recording and monitoring of daily cultivation activities	Daily Monitoring	Production
3	Lack of integrated feed inventory management	Integrated feed inventory and stock management	Inventory	Inventory
4	Unstructured procurement process	Structured purchase order and procurement management	Procurement	Inventory
5	Harvest records are not integrated with cultivation cycles	Integrated harvest management linked to cultivation cycles	Harvest	Production
6	Sales transactions are not integrated with harvest records	Integrated sales transaction management	Sales	Financial
7	Limited monitoring of accounts receivable	Real-time accounts receivable management	Accounts Receivable	Financial
8	Manual generation of operational reports	Automated integrated reporting	Reporting	Reporting

B. Layered Architecture Model

Figure 2 shows the proposed layered architecture model. The architecture gives a modular view of the developed system by dividing the software components into five logical layers: Presentation Layer, Application Layer, Data Access Layer, Data Layer and Cache Layer.

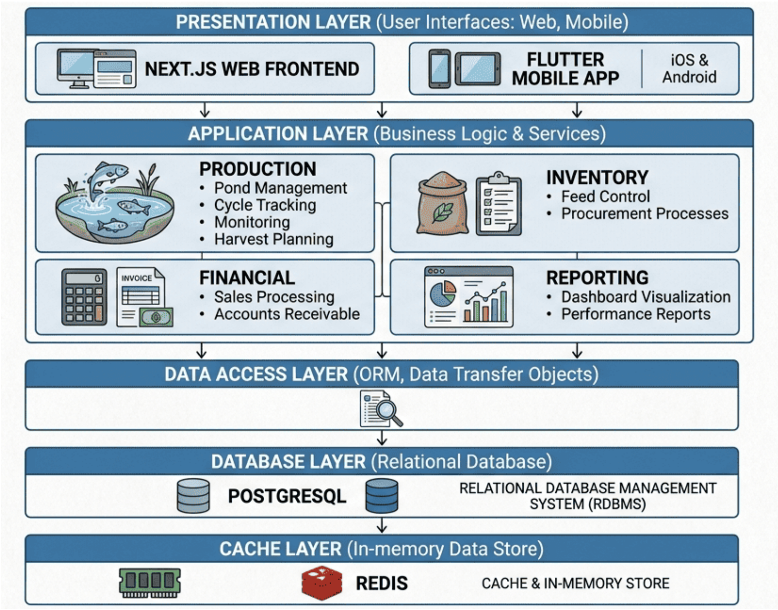

Figure 2. Proposed Layered Architecture

The Presentation Layer represents the interaction interface between users and the system through web and mobile applications. The Application Layer contains business services that execute fish farming management processes. The Data Access Layer provides controlled access of the application services to the database. The Data Layer contains persistent operation information. Cache Layer: Cache Layer helps in improving the system performance by caching the data and tracking the session.

Beyond the conventional layered structure, the proposed model incorporates four business domains to represent the characteristics of single-farm fish farming operations: Production, Inventory, Financial, and Reporting. These domains organize in a logical way the functional modules coming from the system requirements, creating traceability links between the operational processes and the software components.

C. System Implementation Results

The system is implemented as a web application accessible via both desktop and mobile browsers. The interface uses the label "Fish Farming - Solo Mode" to indicate the single-scale operational context. Below is an overview of the key modules generated.

Dashboard

The dashboard presents a real-time operational summary: the number of active ponds, ongoing cycles, feed stock (kg), outstanding receivables, sales for the current month, low-stock alerts, the most recent harvest, and the latest monitoring data. The dashboard enables managers to monitor business conditions without needing to open each module individually. Figure 3 shows the dashboard view.

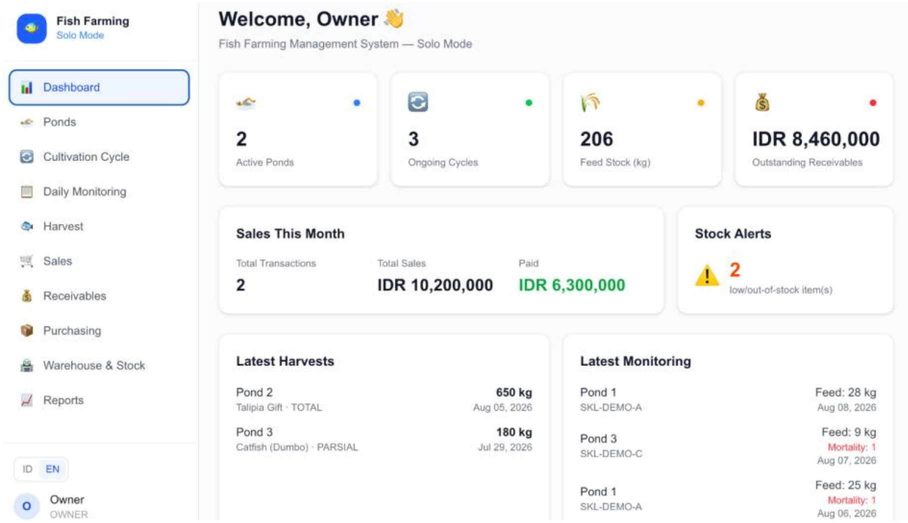

Figure 3. Dashboard fish farming – solo mode

Pond Management and Cultivation Cycle.

The pond module displays a list of ponds along with their operational status and associated active cycles. Each pond entry shows the code, name, type (tarpaulin/earthen/concrete/biofloc), area, capacity, and the name of the ongoing cycle, including the current day of the cycle. The cycle module displays cultivation details-such as commodity, stocking quantity, start date, and estimated harvest date-and offers filtering options based on status. Figure 4 shows the pond management and cultivation cycle.

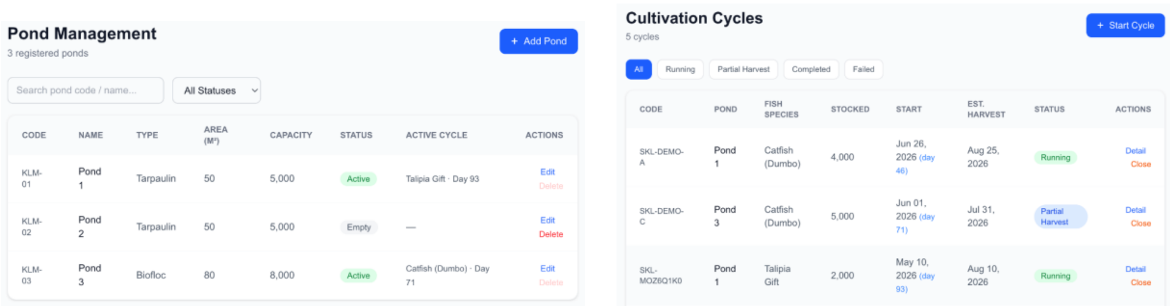

Figure 4. Pond Management (left) and Cultivation Cycle (right) views.

Daily Monitoring.

The monitoring module provides a daily input form with four tabs: Feed, Mortality, Sampling, and Water Quality. Users select the active cycle and date, then enter data based on field conditions. Monitoring history is displayed in the right panel, showing a summary of feed (kg), mortality (count), pH, temperature, and DO by date. Figure 5 shows the daily monitoring view.

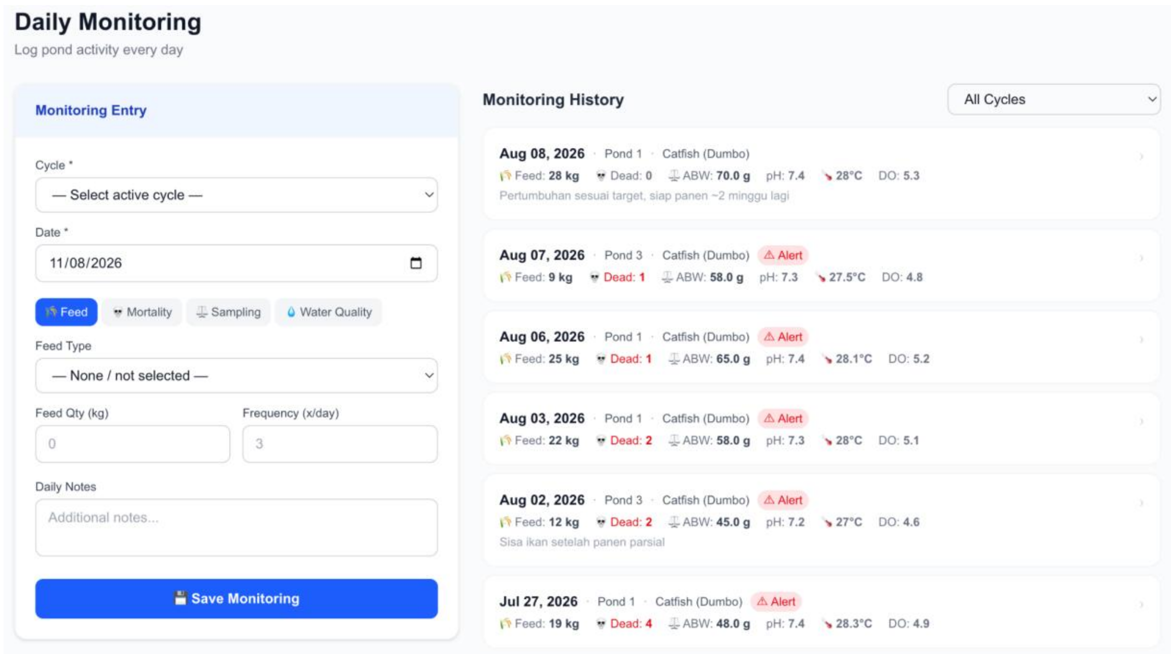

Figure 5. Daily monitoring view with input form and history.

Warehouse, Stock, and Purchasing.

The warehouse module displays the stock status of each item using color-coded indicators: Out of Stock (red), Critical (orange), and Normal (green). The system automatically detects items with stock levels below the minimum threshold and displays a warning badge. The purchasing module records purchase orders (POs) sent to suppliers; in Solo Mode, POs are immediately set to "Approved" status without requiring a separate approval step. Figure 6 shows the purchase view.

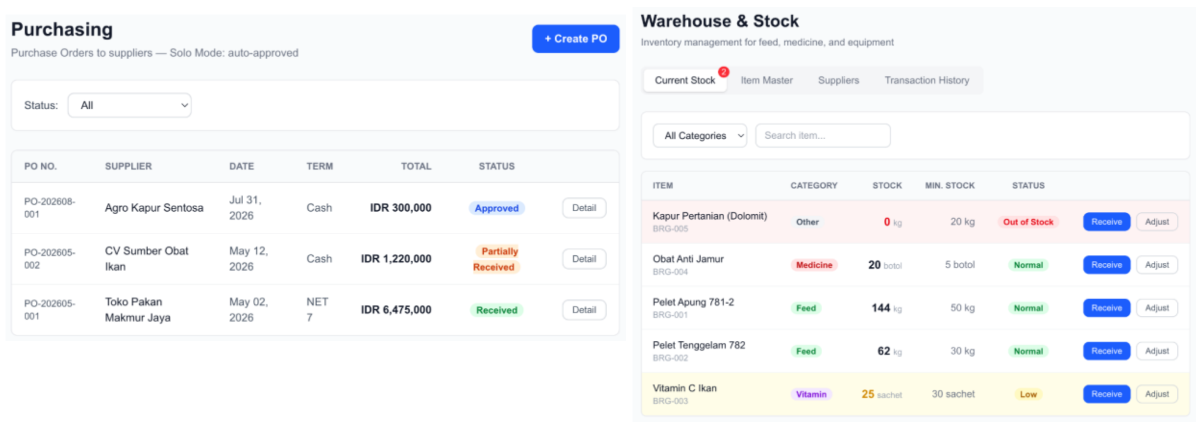

Figure 6. Purchase/PO and warehouse views.

Harvest, Sales, Accounts Receivable, and Reports.

The harvest module records production output linked to an active cycle. Harvested produce becomes inventory available for sale via the sales module. The accounts receivable module displays customer receivable aging across four categories: not yet due, 1–60 days overdue, and >60 days overdue. The reports module presents a production summary per cycle, featuring columns for: commodity, days, seed, feed (kg), harvest (kg), FCR, and SR (%). Figure 7 shows the report view.

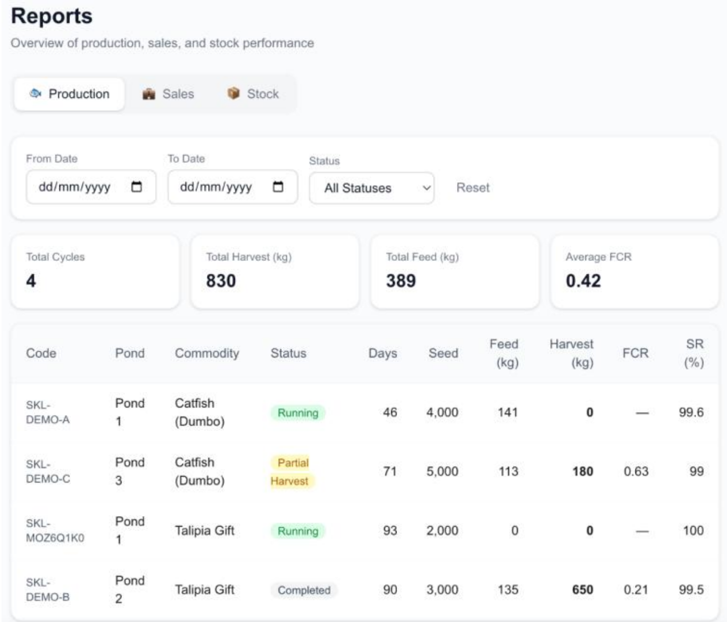

Figure 7. Report View

4.4 Functional Verification Results

The implemented system was tested functionally to check the compliance of the functional requirements mentioned in the Software Requirements Specification (SRS). The results of the nine test cases for the core functional modules are summarized in Table 3 and all the test cases were successfully completed. The system met the operational requirements specified and worked as expected.

Table 3. Functional Verification Results Using Black-Box Testing

No.	Module	Test Scenario	Expected Output	Actual Output	Status
1	Login	Input valid username and password	Dashboard opens according to user role	Dashboard opened as Owner	Passed
2	Pond	Add a new pond	Pond data are stored and displayed	Pond data successfully stored	Passed
3	Cultivation Cycle	Create a new cultivation cycle	Active cultivation cycle is created	Cycle created and pond status updated	Passed
4	Daily Monitoring	Record daily monitoring data	Monitoring data stored and feed stock updated	Data stored and inventory updated	Passed
5	Inventory	Receive goods from purchase order	Inventory quantity and stock status updated	Inventory updated successfully	Passed
6	Procurement	Create a purchase order	Purchase order stored with approved status	Purchase order successfully created	Passed
7	Harvest	Record harvesting activity	Harvest data stored and harvest stock created	Harvest data successfully recorded	Passed
8	Sales	Record a sales transaction	Sales transaction stored and receivable generated	Sales and receivable successfully recorded	Passed
9	Reporting	Generate production report	Production report displayed correctly	Report generated successfully	Passed

4.5 Discussion

The proposed layered architecture integrates the processes of cultivation and business management in a single software architecture for fish farming at the level of one farm. The architecture divides the system into five logical layers and clusters the business logic into four operational domains. This provides a clear separation of responsibilities and creates traceability between the system requirements and the software components, which is consistent with design principles generally associated with improved maintainability, although maintainability itself was not directly measured in this study. The implemented and functionally verified results presented demonstrate that the proposed architecture is capable of supporting end-to-end operational workflows and satisfies all the specified functional requirements; however, functional correctness alone does not constitute evidence of maintainability, scalability, or overall software quality.

Compared with previous studies such as [4], [9], [10], [11], [12] which primarily focus on environmental monitoring or isolated aquaculture applications, and with broader agri-food information system studies [3], [13], [14], [15] that address integration at an enterprise or multi-farm scale, this research contributes an architecture-oriented approach that integrates cultivation and business processes specifically for single-farm operations. Nevertheless, the proposed architecture was evaluated only in a single-farm operational environment using functional black-box verification, without quantitative assessment of response time, performance, usability, or modularity/maintainability metrics. Consequently, the maintainability and scalability of the architecture are grounded in the design principles applied-layer separation and domain-based decomposition-rather than in empirical measurement, which is a practical limitation of the present study. Future research may extend the architecture to support multi-farm management, IoT integration, and quantitative evaluation of architectural quality attributes, such as scalability, maintainability, and performance.

5. Conclusions

In this study, the Prototype method was used to propose a layered architecture for a web-based single-farm fish farming management system. The proposed architecture is initiated by the determination of operational requirements and structures the system functionalities into five logical layers and four business domains, offering a clear separation of responsibilities and traceability between business requirements and software components. The architecture was successfully deployed as a web-based management system that integrated cultivation, inventory, financial and reporting processes in a centralized platform. Functional verification using black-box testing demonstrated that all functional modules operated according to the specified requirements, achieving a 100% success rate across all test scenarios. The main contribution of this study is a layered architecture that maps single-farm aquaculture business processes onto five structured software layers and four business domains, offering a feasible foundation for developing integrated fish farming management systems in single-farm operational environments. Its maintainability and scalability are supported by the applied architectural design principles but were not empirically validated in this study, which is a limitation to be addressed in future work. Future research may extend the proposed architecture by supporting multi-farm management, integrating Internet of Things (IoT) technologies for real-time monitoring, and empirically validating architectural quality attributes, such as scalability, maintainability, and performance, using quantitative software engineering metrics.

References

[1] H.-Y. Lan, N. A. Ubina, S.-C. Cheng, S.-S. Lin, and C.-T. Huang, “Digital Twin Architecture Evaluation for Intelligent Fish Farm Management Using Modified Analytic Hierarchy Process,” *Applied Sciences*, vol. 13, no. 1, p. 141, Dec. 2022, doi: 10.3390/app13010141.

[2] V. Biazi and C. Marques, “Industry 4.0-based smart systems in aquaculture: A comprehensive review,” *Aquacultural Engineering*, vol. 103, p. 102360, Nov. 2023, doi: 10.1016/j.aquaeng.2023.102360.

[3] E. Ahoa, A. Kassahun, C. Verdouw, and B. Tekinerdogan, “Challenges and Solution Directions for the Integration of Smart Information Systems in the Agri-Food Sector,” *Sensors*, vol. 25, no. 8, p. 2362, Apr. 2025, doi: 10.3390/s25082362.

[4] M. A. Subandri and D. Danuri, “Early Warning System Of Water Quality Changes In Fishponds,” *JTOS*, vol. 6, no. 1, pp. 55–62, Jun. 2023, doi: 10.36378/jtos.v6i1.3109.

[5] B. Soongpol, P. Netinant, and M. Rukhiran, “Practical Sustainable Software Development in Architectural Flexibility for Energy Efficiency Using the Extended Agile Framework,” *Sustainability*, vol. 16, no. 13, p. 5738, Jul. 2024, doi: 10.3390/su16135738.

[6] M. Broy and B. Selić, “Specifying and Composing Layered Architectures.,” *JOT*, vol. 23, no. 1, p. 1:1, 2024, doi: 10.5381/jot.2024.23.1.a2.

[7] M. Rukhiran, S. Buaroong, and P. Netinant, “Software Development for Educational Information Services Using Multilayering Semantics Adaptation:,” *International Journal of Service Science, Management, Engineering, and Technology*, vol. 13, no. 1, pp. 1–27, Sep. 2022, doi: 10.4018/IJSSMET.307153.

[8] A. Perisic and B. Perisic, “Towards a Digital Transformation Hyper-Framework: The Essential Design Principles and Components of the Initial Prototype,” *Applied Sciences*, vol. 15, no. 2, p. 611, Jan. 2025, doi: 10.3390/app15020611.

[9] Md. N. I. Nayoun, S. A. Hossain, K. M. Rezaul, K. N. E. A. Siddiquee, Md. S. Islam, and T. Jannat, “Internet of Things-Driven Precision in Fish Farming: A Deep Dive into Automated Temperature, Oxygen, and pH Regulation,” *Computers*, vol. 13, no. 10, p. 267, Oct. 2024, doi: 10.3390/computers13100267.

[10] C.-H. Chen, Y.-C. Wu, J.-X. Zhang, and Y.-H. Chen, “IoT-Based Fish Farm Water Quality Monitoring System,” *Sensors*, vol. 22, no. 17, p. 6700, Sep. 2022, doi: 10.3390/s22176700.

[11] M. A. Abid, M. Amjad, K. Munir, H. U. R. Siddique, and A. D. Jurcut, “IoT-Based Smart Biofloc Monitoring System for Fish Farming Using Machine Learning,” *IEEE Access*, vol. 12, pp. 86333–86345, 2024, doi: 10.1109/ACCESS.2024.3384263.

[12] E. E. Malandrakis, “Design and Implementation of a Cost-Effective IoT-Based Monitoring and Alerting System for Recirculating Aquaculture Systems (RAS),” *Sensors*, vol. 25, no. 21, p. 6692, Nov. 2025, doi: 10.3390/s25216692.

[13] H. Ihsan, R. Efendi, and E. Yandani, “Digital Transformation in Fisheries Cultivation: Catfish Farmer Group Management Information System: Transformasi Digital dalam Budidaya Perikanan: Sistem Informasi Manajemen Kelompok Pembudidaya Lele,” *JVEIT*, vol. 5, no. 2, pp. 62–75, Dec. 2024, doi: 10.56667/jveit.v5i2.1570.

[14] M. N. N. Norzeri, S. F. Mat Yatin, and A. Z. H. Samsudin, “Information Management Monitoring System for Fish Cage Farming,” *E-BPJ*, vol. 10, no. SI27, pp. 149–153, May 2025, doi: 10.21834/e-bpj.v10iSI27.6926.

[15] Tedyyana, A., Ratnawati, F., & Danuri, D. (2023). Platform monitoring berbasis web untuk sistem stabilitas dan pelacakan kapal nelayan. *Digital Zone: Jurnal Teknologi Informasi dan Komunikasi*, *14*(2), 168-178..

[16] S. B. A. Guetat and S. B. D. Dakhli, “A multi-layered software architecture model for building software solutions in an urbanized information system,” *IJISPM*, vol. 1, no. 1, pp. 19–34, Feb. 2022, doi: 10.12821/ijispm010102.

[17] L. Chamari, E. Petrova, and P. Pauwels, “An End-to-End Implementation of a Service-Oriented Architecture for Data-Driven Smart Buildings,” *IEEE Access*, vol. 11, pp. 117261–117281, 2023, doi: 10.1109/ACCESS.2023.3325767.

[18] E. Bjarnason, F. Lang, and A. Mjöberg, “An empirically based model of software prototyping: a mapping study and a multi-case study,” *Empir Software Eng*, vol. 28, no. 5, p. 115, Sep. 2023, doi: 10.1007/s10664-023-10331-w.

[19] S. Perera, A. A. Hijazi, G. T. Weerasuriya, S. Nanayakkara, and M. N. N. Rodrigo, “Blockchain-Based Trusted Property Transactions in the Built Environment: Development of an Incubation-Ready Prototype,” *Buildings*, vol. 11, no. 11, p. 560, Nov. 2021, doi: 10.3390/buildings11110560.