



Design and Construction of a Sales Information System Using the Reactjs and Expressjs Frameworks: case study of Fa_al.store

Kevin Yohanes Wuryanto¹, Agung Brastama Putra², Nur Cahyo Wibowo³

^{1,2,3}Universitas Pembangunan Nasional "Veteran" Jawa Timur, Indonesia

Article Info

Article history:

Received 09 30, 2025

Revised 10 25, 2025

Accepted 11 28, 2025

Keywords:

Sales Information System

ReactJS

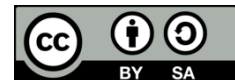
ExpressJS

MSME

ABSTRACT

Fa_al.store is a micro, small, and medium enterprise (MSME) that sells socks both online and offline. In its operational activities, the process of recording transactions and managing stock is still done manually, so the business owner experiences difficulties in monitoring product availability, assessing sales performance, and preparing periodic reports. This research aims to design and build a web-based sales information system that can facilitate transaction recording, product stock management, customer data, promotions, and presenting sales reports systematically and structured. The development of this system applies the Waterfall method, which includes the stages of needs analysis, design, implementation, and testing. On the frontend side, the ReactJS framework is used to build an interactive user interface, while the backend is developed using ExpressJS to handle business logic and communication with the database. This system is also integrated with the Midtrans payment gateway to support a secure and efficient online payment process. The results of testing using the User Acceptance Testing (UAT) method show that the system has been able to meet the functional needs of users with an acceptance rate of 94.2%, which indicates that the developed system is feasible to use and can help business actors in carrying out business processes effectively.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Kevin Yohanes Wuryanto

Department of Information Systems, Faculty of Computer Science

Universitas Pembangunan Nasional "Veteran" Jawa Timur, Indonesia

Email: kevinyohanesteen@gmail.com

© The Author(s) 2025

1. Introduction

Micro, Small, and Medium Enterprises (MSMEs) are a crucial foundation of the Indonesian economy. According to data from the Ministry of Cooperatives and SMEs, this sector contributes more than 60% to the nation's Gross Domestic Product (GDP) and plays a significant role in absorbing the country's workforce [1]. This significant contribution demonstrates that MSMEs play a highly strategic role in maintaining economic stability, expanding employment opportunities, and driving economic growth across various sectors [2]. Therefore, increasing the competitiveness of MSMEs is crucial for their survival amidst increasingly fierce market competition.

Despite their significant role, MSMEs still face several challenges in the digitalization process. One of these is limited understanding and mastery of technology among business owners, resulting in manual operational processes [3][4]. Sales recording, inventory management, and financial reporting are often poorly documented, potentially leading to errors. However, in the era of digital transformation, the use of

information technology is a key factor in increasing business efficiency. This change also aligns with the consumer shopping trend, which is increasingly shifting towards digital platforms.

Fa_al.store is an MSME specializing in the sale of socks in various designs and sizes. The products are of high quality and competitively priced. Sales are conducted in two ways: online through marketplaces like Shopee, and offline through face-to-face meetings between buyers and the owner. All operational activities at Fa_al.store are managed by the owner, with the assistance of family members, particularly in the packaging and offline sales processes. Fa_al.store faces several challenges, including manual and unstructured stock management, manual transaction recording, scattered inventory storage without a systematic grouping system, and manual revenue calculations. These challenges make it difficult to ensure stock availability and optimally monitor sales performance.

A sales information system is a system designed to support digital sales management processes via the internet [5]. This system functions as a medium that integrates various business activities, from recording transactions, managing product and stock data, to preparing sales reports that can be accessed in real time [6]. This system can be accessed through a browser, making it easier for business owners to flexibly monitor sales performance while also providing customers with the convenience of conducting online transactions anywhere.

Implementing a sales information system is a relevant solution to address the challenges faced by Fa_al.store. This system is expected to support automatic transaction recording, more organized product inventory management, and provide easily accessible revenue reports. With sales information, MSME operational processes will become more efficient, allowing owners to conduct quick and accurate business evaluations. Furthermore, the sales information system also helps increase data transparency, which is crucial for business sustainability.

This sales information system utilizes the ReactJS framework on the frontend, based on its ability to build dynamic, responsive, and easily maintainable user interfaces through the component concept [7] [8]. ReactJS also has a broad ecosystem, facilitating integration with other supporting libraries [9]. Meanwhile, ExpressJS was chosen as the backend framework due to its lightweight, flexible nature, and strong performance for managing business logic and communicating with databases [8], [10]. The combination of ReactJS and ExpressJS enables the development of a fast, scalable, and structured sales information system, meeting the website's needs to provide a modern and efficient transaction experience for both users and admins.

The system was developed using the Waterfall method, which offers a structured approach at each stage [10]. This method encompasses five main stages: requirements analysis, design, implementation, testing, and maintenance, with each stage being thoroughly completed before moving on to the next [11]. This system is expected to simplify transaction management and reporting, while providing data that can help support better business decision-making. To support system development, the ReactJS framework is used on the frontend to create an interactive user interface, and ExpressJS on the backend to manage business logic and communicate with the database.

With this sales information system, Fa_al.store is expected to overcome operational challenges it has faced, improve business management efficiency, and strengthen its competitiveness in the digital marketplace. Furthermore, this research is expected to provide practical benefits to other MSMEs facing similar challenges and serve as a reference for utilizing information technology to drive small business growth in Indonesia.

2. Research Method

According to Roger S. Pressman, the Waterfall method is a sequential software development model, where each stage, from requirements analysis, design, coding, testing, to maintenance, is completed in a gradual and linear manner. Each phase has a clear focus and documentation, allowing for good quality control before proceeding to the next stage [12]. The following stages include the process of data collection, design, and system testing. The flow of this research work is depicted through the flowchart in Figure 1.

2.1. Metode Waterfall

The Waterfall method is a software development model that uses systematic and sequential stages [13]. In this model, the process begins with the planning stage and continues linearly through analysis, design, coding, testing, and support or maintenance. This model provides systematic steps, making it suitable for projects with clearly defined requirements from the outset. Each stage must be completed before the next stage begins, providing clarity in the work sequence [13].

While this model has a clear structure, it has several drawbacks. One is the inability to handle change flexibly, as each stage must be completed before the next stage begins [13]. This makes it more difficult for projects to adapt to changing requirements that arise mid-development. Furthermore, users must wait until the end of the project to see the final results, which can slow down the introduction of feedback. This inflexibility often leads to project teams experiencing a "blocking state," where some team members must wait until others complete interdependent tasks [13]. The Waterfall method has several main stages, which are explained below:

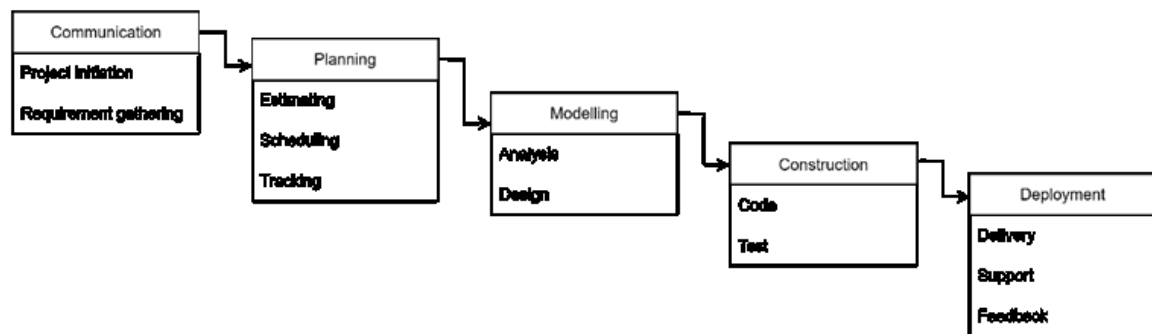


Figure 1. Waterfall method

1. *Communication*

This stage is the initial step in system development, where project initiation and gathering of needs from users or stakeholders are carried out. The goal is to clearly understand what users need for the system to be built [12].

2. *Planning*

After the requirements are collected, the planning stage is carried out. This stage includes analyzing the requirements and preparing the system specifications for the research. Careful planning will minimize the risk of errors and delays in the next stage [12].

3. *Modelling*

At this stage, the analyzed requirements will be expressed in the form of a system model, UML, database design, and user interface design. The aim is to provide a clearer picture of how the system will be built and operated [12].

4. *Modelling*

This stage is the process of implementing or building a system based on a previously created model. The developer will write program code and conduct testing to ensure that the system functions according to specifications [12].

5. *Deployment*

The final stage is the implementation of the system into the user environment. The completed system will be installed (deployed) and put into actual use. In addition, this stage also includes the maintenance and repair process if problems are found during use [12].

3. Result and Discussion

3.1. *Communication*

1. *Problem Analysis*

Fa_al.store is an MSME that sells socks. In procuring stock, Fa_al.store maintains simple inventory records using a ledger, then stores the items in a warehouse without any organized classification, spread across two rooms and a motorcycle shed. This results in disorganized stock management, relying solely on the owner's memory, making it difficult to track stock levels. Online sales are conducted through marketplaces like Shopee and Lazada, while offline sales are handled manually by the business owner. The entire transaction recording and revenue management process is still carried out in a simple manner, without

the support of an automated and structured system. While transaction data is readily available, it has not been further processed to analyze monthly sales performance. This makes it difficult for business owners to make data-driven decisions, such as stock planning, promotional strategies, and revenue projections.

2. *Interview*

Interviews were conducted with the owner of Fa_al.store to identify challenges faced in business operations and the needs of the system to be developed. The interviews focused on stock management, transaction recording, and revenue reporting, which are currently handled manually. The information gathered from the interviews provided the basis for understanding key issues and helped formulate appropriate system features to support business operations.

3. *Observations*

Observations were conducted by observing Fa_al.store's operational processes, particularly stock management and transaction recording. These observations aimed to understand the workflow and challenges encountered in daily activities. The observations supplemented interview data to design a system that meets business needs.

3.2. *Planning*

1. *Functional Requirements*

Functional requirements are descriptions of the services or functions that a system must provide to function as intended.

a. *Inventory management*

The inventory management feature allows you to record and monitor product availability. The system records any additions or reductions to stock based on sales or procurement activity. Furthermore, the system allows users to monitor the number of items available in real time, set minimum stock levels, and provide alerts when stocks are running low. This feature is expected to aid inventory management and minimize the risk of stockouts.

b. *Sales Transaction Recording*

The system provides a sales transaction recording feature that records every purchase made by customers, both online and offline. Each recorded transaction stores important information such as the transaction date, number of products purchased, total payment, and payment method used. Furthermore, the system can generate transaction receipts, such as invoices or receipts, and store sales history for reporting and evaluating daily and monthly sales performance.

c. *Sales Transaction Recording*

The system is integrated with the Midtrans payment gateway to facilitate cashless payments. This feature allows customers to make payments using various Midtrans-provided methods, such as QRIS, e-wallets, and bank transfers. Payment statuses are automatically validated in the system, simplifying the confirmation process and expediting customer service.

2. *Non-functional requirements*

Non-functional requirements include technical elements that are not directly related to the main features, but have a significant impact on the overall quality of the system to be created.

1. *Authentication (login/logout)*

The system provides an authentication feature to ensure that only registered users can access certain features. Users are required to log in by entering a valid email address and password. The system also provides a logout feature to securely end a user session.

2. *Authorization*

Once a user successfully logs into the system, authorization is used to restrict access based on role. For example, only users with admin privileges can access the stock management and financial reporting pages. This is crucial to maintain data integrity and security.

3. *Security*

Once a user successfully logs into the system, authorization is used to restrict access based on role. For example, only users with admin privileges can access the stock management and financial reporting pages. This is crucial to maintain data integrity and security.

4. Input Validation

Input validation in the system is performed directly within the controller using basic programming logic. All data received from the user is manually checked, ensuring that the value is not empty, that the numeric format is correct, or that the submitted data type meets requirements. This approach is sufficient to prevent simple data errors and ensure that the system continues to accept input that conforms to the expected structure.

3.3. Modelling

1. Use Case Diagram

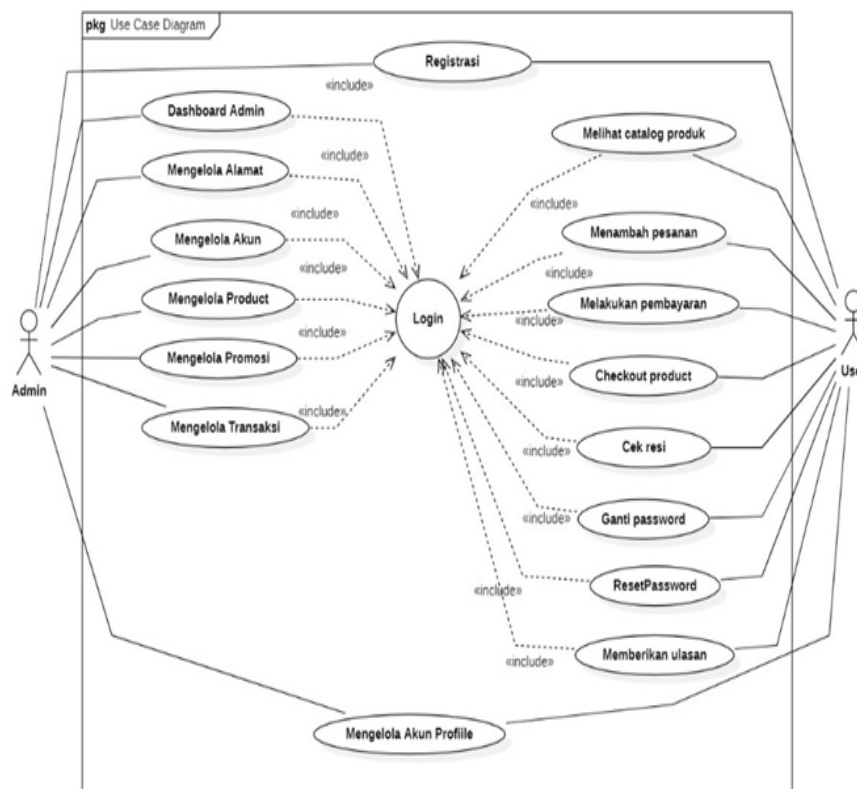


Figure 2. Use Case System

Use case diagrams serve as an essential tool for illustrating the functional requirements of a system from the user's perspective. These diagrams provide a clear visualization of how each actor, or system user, interacts with the core functions offered within the application. Through this representation, developers can better understand the roles, responsibilities, and access rights assigned to every type of user. Each interaction displayed in the diagram helps identify what users are allowed to do, which features they can access, and how they are expected to communicate with the system throughout various processes.

In addition, use case diagrams play a vital role in identifying system boundaries and clarifying the scope of development. They allow stakeholders to gain a shared understanding of the system's capabilities and limitations before moving forward to more detailed design stages. By mapping all user interactions, the diagrams help ensure that no essential functionality is overlooked during development.

In the context of this sales information and revenue forecasting system, the use case diagram illustrates the complete flow of activities performed by different actors, including administrators, staff, and other relevant users. This diagram provides a structured overview of the main features such as data management, reporting, forecasting, and system monitoring. The complete use case diagram used in the development process is presented in Figure 2.

2. Activity Diagram

a. Activity Diagram Login

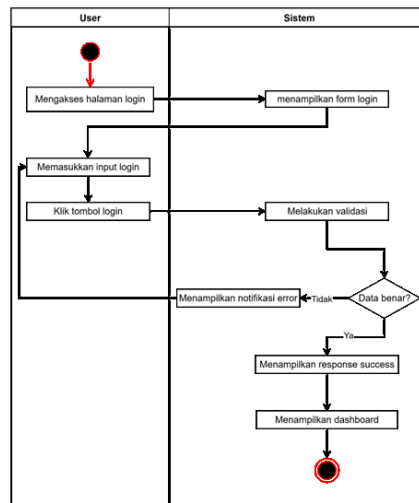


Figure 3. Activity Diagram Login

Figure 3 illustrates the process flow when a user logs into the system. The process begins when the user accesses the Login page and then enters authentication data such as email and password. After pressing the Login button, the system validates the input. If the entered data is incorrect, the system displays an error notification to the user. However, if the data is valid, the system displays a success response and redirects the user to the Dashboard page, indicating a successful login.

b. Activity Diagram Checkout

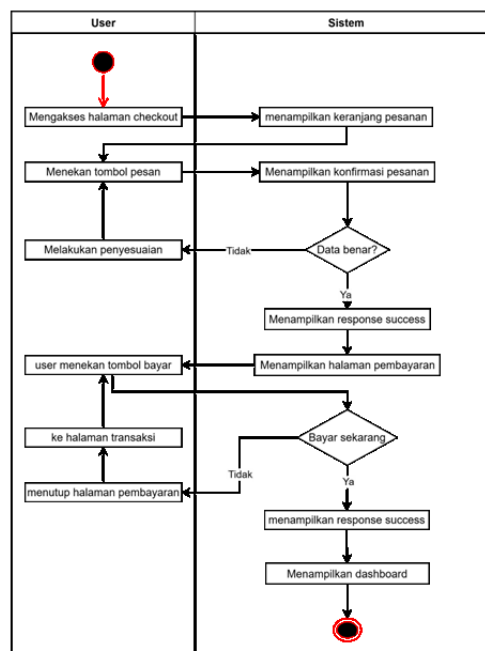


Figure 4. Activity Diagram Checkout

Figure 4 illustrates the user process for checking out an order on the system. The process begins when the user accesses the checkout page. The system displays a cart containing a list of items to be purchased. Next, the user presses the "order" button, which then triggers the system to display the order confirmation page. The user can adjust the data if necessary. The system validates the entered data. If the data is incorrect, the system displays an error notification. However, if the data is correct, the system displays a success notification. After that, the user can press the "pay" button to proceed to the payment page. The

system displays the payment interface, and the user is given the option to complete the payment. If the payment is successful, the system displays a success notification and redirects the user to the Dashboard page. Conversely, if the user does not make the payment and closes the page, the process ends without a successful transaction.

c. *Product Management Activity Diagram*

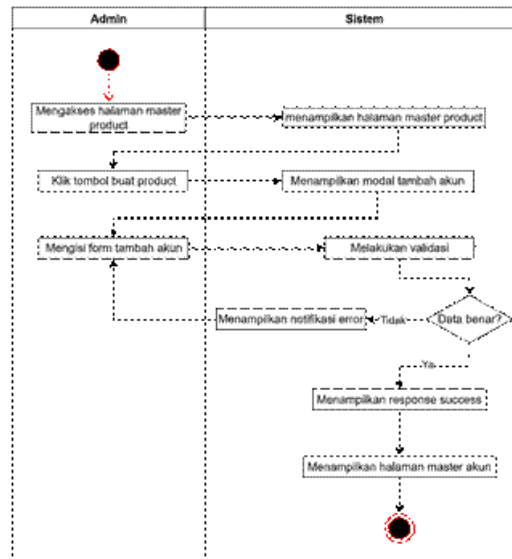


Figure 5. Product management activity diagram

Figure 5 illustrates the process when an admin adds a new product to the system via the product management page. The process begins when the admin accesses the product master page, where the system displays a list of existing products. The admin then clicks the add product button, which displays a product entry form in a modal or separate page. The admin then fills out the product form, which typically includes data such as product name, category, price, description, stock, image, and other attributes. Once the data is filled in, the admin clicks the save button, and the system validates all input. If errors are found, such as empty input, incorrect formatting, or duplication, the system displays an error notification to inform the admin that the data is invalid. If all data is valid, the system saves the new product to the database and displays a success response. The system then returns to the product master page with the newly added product data.

3. *Sequence Diagram*

a. *Sequence Diagram Login*

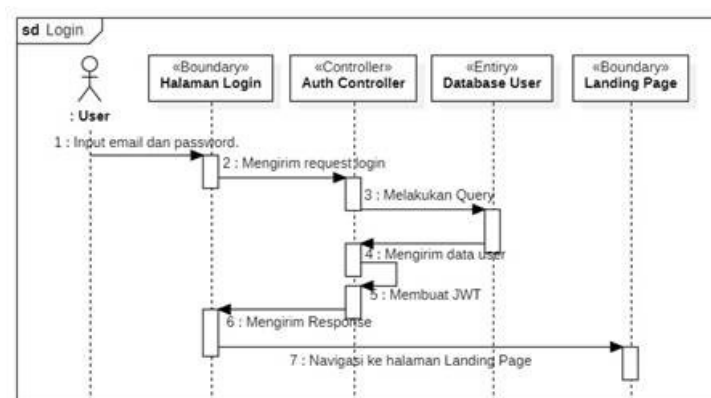


Figure 6. Sequence diagram login

Figure 6. illustrates the user authentication process flow during login. The process begins when the user enters their email and password on the login page (boundary) and then sends a request to the Auth Controller (controller). The Auth Controller queries the User Database (entity) to validate the entered credentials. If the data matches, the database returns the user data to the controller, which then creates a JWT as an authentication token. After that, the controller sends a response containing the token to the login page. If the process is successful, the system directs the user to the landing page (boundary) so they can access the system according to their access rights.

b. *Sequence Diagram Checkout*

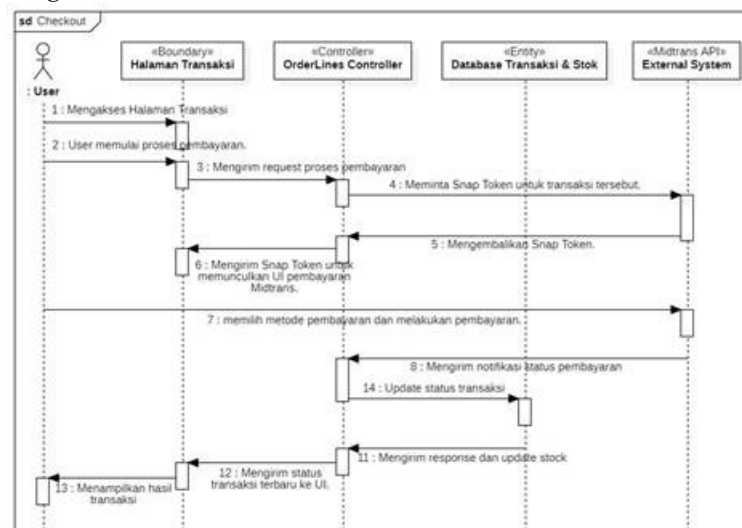


Figure 7. Sequence diagram checkout

Figure 7. illustrates the checkout process until payment is complete. The process begins when the user presses the checkout button on the Transaction Page (boundary), then the system sends a request to the OrderLines Controller (controller). The controller saves the initial transaction data to the Transaction & Stock Database (entity) and requests a Snap Token from the Midtrans API (external system). Once the token is received, the system displays the Midtrans payment UI so the user can select a payment method and complete the transaction. Next, Midtrans sends a payment status notification to the system via a webhook. The controller then updates the transaction and stock status in the database, and sends a response to the Transaction Page to display the latest payment status to the user.

c. *Sequence Diagram Manage Products*

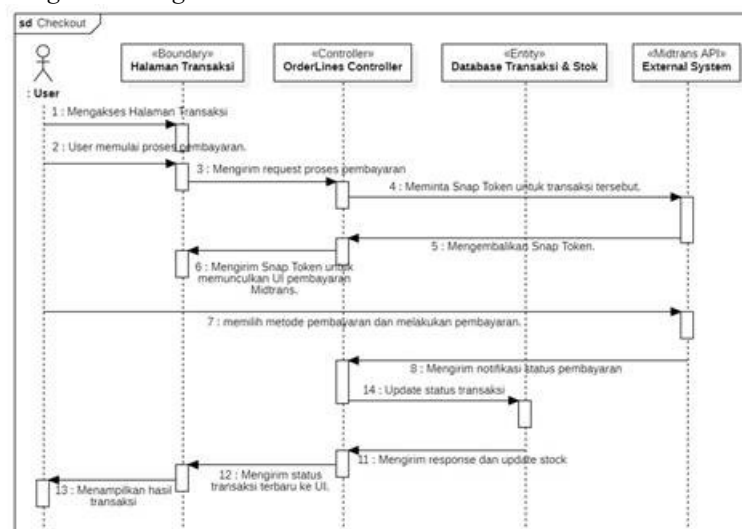


Figure 8. Sequence Diagram Manage Products

Figure 8. illustrates the checkout process until payment is complete. The process begins when the user presses the checkout button on the Transaction Page (boundary), then the system sends a request to the OrderLines Controller (controller). The controller saves the initial transaction data to the Transaction & Stock Database (entity) and requests a Snap Token from the Midtrans API (external system). Once the token is received, the system displays the Midtrans payment UI so the user can select a payment method and complete the transaction. Next, Midtrans sends a payment status notification to the system via a webhook. The controller then updates the transaction and stock status in the database, and sends a response to the Transaction Page to display the latest payment status to the user.

4. Class Diagram

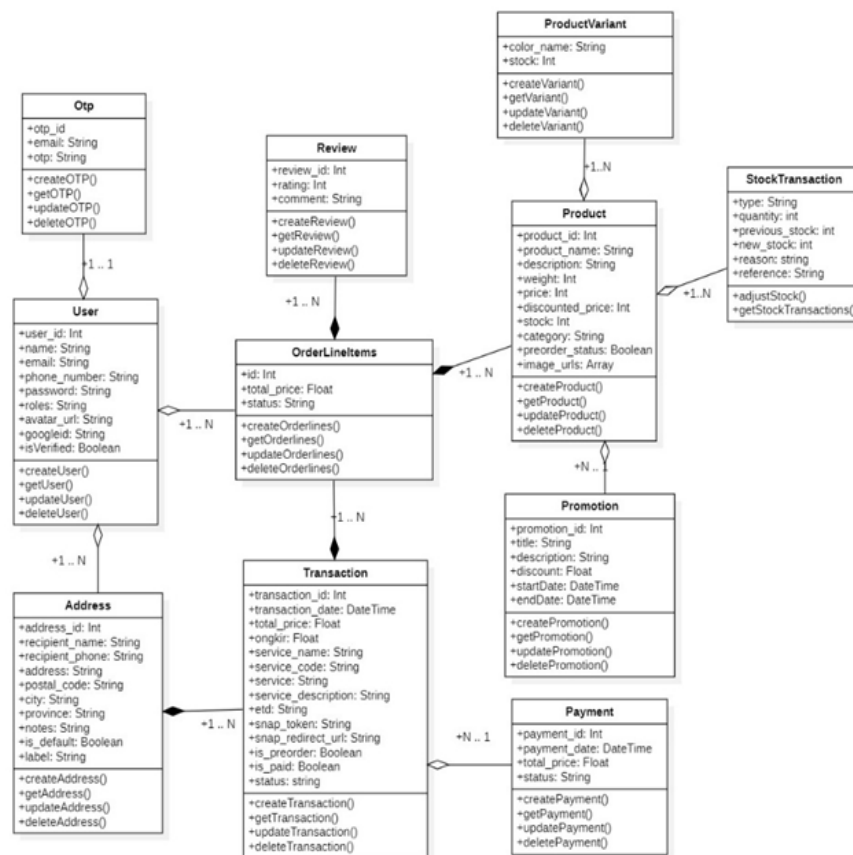


Figure 9. Class diagram

Class diagrams, as shown in Figure 9, represent a static structure diagram in UML that plays a crucial role in illustrating the architectural design of an object-oriented system. This diagram presents a detailed visualization of the classes that form the system, complete with their attributes, methods or functions, and the various types of relationships that connect them. These relationships may include associations that describe links between classes, generalizations that represent inheritance structures, and dependencies that indicate how one class relies on another to perform specific operations. Through this visual representation, developers can gain a clearer understanding of how data flows within the system and how individual components interact to support overall functionality.

Furthermore, class diagrams act as an essential blueprint during the development phase, as they ensure that both data structure and business logic are consistently aligned with system requirements. By mapping out the internal compositions and interactions at an early stage, potential design issues can be identified and addressed before implementation. This contributes to a more effective and maintainable system architecture. In addition, class diagrams facilitate communication among team members, allowing designers, developers, and stakeholders to share a unified perception of how the system is organized. Ultimately, this diagram becomes a foundational reference that supports the entire lifecycle of system development, from analysis and design to implementation and future enhancements.

5. Entity relational database

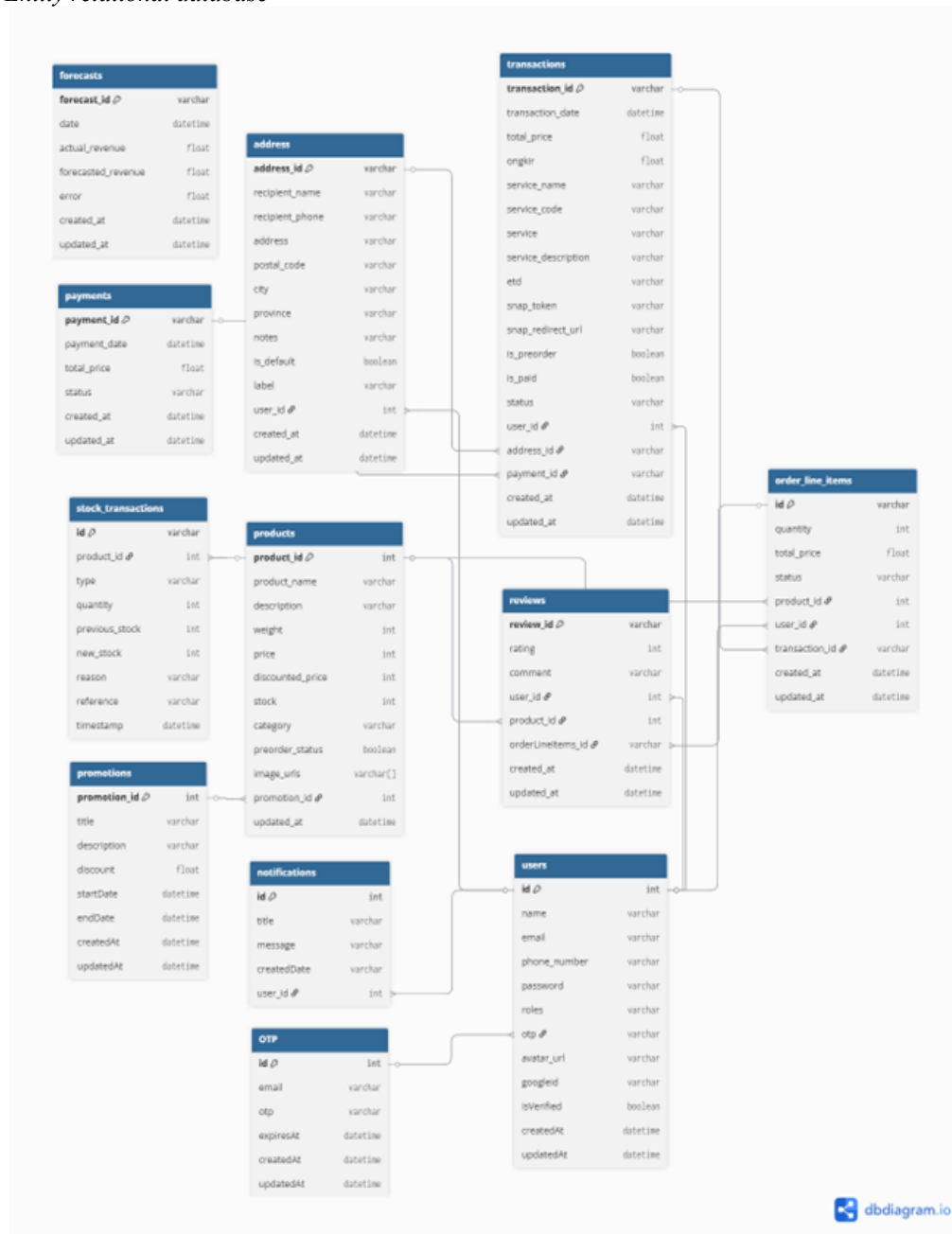


Figure 10. Entity relational database

Figure 10 presents the database design in the form of an Entity Relationship Diagram (ERD) that illustrates how various entities within the fa_al.store information system are interconnected to support end-to-end business processes. The diagram visualizes core entities such as users, products, transactions, payments, order_line_items, addresses, and reviews, each equipped with attributes that store essential operational data. In addition, supporting entities such as notifications, OTPs, promotions, and shipping details are also included to ensure the system can accommodate more complex service flows.

The relationships between these entities are defined through primary key and foreign key constraints, enabling structured data integration and ensuring data consistency across modules. This ERD serves not only as a reference for developers in understanding the logical structure of the system but also as a foundation for optimizing data processing, including product management, ordering, payment handling, delivery coordination, and post-transaction service evaluation. With this design, the overall system architecture becomes more systematic, scalable, and aligned with the needs of the business..

6. Wireframe

1. Wireframe Login

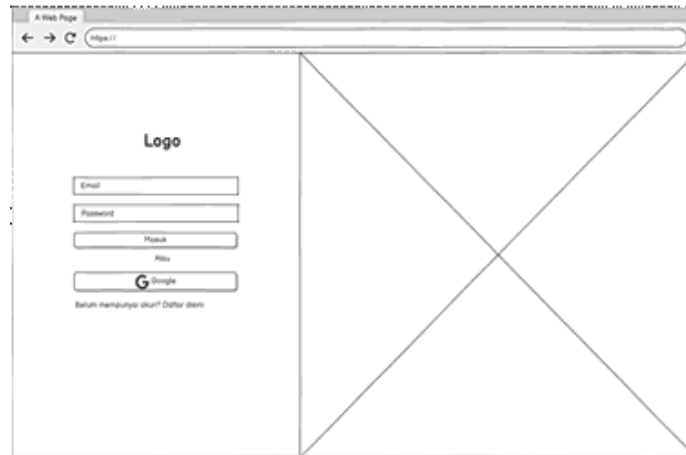


Figure 11. Wireframe login

Figure 11 menampilkan halaman login yang dirancang untuk memudahkan pengguna mengakses sistem secara cepat dan aman. Halaman ini menyediakan formulir berisi email dan kata sandi, tombol login utama, serta opsi masuk menggunakan akun Google untuk mempercepat proses autentikasi. Bagi pengguna baru yang belum memiliki akun, disediakan tautan menuju halaman pendaftaran agar proses registrasi dapat dilakukan secara praktis. Desain antarmuka dibuat sederhana, jelas, dan responsif sehingga dapat digunakan dengan nyaman melalui berbagai perangkat, baik desktop maupun mobile, sekaligus memastikan pengalaman login yang efisien dan user-friendly.

2. Wireframe Checkout

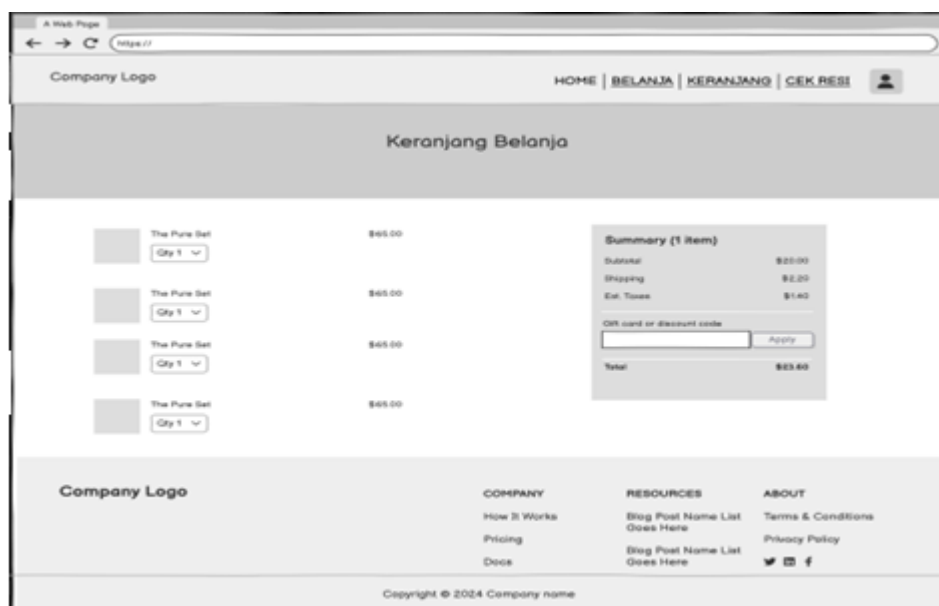
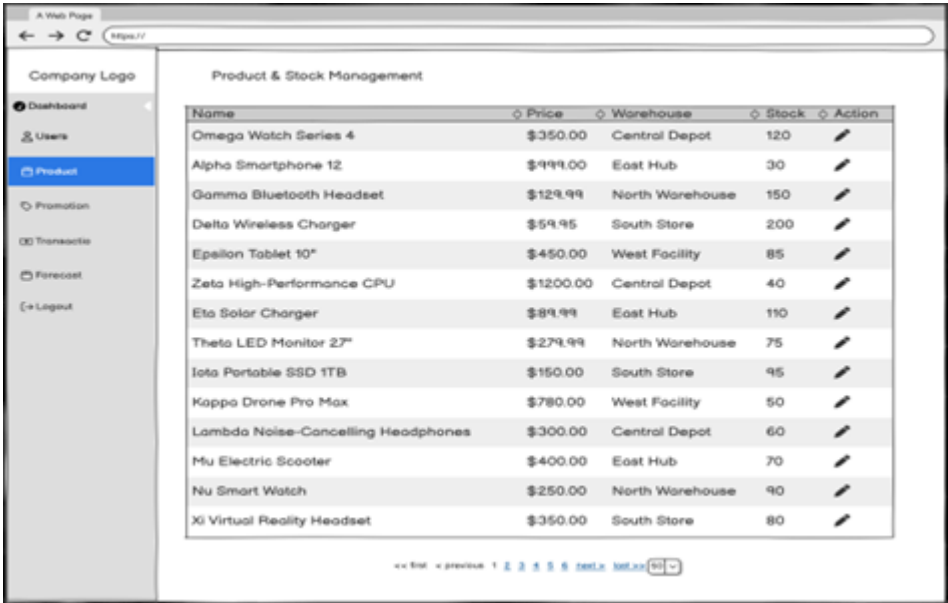


Figure 12. Wireframe checkout

Figure 12 shows the cart page that presents a complete overview of the products selected by the user before finalizing a purchase. On this page, users can review detailed information for each item, including product name, price, variation, and subtotal calculations. The interface also provides intuitive controls that allow users to increase or decrease product quantities, remove items if no longer needed, or continue browsing for additional products. This page is designed with a clear layout to support user decision-making

and minimize errors before checkout. Through this structured presentation, users can efficiently manage their orders and proceed smoothly to the payment process.

3. Wireframe Manage Product



Name	Price	Warehouse	Stock	Action
Omega Watch Series 4	\$350.00	Central Depot	120	
Alpha Smartphone 12	\$999.00	East Hub	30	
Gamma Bluetooth Headset	\$129.99	North Warehouse	150	
Delta Wireless Charger	\$59.95	South Store	200	
Epsilon Tablet 10"	\$450.00	West Facility	85	
Zeta High-Performance CPU	\$1200.00	Central Depot	40	
Eta Solar Charger	\$89.99	East Hub	110	
Theta LED Monitor 27"	\$279.99	North Warehouse	75	
Iota Portable SSD 1TB	\$150.00	South Store	95	
Kappa Drone Pro Max	\$780.00	West Facility	50	
Lambda Noise-Cancelling Headphones	\$300.00	Central Depot	60	
Mu Electric Scooter	\$400.00	East Hub	70	
Nu Smart Watch	\$250.00	North Warehouse	90	
Xi Virtual Reality Headset	\$350.00	South Store	80	

Figure 13. Kelola product

Figure 13. shows the admin page for managing all products available in the system. Product data is displayed in a table containing information such as product name, category, price, stock, and status. Admins can add new products, edit existing data, or delete products that are no longer relevant. The product management form includes input such as name, description, image, price, and stock. The interface is designed to be intuitive to facilitate the product management process.

3.4. Construction

1. Login Page Implementation

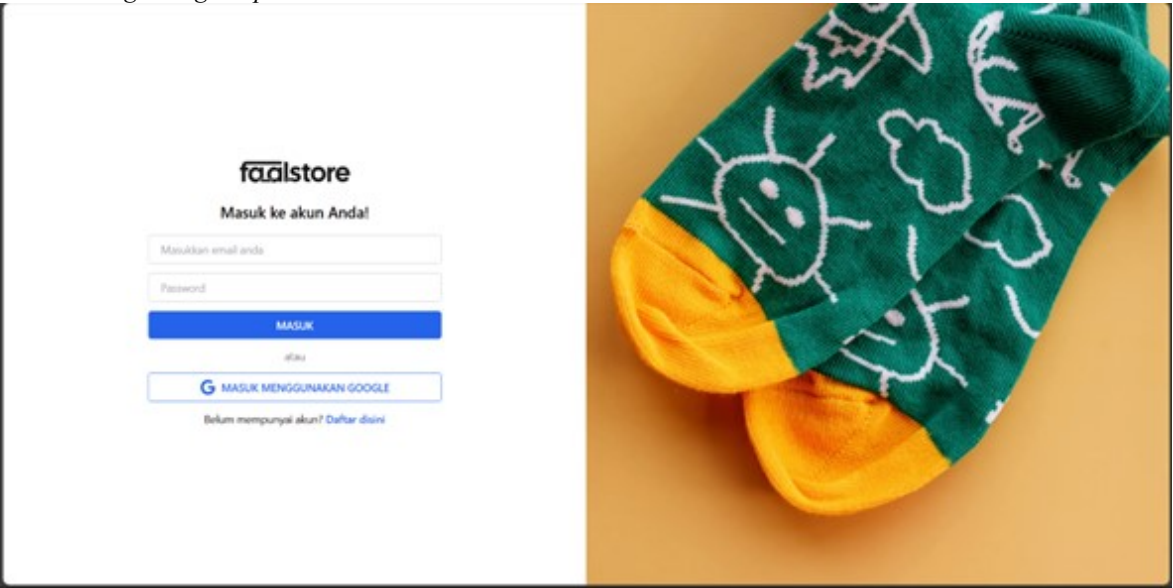


Figure 14. Login Page Implementation

Figure 14 displays the login page, which serves as the entry point for users who already have an account to access the system. On this page, users are asked to enter their previously registered email address and password. An alternative login button using a Google account is also provided to simplify the authentication process. The page design is simple and responsive for ease of use on various devices.

2. checkout page implementation

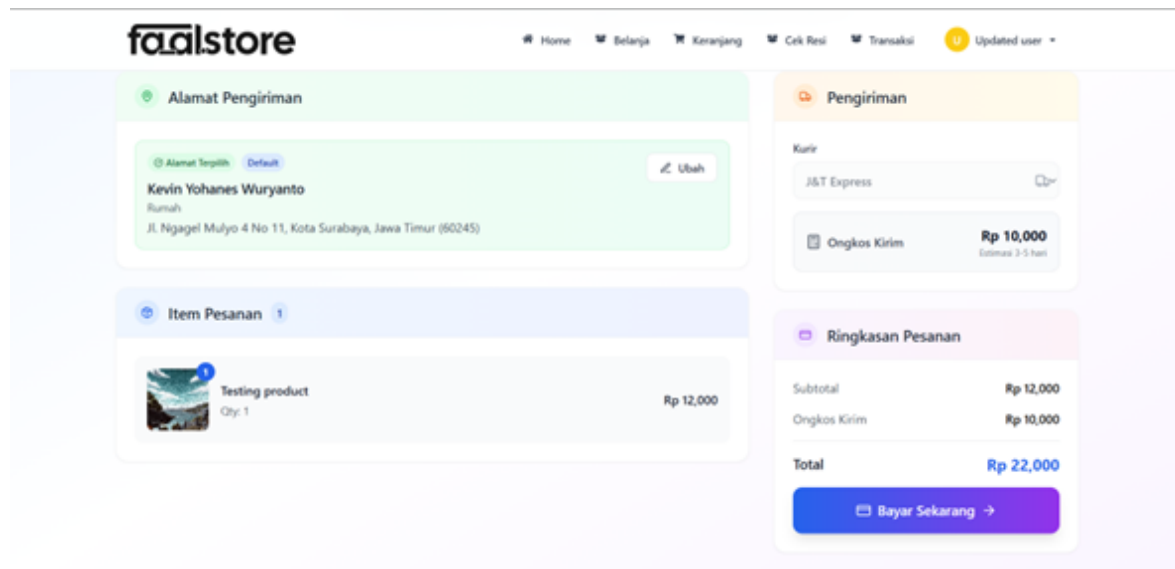


Figure 15. checkout page implementation

Figure 15 displays the transaction page interface on the website. This page is designed to facilitate the user's checkout process after selecting products and adding them to their cart. Users can create a shipping address, automatically calculate shipping costs based on the address and courier, and proceed to payment using a third-party service such as Midtrans. This interface also displays information about the payment process step by step, providing users with clear feedback as the transaction progresses. The design emphasizes ease, speed, and clarity in completing the purchase process.

3. Implementation of manage product page

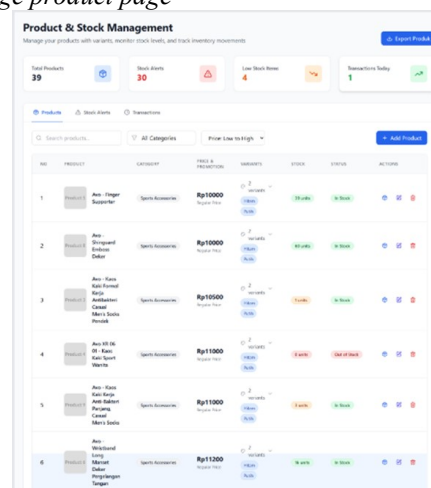


Figure 16. Implementation of manage product page

Figure 16 shows the Master Product page used by admins to comprehensively manage product data. This page provides various features such as search, filtering by category and price order, adding new products, and managing stock and sales. In addition, admins can monitor stock alerts in real time and view

stock transaction history, which is displayed periodically. This view is designed to give admins full control over the product data available in the system.

3.5. User acceptance testing

User Acceptance Testing (UAT) is a final testing method carried out by system users to ensure that the developed system meets functional requirements and is ready for use in a real operational environment. This testing is crucial for identifying whether the features in the system are functioning properly from the end-user perspective [14]. The success of an information system depends heavily on user acceptance of functional and interface aspects, which are assessed through a quantitative approach with a Likert scale in UAT [15][16]. Therefore, UAT is an essential stage in the development of an information system that is oriented towards satisfaction and successful use by end users. Table 1. is the user acceptance testing research question.

Table 1. User acceptance testing question

No.	Question
1	The system's features and functions meet the business needs of users.
2	The system's interface and navigation are easy to understand and use.
3	The system performs tasks effectively by displaying the expected data.
4	The system responds quickly to every user action.
5	The system complies with applicable regulations and standards, such as security and privacy.
6	The business processes automated by the system run smoothly and efficiently.
7	The system can be used across various devices and browsers.
8	The system's reports and output are accurate and meet user needs.
9	System error messages are clear and helpful in resolving issues.
10	The system is capable of handling complex processes without errors or disruptions.

Table 2 is a recapitulation of the answers from the Research User Acceptance Testing form.

Table 2. User acceptance testing form recapitulation

Respondent	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
1	4	5	4	5	5	4	4	5	5	5	47
2	5	4	5	4	5	5	4	5	5	5	48
3	5	5	5	4	5	5	4	5	5	5	47
4	5	5	5	4	5	5	4	5	5	4	43
5	5	5	5	5	5	5	5	5	5	5	48
6	5	4	5	4	5	5	5	4	5	5	49
7	4	5	5	4	4	5	5	5	5	5	43
8	5	5	5	4	5	5	3	5	5	5	48
9	4	5	4	4	5	5	4	5	5	5	50
10	5	5	4	5	4	5	5	4	5	4	48
Average	4,8	4,6	4,9	4,8	4,8	4,8	4,6	5,0	4,7	4,7	
Total Results											471

$$UAT \text{ Percentage} = ((\text{Total UAT score})/(\text{Maximum Score})) \times 100$$

$$UAT \text{ Percentage} = (471/500) \times 100$$

$$UAT \text{ percentage} = 94.2\%$$

User Acceptance Testing (UAT) was conducted to evaluate the extent to which the system met user needs and expectations. This testing involved asking ten respondents ten questions, each of whom provided a rating on a scale of 1 to 5.

Based on the results, the total score for all respondents was 471 out of a maximum score of 500. This result indicates that the UAT percentage was 94.2%. This score indicates that the system is in the "very good" category, meaning that it has been very well received by users in various aspects, both in terms of functionality and usability. Thus, the system is deemed suitable for use and has successfully met its intended objectives.

4. Conclusion

This research has successfully designed and built a website-based e-commerce information system using the ReactJS framework on the frontend and ExpressJS on the backend. This system was developed to provide convenience for business actors, especially online store owners, in managing products, transactions, and reports. The implemented features include inventory management, sales transaction recording, customer data management, promotion management, and the provision of sales reports. The integration of the Midtrans payment gateway makes the online payment process easier, safer, and supports various available payment methods. The results of system testing using User Acceptance Testing (UAT) indicate that the system has been able to meet the functional needs of users. Based on respondent feedback, the system's main features are considered to be running well and according to expectations. The user acceptance rate of 94.2% indicates that the developed system is feasible to use and can significantly support business processes.

References

- [1] C. Yolanda, "Peran Usaha Mikro, Kecil Dan Menengah (UMKM) Dalam Pengembangan Ekonomi Indonesia," J. Manaj. DAN BISNIS, vol. 2, no. 3, hlm. 170–186, Apr 2024, doi: 10.36490/jmdb.v2i3.1147.
- [2] I. Indriyani, I. P. B. Wiranata, dan S. Hiu, "Strategi Peningkatan Efisiensi Operasional UMKM di Era Digital: Pendekatan Kualitatif dengan Business Intelligence dalam Implementasi E-commerce," Inform. Educ. Prof. J. Inform., vol. 9, no. 1, hlm. 23, Jun 2024, doi: 10.51211/itbi.v9i1.2760.
- [3] A. R. Sijabat dan E. Elisa, "RANCANG BANGUN SISTEM INFORMASI PENJUALAN ALAT PANCING DI KOPERASI NELAYAN BATAM MADANI," vol. 10, no. 05, 2024.
- [4] N. I. Eltiana dan H. K. Saputra, "Elevating Efficiency: Designing an Integrated E-commerce Platform for Atlanta Sport Store Based on Yii Framework," J. Hypermedia Technol.-Enhanc. Learn., vol. 2, no. 1, hlm. 48–62, Feb 2024, doi: 10.58536/j-hytel.v2i1.111.
- [5] M. Tizar dan N. Azizah, "RANCANG BANGUN SISTEM INFORMASI PENJUALAN BERBASIS WEB (E-COMMERCE) PADA TOKO RUMAH POPOK KINAN," EDUSAINTEK J. Pendidik. Sains Dan Teknol., vol. 10, no. 1, hlm. 154–170, Des 2022, doi: 10.47668/edusaintek.v10i1.664.
- [6] J. H. P. Sitorus dan M. Sakban, "Perancangan Sistem Informasi Penjualan Berbasis Web Pada Toko Mandiri 88 Pematangsiantar".
- [7] M. I. Surya Pratama, "INTEGRASI PAYMENT GATEWAY PADA APLIKASI POINT OF SALES BERBASIS WEBSITE MENGGUNAKAN FRAMEWORK REACTJS (STUDI KASUS : TOKO ADIDA PRATAMA)," J. Inform. Dan Tek. Elektro Terap., vol. 13, no. 3, Jul 2025, doi: 10.23960/jitet.v13i3.7099.
- [8] M. Izzuddin, N. C. Wibowo, dan E. D. Wahyuni, "RANCANG BANGUN SISTEM INFORMASI MANAJEMEN RUANG RAPAT PADA PT XYZ MENGGUNAKAN JAVASCRIPT," J. Inform. Dan Tek. Elektro Terap., vol. 13, no. 2, Apr 2025, doi: 10.23960/jitet.v13i2.6305.
- [9] S. K. Murti dan A. Sujarwo, "Membangun Antarmuka Pengguna Menggunakan ReactJs untuk Modul Manajemen Pengguna".
- [10] "Rancang Bangun E-commerce Berbasis Single Page Application (SPA) Menggunakan ReactJS."
- [11] A. E. Pradina, N. Vendyansyah, dan R. P. Prasetya, "PENERAPAN METODE SINGLE MOVING AVERAGE DALAM SISTEM PERAMALAN PENJUALAN PADA TOKO SERAGAM SEKOLAH AYZAM," vol. 7, no. 5, 2023.
- [12] R. S. Pressman, Software engineering: a practitioner's approach, 5th ed. dalam McGraw-Hill series in computer science. Boston, Mass: McGraw Hill, 2000.
- [13] R. O. Obe dan L. S. Hsu, POSTgreSQL: up and running, Second edition. Beijing: O'Reilly, 2015.
- [14] S. R. Yulistina, T. Nurmala, R. M. A. T. Supriawan, S. H. I. Juni, dan A. Saifudin, "Penerapan Teknik Boundary Value Analysis untuk Pengujian Aplikasi Penjualan Menggunakan Metode Black Box Testing," J. Inform. Univ. Pamulang, vol. 5, no. 2, hlm. 129, Jun 2020, doi: 10.32493/informatika.v5i2.5366.
- [15] H. Thabibi, S. F. A. Wati, dan T. P. Rinjeni, "Implementasi User Acceptance Testing (UAT) Pada Website E-commerce UMKM BBhealthy," Adopsi Teknol. Dan Sist. Inf. ATASI, vol. 4, no. 1, hlm. 19–26, Jun 2025, doi: 10.30872/atasi.v4i1.2904.
- [16] Aliyah Aliyah, Nahrin Hartono, dan Asrul Azhari Muin, "Penggunaan User Acceptance Testing (UAT) Pada Pengujian Sistem Informasi Pengelolaan Keuangan Dan Inventaris Barang," Switch J. Sains Dan Teknol. Inf., vol. 3, no. 1, hlm. 84–100, Des 2024, doi: 10.62951/switch.v3i1.330.

- [17] JI. J. Habibah Nurfauziah, "PERBANDINGAN METODE TESTING ANTARA BLACKBOX DENGAN WHITEBOX PADA SEBUAH SISTEM INFORMASI," J. Vis., vol. 8, no. 2, pp. 105–114, 2022.
- [18] R. S. U. Fitria Nur Hasanah, Rekayasa Perangkat Lunak. Sidoarjo: UMSIDA PRESS, 2020.
- [19] L. Setiyani, Rekayasa Perangkat Lunak[Software Engineering], no. May. 2019.
- [20] A. N. Athaya and N. L. Marpaung, "Rancang Bangun Aplikasi Bon Permintaan Dan Pengeluaran Barang Menggunakan Metode Prototype Berbasis Website," J. Inform. J. Pengemb. IT, vol. 8, no. 2, pp. 134–141, 2023.
- [21] W. Herwansyah and N. L. Marpaung, "Rancang Bangun Sistem Informasi Toko Izdiihar Laptop Berbasis Website," pp. 484–497, 2023.
- [22] F. Z. Badeni, A. Guntara, and I. Fadil, "Implementasi Sistem Informasi Manajemen Aset Kelompok Ternak Ikan Hias Sumedang Menggunakan Metode Prototype," vol. 18, pp. 196–208, 2024.
- [23] D. Ardiyansah et al., "IMPLEMENTASI METODE PROTOTYPING PADA SISTEM INFORMASI," vol. 2, pp. 17–22, 2021.
- [24] M. T. I Putu Agus Eka Pratama, S.T., PROTOTYPING SEBAGAI MODEL PENGEMBANGAN SOFTWARE. CV.RUANG TENTOR, 2023.
- [25] N. Hartono and A. A. Muin, "Penggunaan User Acceptance Testing (UAT) Pada Pengujian Sistem Informasi Pengelolaan Keuangan Dan Inventaris Barang," 2025.