Contents lists available at JESA

Journal of Engineering and Science Application

journal homepage: https://jesa.aks.or.id/index.php/jesa/index

Web-Based Point-of-Sale System with Inventory and Membership Management for KODAGAMA

Ridwan Cahya Gumilar, Ananda Syafira, Aditia Putra, Wasis Haryono
Informatics Engineering Study Program, Faculty of Computer Science, Universitas Pamulang, South Tangerang, Indonesia

Article Info

Keywords:

Black-Box Testing
Cooperative Retail
Inventory Management
Membership
Point Of Sale
User Acceptance Testing

ABSTRACT

The KODAGAMA cooperative retail unit previously maintained inventory records separately from sales, requiring staff to reconcile products sold with stock balances. This study presents a web-based point-of-sale application integrating product management, sales transactions, inventory, membership benefits, and reporting. Development followed the Waterfall stages of requirements analysis, design, implementation, and verification, with requirements identified through observation, interviews, and review of existing records. Use case, activity, sequence, and class diagrams explain staff responsibilities and application behavior, while an entity relationship diagram and an architecture view describe data relationships and system components. A checkout wireframe and eight application screenshots document the interface and implemented features. Functional evaluation uses the black-box cases recorded in the work-placement report. Of 172 cases, 170 were recorded as matching the expected outputs and two did not, both involving options on an earlier sign-in interface. These records do not establish a verified pass rate for a later deployment. User Acceptance Testing is specified through representative staff tasks and acceptance criteria for product management, membership, checkout, transaction history, and reporting. Acceptance results are not yet available; no satisfaction score or acceptance decision is reported. The findings describe the documented functionality and provide an acceptance protocol for evaluating operational suitability. Further execution should reconcile invoice and stock records during settlement. Improvements in operational time, stock accuracy, and user satisfaction were not measured.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Ananda Syafira
Program Studi Teknik Informatika, Fakultas Ilmu Komputer
Universitas Pamulang
Tangerang Selatan, Indonesia
Email: anandasyafira878@gmail.com

1. INTRODUCTION

The retail unit of KODAGAMA, a consumer cooperative in Jakarta, maintained spreadsheet inventory records separately from sales. Staff reconciled quantities sold against stock balances, while payment records alone did not identify each product and quantity. The cooperative also needed to link purchases to member benefits without making membership compulsory at checkout. Inventory studies

establish the significance of record accuracy and visibility [1], [2], [3], [4], [5]. This study addresses the organization of related retail records; stock discrepancies at KODAGAMA were not measured.

Previous web applications integrate sales and inventory for stationery shops [3], [4], coffee businesses and general retailers [5], [6], pharmacies [7], perfume retailers [8], and stores with distinct staff roles [9].

The KODAGAMA case requires these functions to operate around the same sale. A product must be identifiable during selection, its quantity and price must be retained in the invoice, and payment status must determine the subsequent inventory effect. Membership adds a further relationship because an eligible purchase may earn points or use discounts. Treating each function in isolation would leave the relationship between the records unexplained. The design therefore uses the invoice as the link between checkout, stock, member benefits, history, and reporting.

The study asks how these records are connected, which observable behaviors are supported by the documented functional cases, and how staff can assess the application against their daily tasks. Its contribution is a complete design account covering user responsibilities, workflow, interactions, classes, data, architecture, and interface, together with implementation screenshots and functional test records. UAT is provided as a separate acceptance protocol because completed acceptance results are not yet available.

2. RESEARCH METHOD

2.1. Case setting and development process

The work-placement case concerns Koperasi Konsumen Juanda Tiga Lima (KODAGAMA). Requirements were identified through observation, staff interviews, and review of existing retail records. Development followed the Waterfall stages of requirements analysis, design, implementation, and verification [13]. The design account draws on the project source at commit 97843f3; the supplementary interface captures show the application on 20 November 2026. These sources establish the scope of the case rather than a performance comparison between deployments [10]. Review points between stages help relate the implementation to the earlier requirements [11].

2.2. Functional requirements and evaluation evidence

Table 1 organizes the requirements by operational responsibility and the records needed to carry out that responsibility. UML views describe the actor, activity, interaction, and object perspectives [16], [3], [13]. The ERD and architecture views complement them by describing storage relationships and component boundaries. Functional evaluation uses the report's 172 black-box cases. Screenshots illustrate implemented features, while planned UAT tasks address staff acceptance. These evidence types are reported separately.

Table 1. Functional requirements and the principal records involved

Operational area	Required behavior	Principal records
Product catalogue	Identify and maintain product classification, selling price, and stock.	Products, categories, store categories
Checkout	Record selected quantities, discounts, tender, change, and invoice details.	Invoices and sale lines
Membership	Register and retrieve members and associate eligible purchases with benefits.	Loyalty members and member-linked invoices
Payment settlement	Distinguish paid cash sales from pending noncash invoices and their confirmation.	Payment method, status, and inventory balance
History and reporting	Retrieve invoices and summarize paid sales for a selected period.	Transactions, sale lines, report filters
Staff access	Provide authenticated access for administrator and cashier work.	Staff accounts and roles

3. RESULTS AND DISCUSSION

3.1. Use case diagram

Figure 1 identifies administrator and cashier responsibilities. Shared work includes product lookup, sales, noncash confirmation, history, and reports. Administrative work includes master data, membership, user accounts, and the dashboard. The customer member is a record selected during a sale, not a staff login. The diagram defines operational responsibilities; authorization is implemented separately by the applicable routes and middleware.

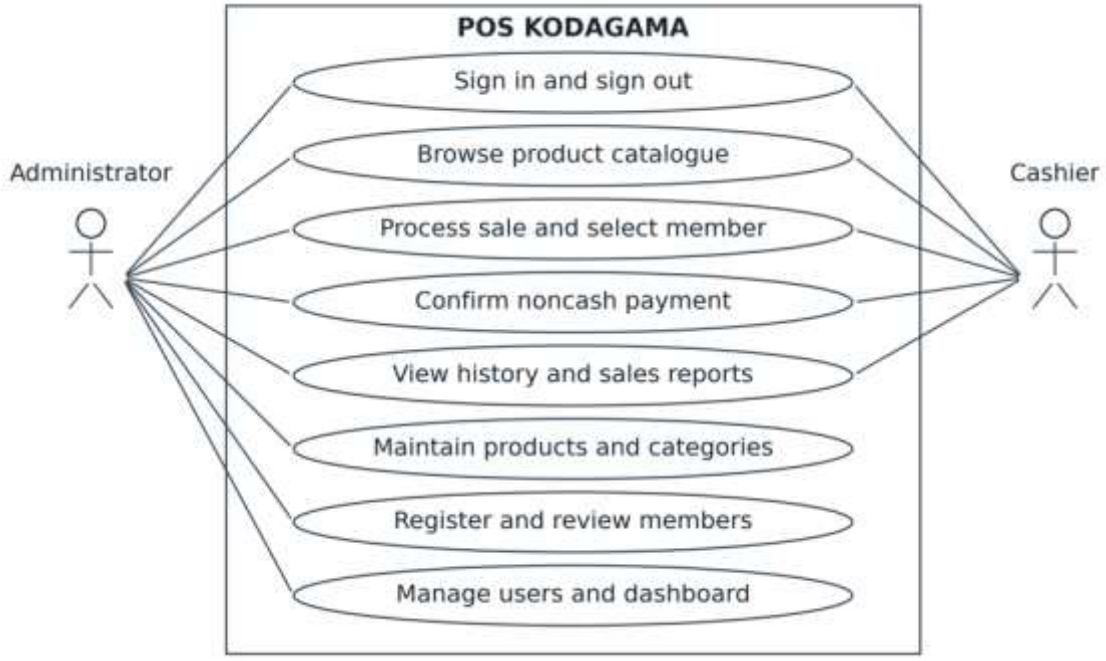

Figure 1. Use case diagram of staff responsibilities in POS KODAGAMA.

3.2. Activity diagram

Figure 2 follows checkout from product selection to settlement. Invalid input returns to correction. A cash sale requires sufficient tender and is recorded as paid. QRIS or transfer creates a pending invoice, followed by staff verification of receipt and confirmation [14]. The pending invoice does not reserve stock. The merge brings the settlement alternatives into the inventory and eligible-point update.

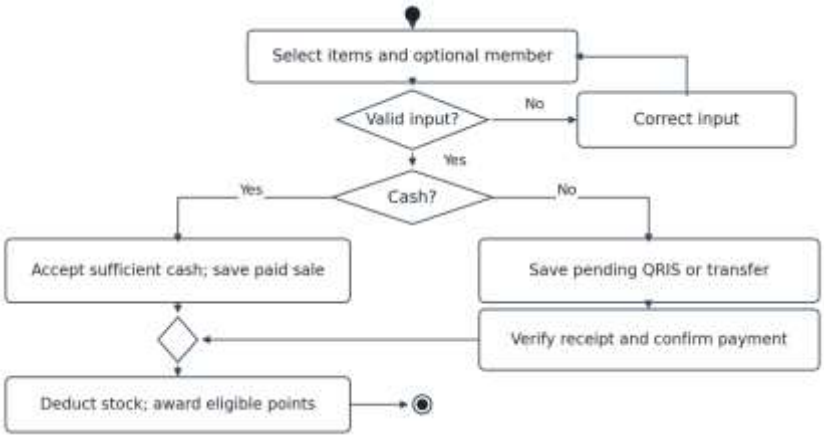

Figure 2. Activity diagram of checkout and payment settlement.

3.3. Sequence diagram

Figure 3 follows cash checkout. TransactionController reads products, validates stock, applies optional member benefits, and creates the invoice and lines. Related writes commit together; processing exceptions trigger rollback [13], [15], [16]. The interface receives the invoice or an error.

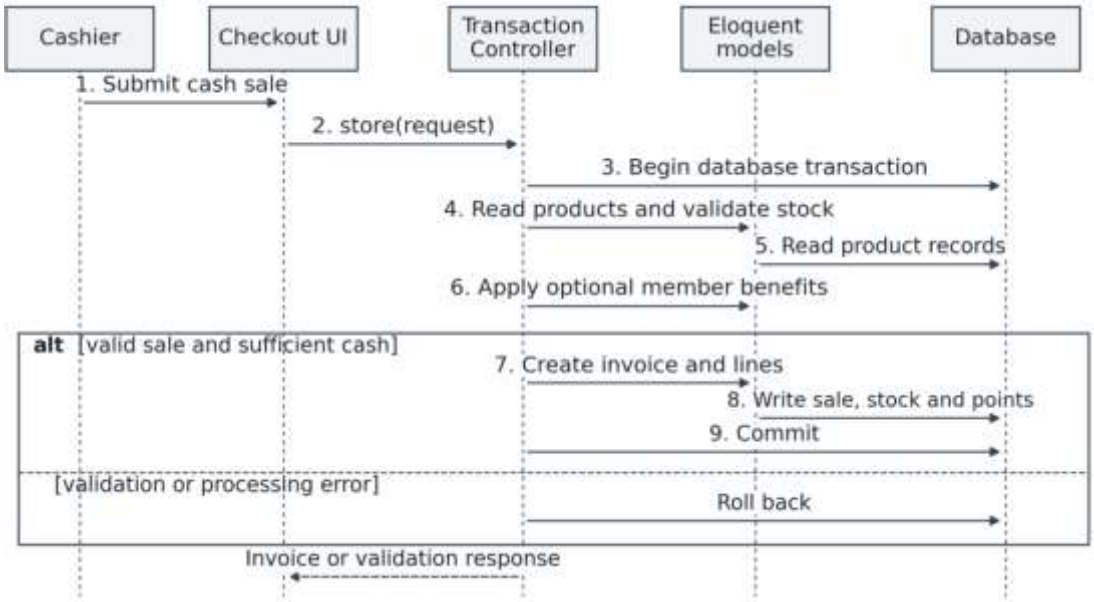

Figure 3. Sequence diagram of cash checkout and processing-error handling.

3.4. Class diagram

The application uses object-oriented PHP. Figure 4 shows the controller and four Eloquent models. Transaction groups TransactionDetail records, each referring to a Product. Optional Member associations connect sales with benefit methods. Dashed arrows represent controller dependencies; solid lines show model relationships and multiplicities.

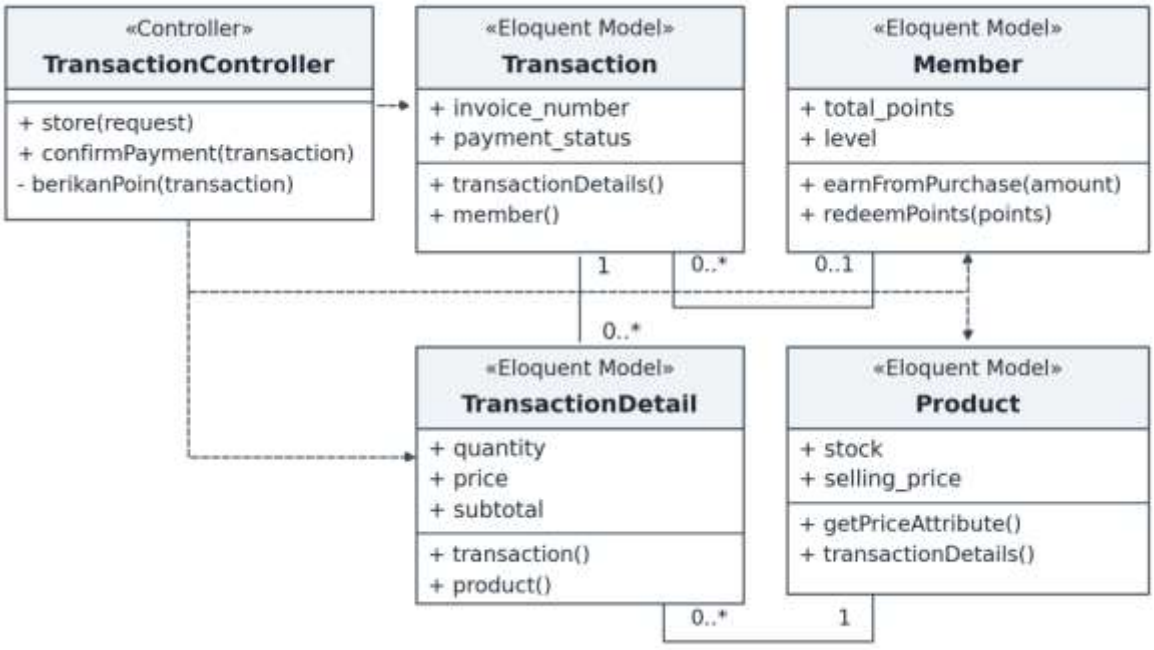

Figure 4. Selected classes, methods, controller dependencies, and model associations.

3.5. Entity relationship diagram

Figure 5 uses the model table names and logical keys. Product types and store categories classify products. Transactions hold invoice-level amounts and payment state, while transaction_details retain product identifiers, quantities, and sale prices. The separate line records permit one invoice to contain several products without repeating invoice information for every item. Membership is stored in loyalty_members and linked by nullable transactions. member_id. A sale may therefore exist without a member, while one member may have several sales. The users entity supports staff authentication; the reviewed transaction model has no creator-user foreign key. PK and FK identify primary and foreign keys. The diagram states the logical model relationships and does not certify a fresh database installation.

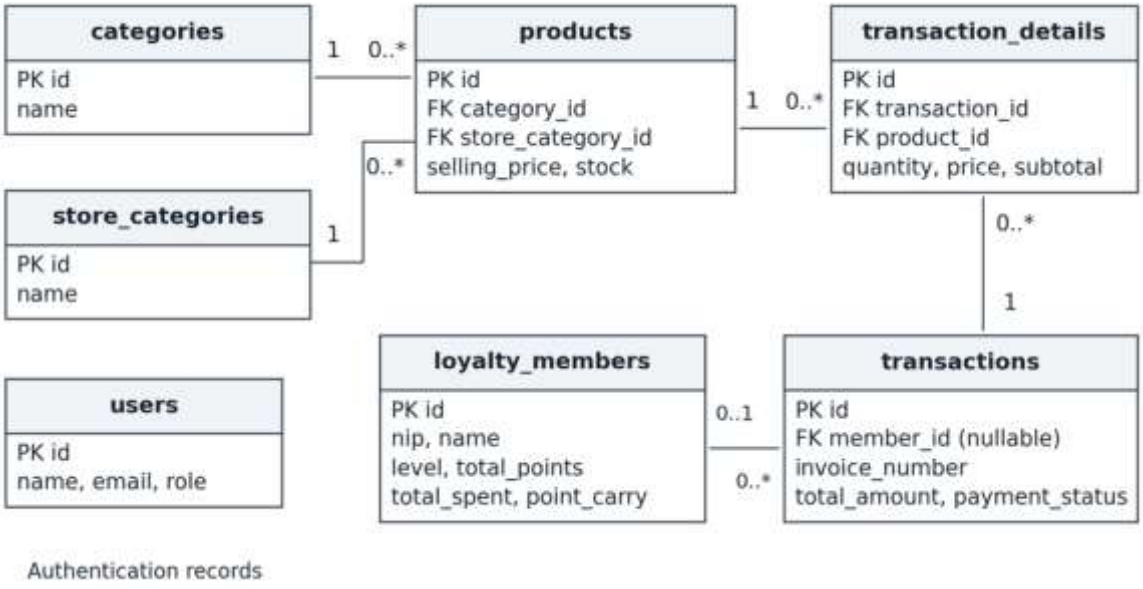

Figure 5. Logical ERD of product classification, sales, membership, and staff accounts.

3.6. System architecture

Figure 6 locates the desktop browser, Laravel routes and middleware, controllers, Eloquent models, database connection, and Blade views. The project specifies PHP 8.2 or later and Laravel 12. HTTP requests reach application logic through routes; models represent data and relationships; views render HTML for staff. The database engine follows the configured connection rather than being inferred from the available driver options [13].

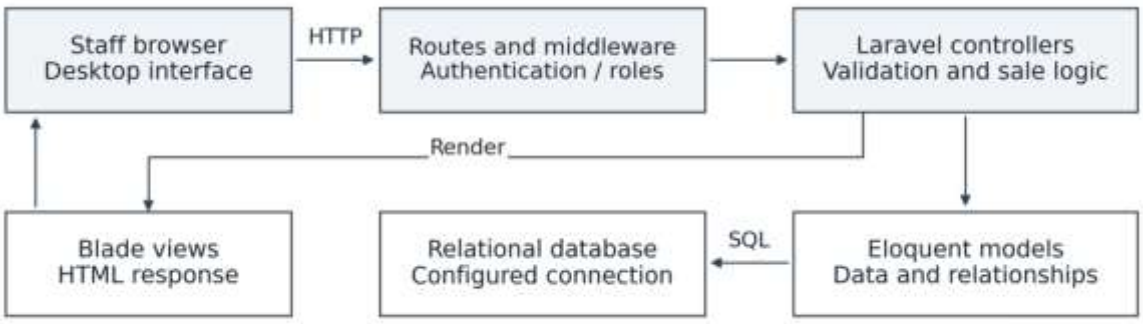

Figure 6. Principal components and request, rendering, and storage paths.

Checkout is the main integration point. Its controller coordinates price calculation, invoice creation, inventory, and member benefits. Request validation checks identifiers, item quantities, payment method, and nonnegative amounts. Middleware applies authentication and role checks to the routes on which it is configured. This arrangement separates interface presentation from stored records, although some business calculations remain in the controller.

3.7. User interface design

The interface design groups tasks around the staff workflow. Navigation provides access to the principal modules, product cards support recognition and selection, and the basket keeps the sale calculation visible. Figure 7 represents the desktop checkout layout before attention is directed to its implemented screens. It is a wireframe of information grouping rather than a usability score.

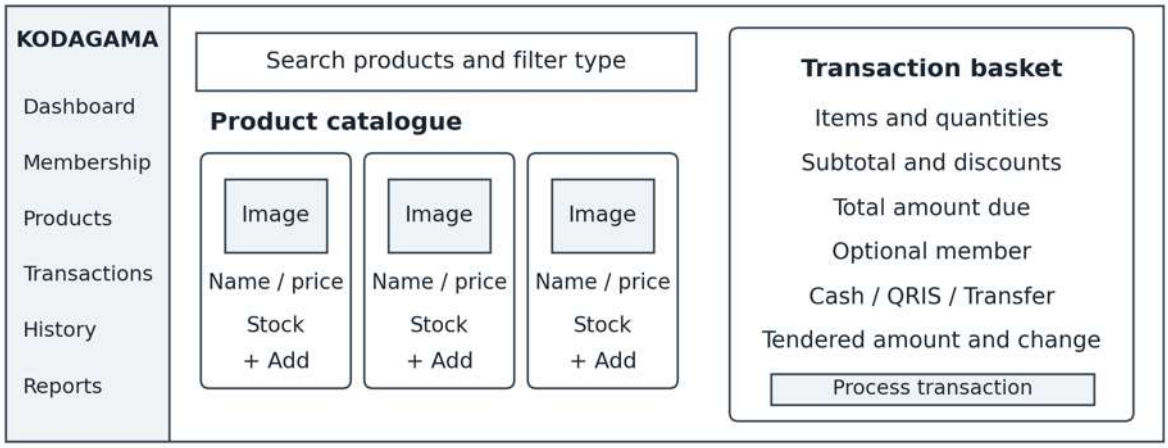

Figure 7. Desktop checkout wireframe showing catalogue, basket, and payment information.

Table 2 connects the visible regions to their intended interactions. Product identification is kept close to price and stock. Quantity controls and discount input belong to the basket, where their effect on the amount due can be inspected. Member selection is optional and should not prevent a nonmember sale. Payment choices are grouped with the information needed for the selected method.

Table 2. Interface regions and intended staff interactions

Interface region	Information presented	Intended interaction
Navigation	Module labels and the active menu.	Move between master data, checkout, history, and reports.
Product catalogue	Product image, name, selling price, available stock, and type filter.	Search, filter, and add the intended item.
Transaction basket	Selected items, quantities, subtotal, discounts, and amount due.	Review or change the sale before submission.
Membership selection	Member lookup associated with the current sale.	Retrieve the intended member or continue without one.
Payment and feedback	Cash, QRIS, or transfer; tender and change where applicable; transaction status.	Submit the sale and distinguish completed payment from a pending invoice.

For cash, the tendered amount and change support completion at the counter. For noncash, the interface needs a visible pending state and a separate confirmation action after receipt is verified. A successful-save message, a validation message, and a pending-payment message describe different outcomes. Their wording should allow staff to understand which action is required next.

Transaction history remains separate from the active basket so that retrieving an earlier invoice does not replace the current checkout context. Report filters identify the period being summarized. The following screenshots show the implemented pages in full so that the design regions can be related to the application used during the work placement.

4. IMPLEMENTATION SCREENSHOTS

4.1. Login interface

Figure 8 shows the login page with staff ID and password fields, the “Remember me” choice, and the sign-in action. The submitted image masks the password. Authentication establishes the staff session

before the operational pages are used. The presence of a control documents its implementation in the interface; the earlier functional verdict for that control is discussed in Section 5.

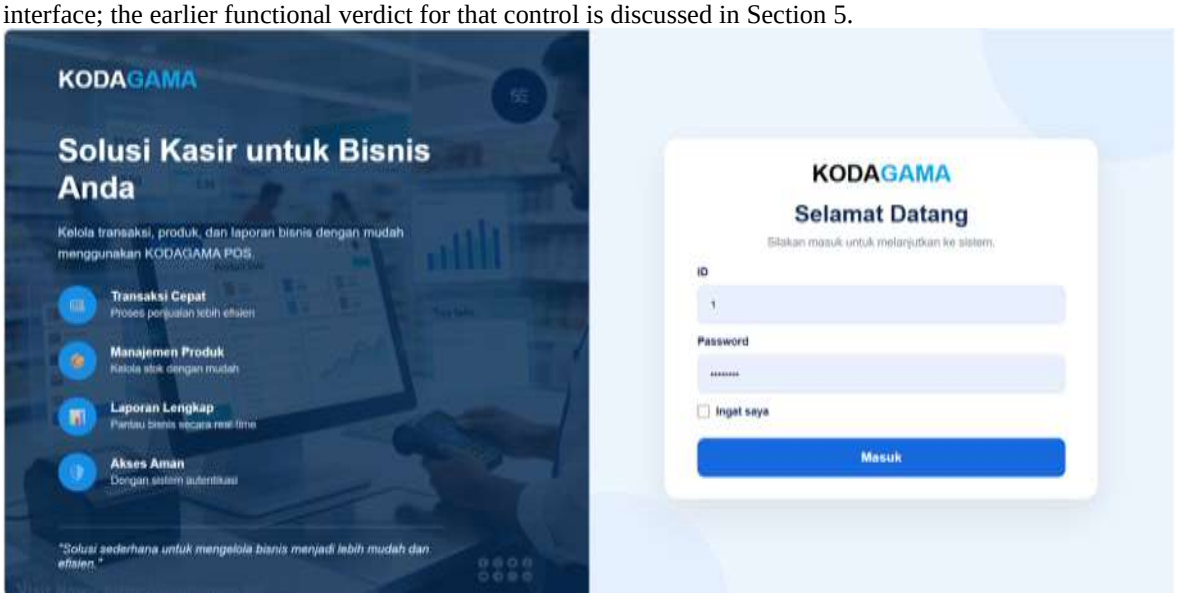

Figure 8. Implemented login page with staff ID and password fields.

4.2. Administrator dashboard

Figure 9 presents the dashboard summary, sales graph, and low-stock region. At capture, the cards show seven products, three categories, 66 transactions, and an IDR 747,800 sales total. The graph displays a selected 30-day view with IDR 183,000. These different summaries must be interpreted with their displayed period and scope before being compared.

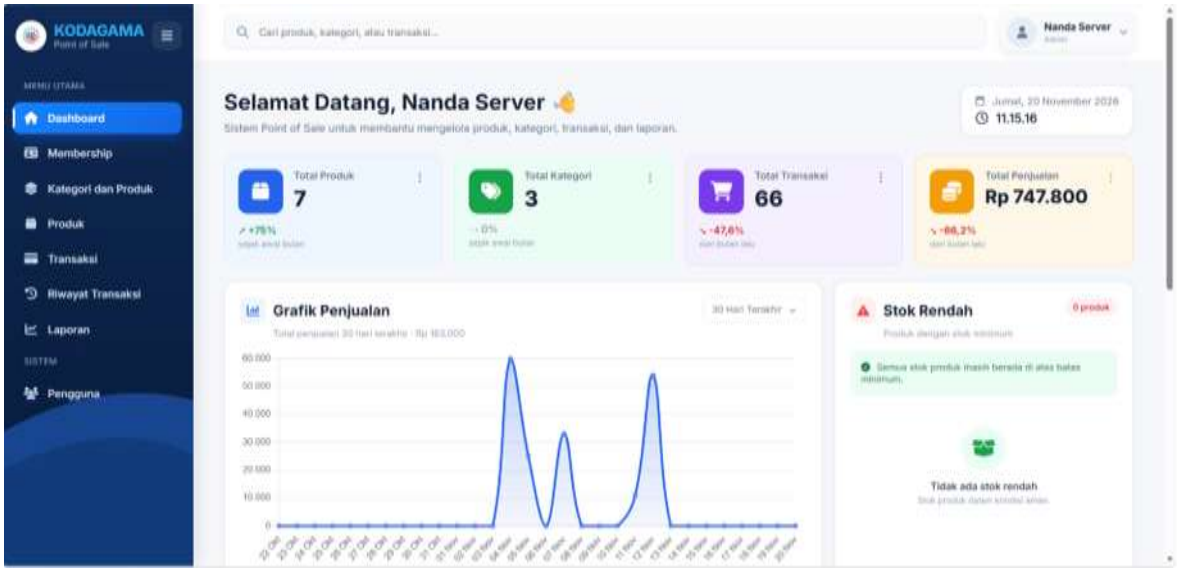

Figure 9. Administrator dashboard with summary cards, sales graph, and low-stock information.

4.3. Membership interface

Figure 10 shows membership totals, active-member totals, accumulated points, the registration action, and the beginning of the member-list region. At capture, the summary displays two members, two active members, and two total points. The banner describes the IDR 50,000 point-earning threshold. Member selection later links an eligible purchase to the relevant record instead of requiring every customer to register.

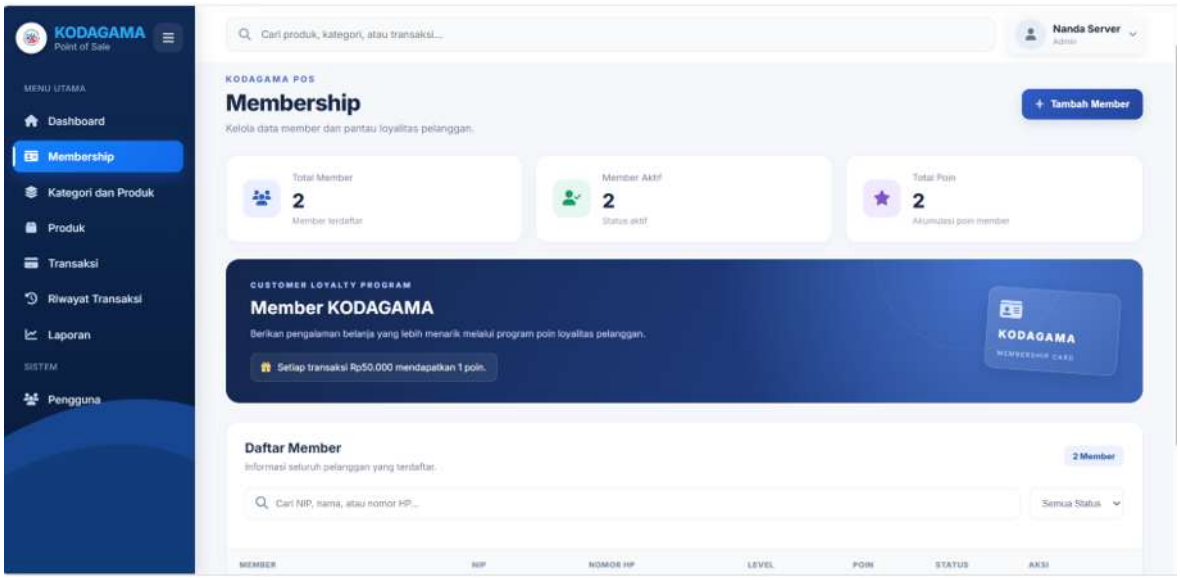

Figure 10. Membership page with summary cards, registration action, and member-list controls.

The Member class retains earning and redemption rules. Its earning calculation can carry unused spending toward a later threshold, while redemption checks the minimum, permitted increment, available points, and applicable cap. The interface summary is a view of stored values; a point update should be reconciled with the associated invoice when that behavior is tested.

4.4. Category and product-type management

Figure 11 shows category input and the existing KODAGAMA and KONSINYASI rows with edit and delete actions. Classification helps distinguish the product's retail grouping from its product type. In the logical model, categories and store_categories provide these classifications to products. Maintaining the classification before checkout supports consistent filtering and interpretation of the product list.



Figure 11. Implemented category-management region and its maintenance actions.

4.5. Product list and maintenance

Figure 12 displays product names, categories, product types, prices, stocks, and row actions. The first visible item, "Susu," is shown at IDR 7,000 with stock 94. The search field supports retrieval, while "Tambah Produk" provides the entry point for a new item. These fields supply the catalogue used during checkout; a product's current data should be checked before it is used as a test fixture.

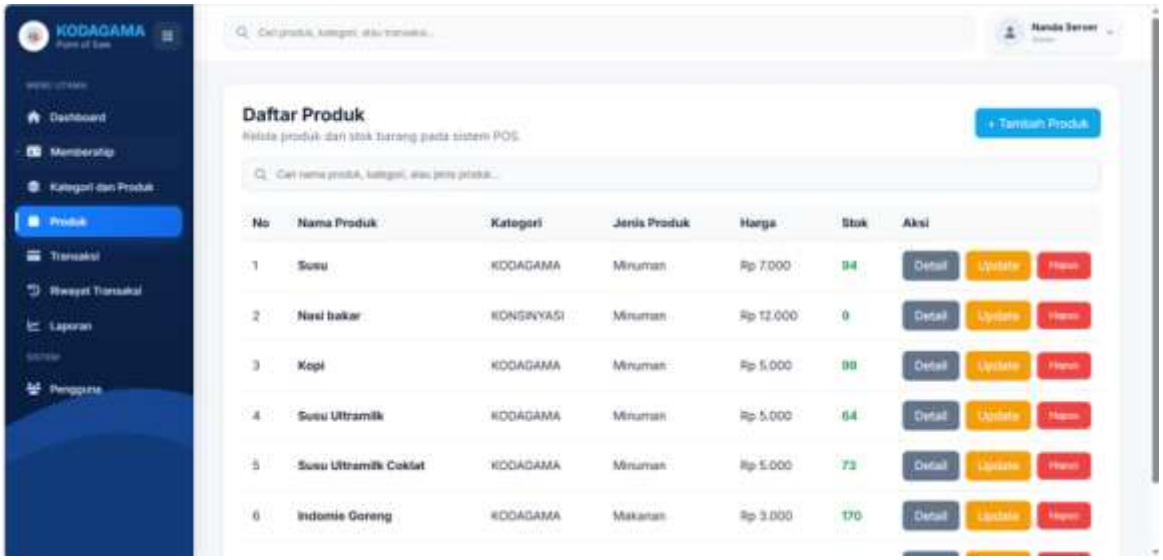

Figure 12. Product list with classification, selling price, stock, and maintenance actions.

4.6. Transaction interface

Figure 13 shows product selection alongside an empty transaction basket. The page combines the search field, product-type filters, product cards, subtotal, discount, amount due, optional member lookup, and payment choices. The captured empty basket has zero items and a zero total. Adding products changes the basket used to construct the invoice; it is a different activity from retrieving a completed invoice in history.

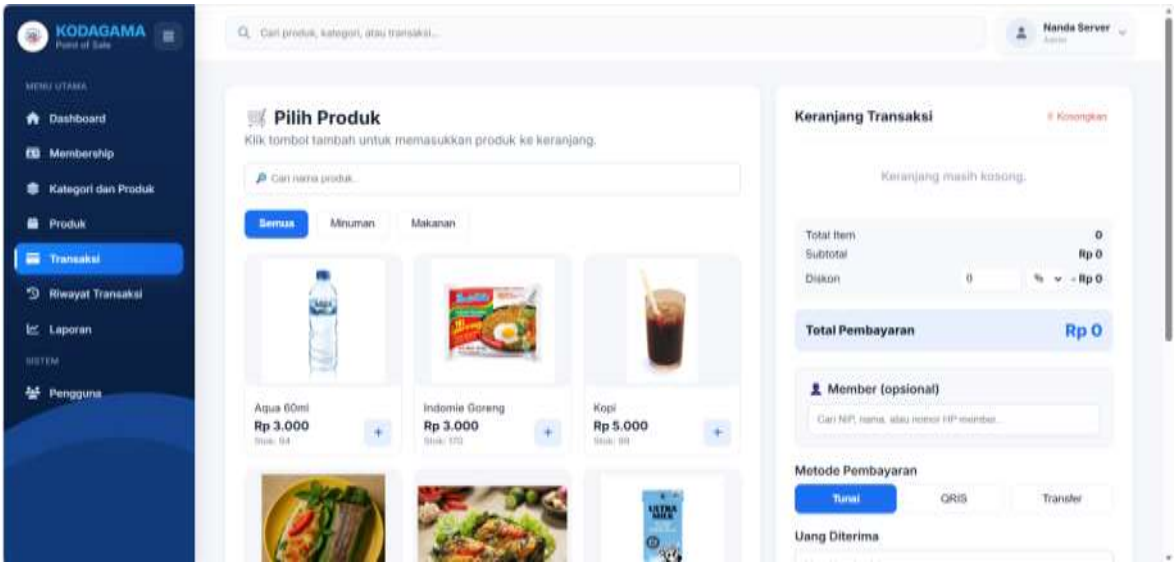

Figure 13. Implemented checkout with product catalogue, basket, optional membership, and payment choices.

4.7. Transaction history

Figure 14 separates invoice retrieval from the active checkout page. Search and filters identify invoice number, payment status, and method. Each visible row provides the member association where present, total, method, status, date, and a detail action. The capture includes cash, QRIS, and transfer records marked “LUNAS.” Displaying the method and status separately supports inspection of how a transaction was recorded and whether it is awaiting settlement.

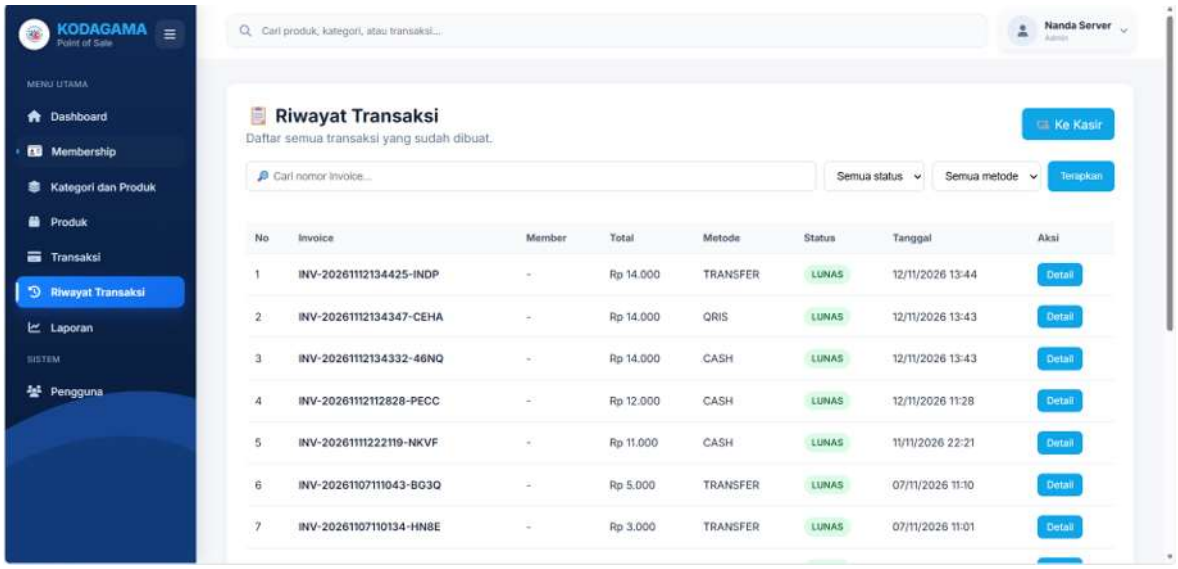

Figure 14. Transaction-history page with search, filters, status, and invoice-detail actions.

4.8. Sales report

Figure 15 shows the period from 22 October to 20 November 2026 and states that only paid transactions are counted. The visible total is IDR 183,000 across 22 transactions, with an IDR 8,318 average after rounding. The method amounts of IDR 47,000, IDR 52,000, and IDR 84,000 sum to that displayed total. This arithmetic explains the view; independent reconciliation still requires the same set of underlying invoices.

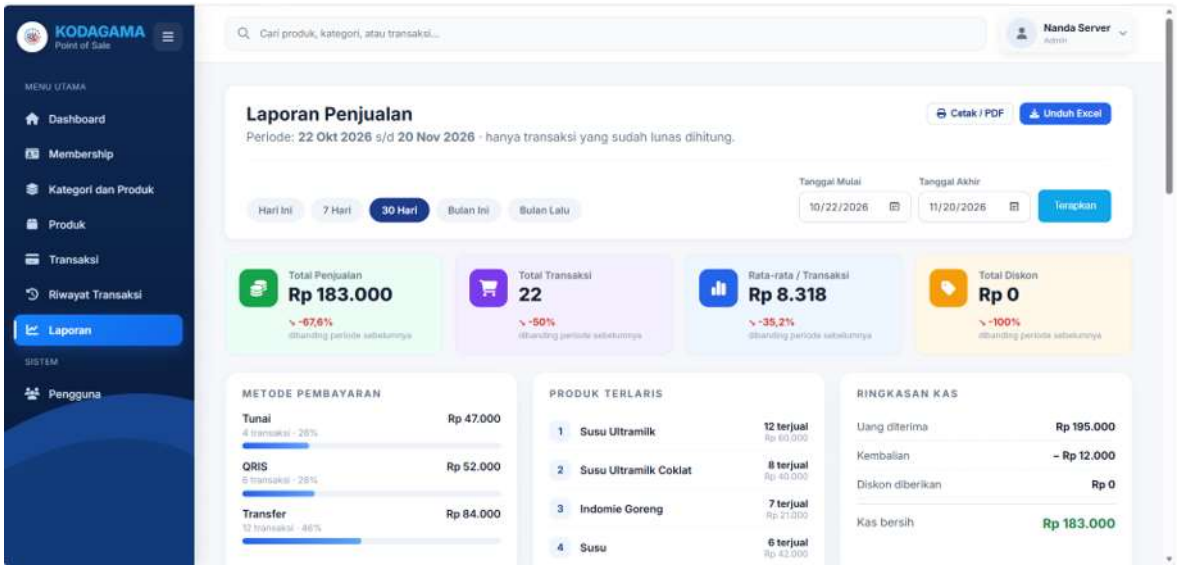

Figure 15. Period-based sales report with method breakdown, product ranking, and cash summary.

5. BLACK-BOX TESTING

5.1. Test approach and recorded verdicts

Black-box testing compares observable output with an expected result for a specified input or action. The functional records include ordinary operations, invalid input, and decisions at input boundaries. Equivalence partitioning groups inputs with the same expected behavior, while boundary-value analysis focuses on values around a change in that behavior [17], [18]. For checkout, quantities below, at, and above the available stock provide useful boundary cases [19], [20]. Related inventory and sales studies illustrate how these techniques can organize representative tests [25], [26], [13].

The unit of evidence is one case with its input, expected output, recorded result, and verdict. A correct rejection is a matching case: for example, insufficient cash should not complete a cash sale. The report documents 172 cases across the system areas in Table 3. Of these, 170 were recorded as matching

expectations and two did not. The mismatches concern “Remember me” and password recovery in an earlier sign-in interface.

Table 3. Documented black-box verdicts by system area

System area	Cases	Matched	Did not match
Sign-in	15	13	2
Administrator dashboard	14	14	0
Membership	9	9	0
Product types	8	8	0
Categories	9	9	0
Administrator product management	14	14	0
Administrator transactions	15	15	0
Administrator sales reports	15	15	0
User management	9	9	0
Search	10	10	0
Cashier checkout and basket	22	22	0
Cashier product list	12	12	0
Cashier sales reports	20	20	0
Total	**172**	**170**	**2**

5.2. Interpretation of the summary

The counts preserve the original records rather than certifying the current deployment. Password recovery was removed from a later interface, but removal does not change the earlier verdict into a passing retest. Likewise, a screenshot of a later login page does not replace the need to execute the relevant session and preference checks again. A later test record should identify the build, inputs, expected behavior, and observed output.

Case totals also do not measure the percentage of code branches or the probability of failure in daily use. More recorded cases for checkout than for membership do not establish that checkout is more reliable. The summary is useful because it identifies the tested areas and the recorded exceptions. Table 4 retains the detail needed to interpret representative cases within those areas.

5.3. Representative functional cases

Table 4 includes the two sign-in mismatches and selected cases for membership, cashier search, discounts, cash, payment confirmation, stock, and reports. “Matched” preserves the source verdict. A brief recorded outcome is retained as a brief outcome rather than being expanded into a more specific application response that was not recorded.

Table 4. Selected black-box test cases and recorded outcomes

Test case	Input or action	Expected result	Recorded result	Verdict
Incorrect password	Enter an incorrect password.	The application rejects sign-in.	Matched the expected outcome, as recorded.	Matched
Remember me	Select “Remember me” on the earlier interface.	The application applies the sign-in preference.	Outcome did not match; details were not recorded.	Did not match
Password recovery	Select the password recovery link on the earlier interface.	The recovery page opens.	Outcome did not match; details were not recorded.	Did not match
Duplicate member ID	Enter a previously registered membership identifier (NIP).	The application rejects the duplicate.	Matched the expected outcome, as recorded.	Matched

Test case	Input or action	Expected result	Recorded result	Verdict
Cashier product search	Enter “Indomie” in the product search field.	The application lists Indomie Goreng.	The test record confirms that the product was found.	Matched
Percentage discount	Apply a 10% discount at checkout.	The application reduces the amount due by 10%.	The recorded amount due changed as expected.	Matched
Cash change	For an IDR 15,000 sale, enter IDR 20,000 tendered.	The application calculates IDR 5,000 change.	The record reports correct change.	Matched
Insufficient cash	For an IDR 15,000 sale, enter IDR 10,000 tendered.	The application prevents processing or warns that the amount is insufficient.	The application displayed a validation response.	Matched
Payment confirmation	Select Confirm Payment.	The application updates the payment status.	Marked as matching; the final status value was not recorded.	Matched
Stock deduction	Complete a sale of two units of Aqua.	Stock decreases by two units after the sale succeeds.	The record notes an automatic stock update; it does not give paired balances.	Matched
Insufficient stock	Request more units than are available.	The application prevents the requested quantity from being added.	The record reports that stock validation worked.	Matched
Sales report	Open the sales total for 16 Sep–15 Oct 2026.	The report totals paid transactions for the chosen period.	The displayed total was IDR 1,620,600.	Matched

The stock-deduction case reports an automatic update but lacks paired opening and closing balances. The report case records a displayed total for its stated period without a complete invoice-level reconciliation. For those cases, a stronger execution record would retain the invoice identifier and the corresponding before-and-after data from the same run.

6. USER ACCEPTANCE TESTING

6.1. Purpose and current status

UAT is specified as a follow-up assessment of whether staff can complete the cooperative’s agreed operational tasks. Administrator and cashier representatives are the proposed evaluators because their responsibilities differ. The assessment should use an agreed application build, product data, member data, and transaction set. Completed UAT results are not yet available. Table 5 therefore defines planned scenarios and acceptance criteria; it does not report that the partner has accepted the system.

The session should begin by confirming the test data and the evaluator’s role. The evaluator performs the task through the interface, compares the outcome with the criterion, and records any problem that prevents completion. Where an amount is part of the criterion, the catalogue prices used in that session must match the fixture. For U05, the illustrated prices produce IDR 11,000 from one bottled water, one instant noodle, and one coffee, so IDR 20,000 tendered should yield IDR 9,000 change.

Table 5. Planned UAT scenarios, staff roles, and acceptance criteria

ID	Proposed evaluator	Operational scenario	Acceptance criterion
U01	Admin / cashier	Sign-in and navigation	Staff can sign in and locate the operational screen assigned to their role.

ID	Proposed evaluator	Operational scenario	Acceptance criterion
U02	Admin	Dashboard review	The evaluator can locate the required summary, period graph, and low-stock information and explain their displayed scope.
U03	Admin	Products and categories	Create or revise an item, then verify its classification, selling price, and stock in the product list.
U04	Admin / cashier	Member registration and lookup	Register a member as administrator and retrieve the same member during checkout.
U05	Cashier	Cash checkout	For the agreed IDR 11,000 fixture, record IDR 20,000 tendered and verify the invoice and IDR 9,000 change.
U06	Cashier	Noncash confirmation	Create a pending QRIS or transfer invoice, verify receipt, confirm payment, and inspect the resulting status.
U07	Admin / cashier	Transaction retrieval	Retrieve the intended invoice through search or filters and inspect its method, status, and detail.
U08	Admin / cashier	Sales reporting	Choose an agreed period, reconcile the included paid invoices, and review the required print or export output.

6.2. Recording and acceptance decision

Each execution record should contain the date, build, evaluator, scenario ID, actual outcome, evidence reference, decision, and remarks. The proposed acceptance rule is that agreed mandatory tasks meet their criteria and unresolved issues are reviewed with the partner before sign-off. A blocked task should identify the observed problem and the retest needed after correction. No satisfaction percentage, respondent count, acceptance score, or sign-off is inferred from the screenshots or from the black-box totals.

7. DISCUSSION

7.1. Connection between design and implementation

The design views explain the same transaction at different levels. The use case diagram identifies responsibility, the activity diagram distinguishes payment alternatives, and the sequence diagram locates the cash-processing writes. The class diagram links the relevant controller and models, while the ERD explains how the invoice retains its lines and optional member association. Architecture locates the request, rendering, and storage paths. The wireframe and full screenshots make those relationships visible at the staff interface.

The invoice is the main link between the modules. The catalogue supplies products and prices; sale lines retain the selected quantities and unit prices; the invoice retains the final amounts, payment method, and status. Membership adds optional benefits without replacing the ordinary sale. In the reviewed checkout logic, general discounts precede level-based discounts and point redemption. Product lookup, member selection, and sufficient payment must therefore be considered together when an integrated sale is tested.

7.2. Settlement consistency and further verification

A pending invoice should leave stock unchanged, while first settlement should deduct the sold quantity once. The reviewed cash handler groups related writes inside a database transaction. The reviewed noncash confirmation handler instead changes invoice status before deducting stock and awarding points, without an explicit enclosing transaction or renewed stock check. Its ordinary repeat guard does not establish behavior when two requests read the same pending state concurrently. These are implementation risks inferred from the reviewed source rather than incidents observed at the cooperative.

Further execution should retain paired invoice and inventory records after a normal confirmation, a sequential repeat, concurrent requests, a competing sale, and an interrupted write. The intended result is consistent completion or rollback of related records. Transaction isolation and row-locking reads are relevant mechanisms for examining these conditions [23], [24]. The same build and data should be used when relating a functional retest to the subsequent UAT task.

7.3. Scope and limitations

The case covers one cooperative, documented functional records, a source snapshot, and supplementary screenshots. A visible counter or a recorded match does not provide an independent reconciliation of every

underlying invoice. Later interface changes require their own retest records. Staff-time savings, inventory-accuracy improvement, usability, and performance were not measured and need separate evaluation criteria [25]. UAT is specified here as a protocol; its results and partner acceptance decision remain unavailable.

8. CONCLUSION

POS KODAGAMA connects product, inventory, sales, and optional membership data through the invoice and payment state. The article presents all seven design views and eight full implementation screenshots, followed by the documented black-box cases and a staff acceptance protocol. The functional record contains 170 matches and two sign-in mismatches among 172 cases. Eight UAT scenarios define the proposed operational assessment, without claiming completed acceptance. Follow-up work should verify settlement consistency and retain the staff execution records needed for the partner's decision.

REFERENCES

[1] N. DeHoratius and A. Raman, "Inventory record inaccuracy: An empirical analysis," Management Science, vol. 54, no. 4, pp. 627–641, 2008, doi: 10.1287/mnsc.1070.0789.

[2] B. C. Hardgrave, J. A. Aloysius, and S. Goyal, "RFID-enabled visibility and retail inventory record inaccuracy: Experiments in the field," Production and Operations Management, vol. 22, no. 4, pp. 843–856, 2013, doi: 10.1111/poms.12010.

[3] K. R. Putratama, S. Alfiansyah, S. F. Al Farizi, and W. Haryono, "Sistem inventory koperasi berbasis website di Koperasi SMP PGRI 2 Ciledug," Jurnal Ilmu Komputer dan Teknologi Informasi, vol. 2, no. 2, pp. 109–118, 2025, doi: 10.63447/jikti.v2i2.1534.

[4] R. E. Anugrah, Y. A. Saputra, and W. Haryono, "Perancangan sistem inventory berbasis web untuk optimalisasi manajemen persediaan barang di PT Bumi Daya Plaza," Bridge: Jurnal Publikasi Sistem Informasi dan Telekomunikasi, vol. 2, no. 4, pp. 342–363, 2024, doi: 10.62951/bridge.v2i4.317.

[5] A. R. Prasetya, D. H. Putra, W. Kencana, and W. Haryono, "Perancangan sistem inventory berbasis web dengan barcode scanner untuk pengelolaan barang pada CV D'Lantik Art Project menggunakan metode Agile," Buletin Ilmiah Informatika Teknologi, vol. 4, no. 1, pp. 1–9, 2025, doi: 10.58369/biit.v4i1.136.

[6] M. R. Hasibuan, F. Gultom, and M. B. K. Nasution, "PHP-based stationery sales information system at Ufaira Printing Sigambal, Labuhanbatu," Journal of Engineering and Science Application, vol. 3, no. 2, pp. 190–197, 2026, doi: 10.69693/jesa.v3i2.57.

[7] C. C. Utami, R. I. Desanti, and F. A. Halim, "The development of web-based sales and inventory system for a stationary store," Ultima InfoSys, vol. 14, no. 2, pp. 52–57, 2023, doi: 10.31937/si.v14i2.3358.

[8] F. Azizah, L. Novyanti, N. O. Amri, A. Oktaviani, and R. Nurfalah, "Perancangan aplikasi Point of Sale Pandita Coffee berbasis web dengan kombinasi model SDLC Waterfall," Jurnal Sistem Informasi Kaputama, vol. 6, no. 1, pp. 69–74, 2022, doi: 10.59697/jsik.v6i1.184.

[9] S. Suprianto, M. Fadlan, and D. Prayogi, "Perancangan aplikasi Point of Sale berbasis web pada Toko Project Salfa Tarakan," Sebatik, vol. 25, no. 2, pp. 624–631, 2021, doi: 10.46984/sebatik.v25i2.1519.

[10] I. D. G. A. Pandawana, B. P. P. Putra, I. P. G. Budayasa, I. D. M. A. B. Joni, and P. A. Permata Sari, "Pengembangan Point of Sale berbasis web untuk pengelolaan transaksi apotek," Jurnal Ilmiah Teknik Universitas Mahasaraswati Denpasar, vol. 6, no. 1, pp. 59–64, 2026, doi: 10.36733/jitumas.v6i1.14027.

[11] H. Aliazki Pratama and B. Yulisa Geni, "Web-based POS system to support perfume sales transactions using the Waterfall model," Journal of Information Systems and Technology Research, vol. 5, no. 3, pp. 343–355, 2026, doi: 10.55537/jistr.v5i3.1535.

[12] A. A. Amalia, E. Supriyati, and T. Listyorini, "Design of a web-based Point of Sale system for transaction recording at Sumber Rejeki Store," SAGA: Journal of Technology and Information System, vol. 3, no. 3, 2025, doi: 10.58905/saga.v2025i3.547.

[13] R. F. Agustio, A. I. Baharianto, R. P. Mulia, and W. Haryono, "Perancangan sistem inventory dan transaksi pembelian barang berbasis web dengan metode Waterfall," Jurnal RESTIKOM: Riset Teknik Informatika dan Komputer, vol. 6, no. 3, pp. 554–564, 2024, doi: 10.52005/restikom.v6i3.385.

[14] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," Empirical Software Engineering, vol. 14, pp. 131–164, 2009, doi: 10.1007/s10664-008-9102-8.

[15] W. W. Royce, "Managing the development of large software systems," in Proc. IEEE WESCON, 1970, pp. 1–9. [Online]. Available: https://cse.msu.edu/~cse435/Homework/HW3/royce1970.pdf

[16] Object Management Group, OMG Unified Modeling Language, ver. 2.5.1, 2017. [Online]. Available: https://www.omg.org/spec/UML/2.5.1/

[17] Bank Indonesia, "Quick Response Code Indonesian Standard (QRIS)." [Online]. Available: https://www.bi.go.id/id/fungsi-utama/sistem-pembayaran/ritel/kanal-layanan/QRIS/default.aspx

[18] Laravel, "Database: Getting started," Laravel 12.x Documentation. [Online]. Available: https://laravel.com/docs/12.x/database

[19] J. Gray, "The transaction concept: Virtues and limitations," in Proc. 7th Int. Conf. Very Large Data Bases, 1981, pp. 144–154. [Online]. Available: https://www.vldb.org/conf/1981/P144.PDF

[20] Oracle, "START TRANSACTION, COMMIT, and ROLLBACK statements," MySQL 8.4 Reference Manual. [Online]. Available: https://dev.mysql.com/doc/refman/8.4/en/commit.html

[21] ISTQB, Certified Tester Foundation Level Syllabus, ver. 4.0.1, 2024. [Online]. Available: https://istqb.org/wp-content/uploads/2024/11/ISTQB_CTFL_Syllabus_v4.0.1.pdf

[22] Software and systems engineering—Software testing—Part 4: Test techniques, ISO/IEC/IEEE 29119-4:2021, 2021.

[23] F. Dobslaw, R. Feldt, and F. Gomes de Oliveira Neto, "Automated black-box boundary value detection," PeerJ Computer Science, vol. 9, Art. no. e1625, 2023, doi: 10.7717/peerj-cs.1625.

[24] A. Z. Rafif et al., "Functional testing of web-based input form using boundary value analysis technique: A case study on TemanTernak," Jurnal Teknologi dan Open Source, vol. 8, no. 2, pp. 1064–1070, 2025, doi: 10.36378/jtos.v8i2.5084.

[25] A. Samdono, A. P. Sari, and F. P. Aditiawan, "Pengujian Black Box pada sistem informasi stok dan penjualan berbasis website menggunakan metode equivalence partitioning: Studi kasus CV. Algani Karya Mandiri," JATI (Jurnal Mahasiswa Teknik Informatika), vol. 8, no. 1, 2024, doi: 10.36040/jati.v8i1.8893.

[26] M. Sutomo and F. E. B. Simamora, "Pengujian black box aplikasi inventaris dengan Boundary Value Analysis di PT Duo Roda Motor," Device: Journal of Information System, Computer Science and Information Technology, vol. 5, no. 1, pp. 202–216, 2024,

doi: 10.46576/device.v5i1.4557.
[27] H. Berenson, P. Bernstein, J. Gray, J. Melton, E. O'Neil, and P. O'Neil, "A critique of ANSI SQL isolation levels," in Proc. ACM SIGMOD Int. Conf. Management of Data, 1995, pp. 1–10, doi: 10.1145/223784.223785.
[28] Oracle, "Locking reads," MySQL 8.4 Reference Manual. [Online]. Available: https://dev.mysql.com/doc/refman/8.4/en/innodb-locking-reads.html
[29] Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—Product quality model, ISO/IEC 25010:2023, 2023.