Volume XI Issue 3 Year 2026 | Page 754-766 | ISSN: 2527-9866
Received: ***10-06-2026*** | *Revised:* ***15-07-2026*** | *Accepted:* ***30-07-2026***

Application of RCA and AHP to Identify Functional Testing Failures in Loan Origination Systems by QA Engineers

Made Chandra Wahyudi[1], I Made Candiasa [2], Luh Joni Erawati Dewi [3]
[1,2,3] Postgraduate Programs, Computer Science Study Program, Universitas Pendidikan Ganesha, Bali, Indonesia
email: chandra@student.undiksha.ac.id[1], candiasaimade@undiksha.ac.id[2], joni.erawati@undiksha.ac.id [3]
*Correspondence: chandra@student.undiksha.ac.id

Abstract: *Recurring functional testing failures in DJOIN's digital banking systems during 2023–2025 created operational risks, particularly in the Balance Validation Dashboard and financial transaction modules. This study aimed to identify the principal causes of recurring failures and prioritize appropriate improvement solutions by integrating Root Cause Analysis (RCA) and the Analytical Hierarchy Process (AHP). The analysis used 1,759 production bug tickets, comprising 1,724 tickets from the Coopmax system and 35 tickets from the KOCEK loan origination system. RCA was conducted using a Fishbone Diagram based on the 6M framework and 5 Whys Analysis, supported by bug-tracking records, system logs, testing documentation, and stakeholder interviews. The analysis identified 12 root causes. The three highest-severity causes were hardcoded product-type assumptions in the Balance Validation Dashboard query, with a severity score of 4.67; the absence of a regression test suite after hotfix implementation, with a score of 4.33; and distributed system logs that complicated failure tracing, with a score of 4.33. AHP evaluation assigned the highest weight to solution effectiveness at 42.44%, followed by business impact at 30.65%, implementation cost at 13.07%, implementation time at 8.36%, and technical complexity at 5.48%. The comparison matrix achieved a consistency ratio of 0.024, indicating acceptable judgment consistency. Among 12 alternatives, Automated Testing Tools ranked first with a priority score of 0.1872, followed by Test-Driven Development at 0.1412 and CI/CD Implementation at 0.1372. These findings provide an evidence-based prioritization of proposed corrective and preventive actions; however, their actual effectiveness requires post-implementation evaluation.*

Keywords: *RCA, AHP, Functional Testing, Loan Origination, QA Engineer*

1. Introduction

Digital banking systems increasingly integrate customer data, transaction processing, credit assessment, accounting, reporting, and third-party services within interconnected software environments. A Loan Origination System (LOS) supports the lending process from application and borrower-data validation to credit assessment, approval, and disbursement. Because these activities depend on accurate business rules, data exchange, and service integration, functional defects may disrupt operational processes and generate incorrect financial outputs. Consequently, software quality assurance is essential for maintaining the accuracy, availability, and operational resilience of digital banking services [1] [2].

Functional testing verifies whether system behavior conforms to business requirements and expected user outcomes. In complex banking applications, however, identifying a defect is insufficient when similar failures recur after corrective releases. Modern distributed systems may produce failures involving multiple services, databases, application programming interfaces, configuration changes, and deployment environments. Such dependencies make failure diagnosis difficult because the visible error may only represent a symptom of a deeper technical or procedural problem. Recent studies on software Quality Assurance and microservice failure diagnosis therefore emphasize causal reasoning, evidence from logs and

traces, and systematic Root Cause Analysis (RCA) to distinguish root causes from secondary symptoms [3] [4] [5] [6].

RCA provides a structured mechanism for tracing recurring defects to their underlying causes and formulating corrective actions. Techniques such as Fishbone Diagrams and 5 Whys Analysis are particularly useful for examining interactions among human, process, technology, data, measurement, and environmental factors. Previous research has also applied causal and cyber-physical analysis to complex software failures, demonstrating the importance of examining relationships among system components rather than evaluating incidents in isolation [7]. Nevertheless, RCA does not automatically determine which corrective action should be implemented first when an organization faces limited personnel, budget, and deployment time [8].

The Analytical Hierarchy Process (AHP) addresses this prioritization problem by comparing decision criteria and solution alternatives through pairwise judgments. AHP has been applied to software quality measurement, requirements prioritization, and failure-criticality assessment because it transforms expert judgments into quantitative priority weights and evaluates their consistency [9]. However, previous studies have generally examined RCA and AHP separately or applied them outside the specific context of recurring functional testing failures in digital banking. Limited empirical research has integrated ticket-level RCA evidence with AHP-based solution prioritization while explicitly linking each proposed solution to the root causes identified in interconnected core-banking and loan-origination applications. This creates a methodological gap between diagnosing why failures occur and determining which corrective action should receive implementation priority [10].

This study addresses that gap through a case study of two applications within DJOIN's digital banking ecosystem. Coopmax supports broader cooperative banking operations, including dashboard, savings, accounting, lending, reporting, billing, and related operational functions, whereas KOCEK specifically supports loan-origination activities. The dataset contains 1,759 production bug tickets recorded from 2023 to 2025, comprising 1,724 tickets from Coopmax and 35 tickets from KOCEK. Therefore, the empirical findings predominantly represent recurring functional failures in Coopmax, while KOCEK tickets provide supporting evidence concerning loan-origination and third-party integration failures. This distinction is necessary to avoid treating all analyzed tickets as failures originating exclusively from a standalone loan origination system The module distribution further shows that Coopmax includes dashboard, savings, accounting, loan, reporting, billing, and mobile-banking functions.

The study focuses on recurring failures associated with the Balance Validation Dashboard, financial calculation processes, bug triage, regression testing, system logging, and external-service integration. RCA is used to identify and assess the principal technical and procedural causes, while AHP is used to prioritize 12 proposed improvement alternatives according to business impact, implementation cost, implementation time, solution effectiveness, and technical complexity. The contribution of this research is an evidence-based RCA–AHP workflow that connects historical defect records, validated root causes, severity assessment, decision criteria, and proposed corrective actions within a digital banking Quality Assurance context. The study is limited to identifying root causes and prioritizing proposed solutions; the actual effectiveness of those solutions must be established through post-implementation evaluation.

2. Literature Review

Definition of Root Cause Analysis

RCA is a systematic method used to identify the main cause of a problem or event. RCA is commonly applied in industries that require high quality standards, such as manufacturing, information technology, and banking[11]. In the context of functional testing for a loan origination system, RCA helps QA Engineers identify the root causes of testing failures, which may involve technical errors, system design issues, or human error[12].

Definition of the Analytical Hierarchy Process

The AHP is a decision-making method that helps determine priorities based on multiple criteria. This method was developed by Saaty (1980) and has been applied in various contexts, including software testing. AHP works by breaking down complex problems into several hierarchical levels, where each factor is assigned a relative weight or priority compared to the others. In the context of functional testing, AHP can be used to determine the priority of factors that influence system failures, such as code complexity, testing tool capability, and testing time. Studies on software quality measurement show that the use of AHP in measuring software quality, particularly in functional suitability, performance efficiency, and reliability, can help improve decision-making related to repair prioritization and risk mitigation in the software development lifecycle[13]. AHP also enables QA teams to select the most efficient improvement actions based on the weights assigned to the causal factors that have been identified through RCA[14].

Previous Studies

No.	Author(s) and Year	Research Title/Focus	Method Used	Main Findings
1	Cheng, Z., Choi, H., Shen, J., et al. (2024)	*ROCAS: Root Cause Analysis of Autonomous Driving Accidents via Cyber-Physical Co-Mutation [15]*	Root Cause Analysis (RCA) and Cyber-Physical Co-Mutation Analysis	This study showed that RCA based on cyber-physical co-mutation could improve the accuracy of root cause identification by up to 48% compared to traditional methods. The study focused on complex system interactions and automated failure analysis in autonomous driving systems.
2	Sarwosri, S., Rochimah, S., Yuhana, U. L., & Hidayat, S. B. (2023)	*Software Quality Measurement for Functional Suitability, Performance Efficiency, and Reliability Characteristics Using AHP [16]*	Analytical Hierarchy Process (AHP) based on ISO 25010 standards	This study showed that AHP can be used to prioritize factors affecting software quality, particularly functional suitability, performance efficiency, and reliability. AHP supports a more structured decision-making process in software quality assessment.
3	Cheng, Z., Choi, H., et al. (2024)	*Cyber-Physical Security in Financial Systems*	Cyber-Physical Security Analysis and Co-Mutation Approach	This study discussed the importance of rigorous testing in financial systems to reduce operational risks and improve system reliability. The cyber-physical co-mutation approach was used to analyze the relationship between security aspects, system failures, and the reliability of financial services.

Research Contributions

Based on the review of previous studies, this research provides several main contributions.

First, this study develops an integrated framework that combines Root Cause Analysis (RCA) and the Analytical Hierarchy Process (AHP) to analyze and prioritize solutions for functional testing failures in loan origination systems. The integration of these two methods is an important contribution because RCA is used to identify the root causes of failures, while AHP is used to determine the most prioritized improvement solutions based on objective criteria.

Second, this study contributes to the specific context of digital banking, particularly loan origination systems. Most previous studies have focused on general software systems, autonomous driving, or software quality measurement based on specific standards. This study fills this gap by positioning QA Engineers as key actors in the process of failure identification, root cause analysis, and improvement prioritization in high-risk financial systems.

Third, this study offers a holistic approach by combining corrective and preventive aspects. The corrective aspect is carried out through RCA to identify the root causes of failures that have already occurred, while the preventive aspect is carried out through AHP to determine improvement solutions that can prevent similar failures in the future. Therefore, this study is not only reactive to bugs or incidents, but also supports the establishment of a more systematic and sustainable QA process.

Fourth, this study provides practical contributions to the digital banking industry through the formulation of technical mitigation strategies and recommendations for improving the Testing Life Cycle SOP. The output of this study does not stop at problem identification or priority ranking, but is also directed toward improvement actions that can be implemented by QA, Software Engineer, Product, and DevOps teams.

Thus, the main difference between this study and previous studies lies in the integration of RCA and AHP, the focus on functional testing failures in loan origination systems, and the outputs in the form of solution priorities, technical mitigation strategies, and SOP improvement recommendations to enhance the quality assurance process in digital banking.

3.Methods

This study applies a decision-oriented evaluation model with a formative approach to identify the root causes of functional testing failures and determine prioritized improvement solutions for DJOIN's loan origination system. The research integrates RCA and the AHP, where RCA is used to analyze the fundamental causes of system failures, while AHP is used to rank improvement alternatives based on multiple decision-making criteria.

The research begins by identifying functional testing failures that occur when system outputs do not meet requirements, specifications, or user expectations. These failures include requirement mismatches, system errors or unhandled exceptions, malfunctioning features, data inconsistencies, and integration failures with third-party services. The identified failures are then mapped based on type, severity, affected modules, and frequency of occurrence[17], [18].

Data are collected from several sources, including the OpenProject bug tracking system, system logs, error logs, operational reports, and documentation of manual and automated testing. Structured interviews with QA Engineers, Software Engineers, Product teams, and DevOps teams are also conducted to validate the identified failures and obtain additional information regarding severity, business impact, and causal factors.

Root Cause Analysis is conducted using three techniques: Fishbone Diagram, 5 Whys Analysis, and Cyber-Physical Co-Mutation Analysis[19]. The Fishbone Diagram classifies causes based on the 6M framework, namely Man, Method, Machine, Material, Measurement, and Mother

Nature. The 5 Whys Analysis is used to trace cause-and-effect relationships until the main root causes are found. Meanwhile, Cyber-Physical Co-Mutation Analysis is applied to identify causal relationships, co-occurrence patterns, temporal failure sequences, and failure propagation across system components[20].

The results of RCA are then transformed into input for AHP. The identified root causes are used to formulate AHP criteria, solution alternatives, and priority weights. The AHP hierarchy consists of the main goal, evaluation criteria, sub-criteria, and solution alternatives[21]. The main objective is to determine the most appropriate improvement solutions to enhance the quality of functional testing in DJOIN's loan origination system.

The AHP criteria include business impact, implementation cost, implementation time, solution effectiveness, and technical complexity. These criteria are used to evaluate twelve solution alternatives grouped into four categories: human factors, system and tools, process and method, and communication and documentation. Each alternative is assessed comprehensively to determine its implementation priority[22].

The AHP calculation is performed using a pairwise comparison matrix based on Saaty's 1–9 scale. Each criterion and solution alternative is compared to obtain priority weights. The consistency ratio is then used to ensure the reliability of evaluator judgments. The highest-ranked solutions become the basis for preparing corrective and preventive actions[23]. Data analysis is conducted descriptively and quantitatively. Descriptive analysis is used to explain failure patterns, causal categories, and relationships between root causes. Quantitative analysis is used in the AHP weighting process and in evaluating solution effectiveness through pre-test and post-test comparisons of bug recurrence rates[24]. Data validity is ensured through multi-expert peer review involving QA Engineers and relevant stakeholders. Data reliability is tested through triangulation by comparing RCA results, system logs, interviews, historical QA reports, and testing documentation. In addition, consistency ratio and inter-rater reliability are used to strengthen the validity of AHP results.

Overall, this study produces an integrated RCA-AHP framework consisting of four phases: problem identification and RCA execution, RCA output processing, AHP implementation based on RCA input, and integrated decision-making. This framework is expected to support objective, measurable, and applicable solution prioritization to improve functional testing quality and reduce recurring failures in digital banking systems[25].

3. Results and Discussion

Results of Functional Testing Failure Identification

Functional testing failure identification was conducted using bug data from DJOIN's OpenProject bug tracking system during the 2023–2025 period. A total of 1,759 production bug tickets were identified, consisting of 1,724 tickets from the Coopmax system and 35 tickets from the KOCEK loan origination system. The distribution of production bugs is presented in Table 1.

Table 1. Distribution of Production Bugs in DJOIN Systems, 2023–2025

Year	System	Total Bugs	Closed	Unresolved	Unresolved Rate	Dominant Card Type
2023	Coopmax	547	491	56	10.2%	Hotfix: 410 tickets, 74.9%
2024	Coopmax	739	354	385	52.1%	"Pilih": 452 tickets, 61.2%
2025	Coopmax	438	434	4	0.9%	"Pilih": 438 tickets, 100%
2025	KOCEK	35	N/A	N/A	N/A	Bug Report
Total		1,759				

The data show that bug volume increased by 35.1% from 2023 to 2024. The unresolved rate also increased significantly from 10.2% in 2023 to 52.1% in 2024, indicating a serious issue in bug triage and resolution capacity. Although the unresolved rate decreased to 0.9% in 2025, the remaining bug volume was still high, indicating that the underlying causes had not been fully eliminated.

Distribution of Failures by System Module

The bug distribution by module shows that the Dashboard module had the highest number of bugs and continued to increase during the three-year observation period. This finding supports the research focus on the Balance Validation Dashboard.

Table 2. Distribution of Bugs by System Module, 2023–2025

Rank	Module	2023	2024	2025	Total	Percentage	Trend
1	Dashboard	71	104	142	317	18.4%	Continuously increasing
2	Savings	73	97	60	230	13.4%	Peak in 2024
3	Accounting	66	66	63	195	11.3%	Consistently high
4	Loan	62	76	37	175	10.2%	Peak in 2024
5	Reports	66	63	42	171	9.9%	Gradual decrease
6	Billing	19	37	7	63	3.7%	Fluctuating
7	Mobile Banking	16	18	19	53	3.1%	Stable
8	Members	15	21	10	46	2.7%	Fluctuating
9	Others	159	272	60	491	28.5%	Distributed

The Dashboard module recorded 317 bugs, or 18.4% of the total module-based bugs. This module also showed a continuous increase from 71 bugs in 2023 to 142 bugs in 2025, making it the most critical module for further analysis.

Root Cause Analysis Results

Root Cause Analysis was conducted using Fishbone Diagram, 5 Whys Analysis, and Cyber-Physical Co-Mutation Analysis. The analysis identified 12 root causes grouped into the 6M framework.

Table 3. Summary of Root Causes Based on the 6M Framework

Code	6M Category	Root Cause	Empirical Evidence
RC-1	Machine	Hardcoded product type assumption in DVS aggregation query	DVS bugs increased from 30 to 47 and 67 tickets
RC-2	Method	No regression test suite after hotfix	Repeated hotfix loop without regression validation
RC-3	Machine	Algorithm failure in financial transaction calculations	214 algorithm failure tickets in 2023–2024
RC-4	Method	QA not involved in requirement review	Product knowledge and user error tickets indicate interpretation gaps
RC-5	Machine	Server failure in Dashboard infrastructure	46 server failure tickets in 2023
RC-6	Method	Unstructured bug triage and categorization	452 tickets categorized as “Pilih” in 2024

RC-7	Material	Unclean migration data during new cooperative onboarding	54 data migration issue tickets in 2023
RC-8	Machine	Unreliable interest batch job process	Failed interest processing and accounting journal issues in Q3 2025
RC-9	Measurement	Distributed logs across services	Difficult root cause tracing and high unresolved rate
RC-10	Mother Nature	Multi-branch deployment amplifies single defect impact	One DVS bug generated 38 tickets in KSB branches
RC-11	Man	Inconsistent bug ticket metadata entry	High number of uncategorized tickets
RC-12	Mother Nature	Unstable third-party CLIK API	10 out of 35 KOCEK tickets related to CLIK

The RCA results indicate that Machine and Method categories dominated the root causes. This shows that the recurring failures were mainly caused by technical-systemic issues and process weaknesses, not merely by individual human errors.

5 Whys Analysis Summary

The 5 Whys Analysis was applied to three major cases: unbalanced DVS, algorithm failure, and the high unresolved rate in 2024.

Table 4. Summary of 5 Whys Analysis

Case	Main Evidence	Identified Root Cause	Interpretation
Unbalanced Balance Validation Dashboard	144 DVS tickets across 14 cooperatives	Hardcoded product type assumption and missing regression test	DVS logic was not product-agnostic and failed when new product types were added
Algorithm Failure in Financial Calculations	214 algorithm failure tickets in 2023–2024	Absence of shift-left testing	QA was not involved in requirement review, causing financial edge cases to be missed
High Unresolved Rate in 2024	385 unresolved tickets and 452 uncategorized tickets	No structured triage and capacity gap	Lack of mandatory categorization and SLA caused backlog and ad hoc handling

These findings show that recurring failures were not only technical, but also related to weak testing governance, limited regression validation, and insufficient QA involvement in earlier development phases.

Severity Assessment of Root Causes

Each root cause was assessed based on frequency, business impact, and resolution time using a Likert scale from 1 to 5.

Table 5. Severity Scores of Root Causes

Rank	Root Cause	Category	Frequency	Business Impact	Resolution Time	Severity Score
1	RC-1: Hardcoded DVS query	Machine	5	5	4	4.67
2	RC-2: No regression test	Method	5	5	3	4.33
3	RC-9: Distributed logs	Measurement	5	4	4	4.33
4	RC-3: Algorithm failure	Machine	5	4	3	4.00
5	RC-4: No shift-left testing	Method	4	4	4	4.00
6	RC-6: Unstructured triage	Method	4	3	4	3.67
7	RC-10: Multi-branch amplification	Mother Nature	4	5	2	3.67
8	RC-5: Server failure	Machine	4	4	2	3.33
9	RC-7: Data migration issue	Material	3	4	3	3.33
10	RC-8: Interest batch job issue	Machine	3	4	3	3.33
11	RC-11: Metadata inconsistency	Man	4	3	3	3.33
12	RC-12: CLIK API issue	Mother Nature	3	3	3	3.00

The three highest-severity root causes were RC-1, RC-2, and RC-9. These root causes were then used as the main input for determining the AHP criteria weights and prioritizing improvement alternatives.

Analytical Hierarchy Process Results

AHP was used to rank improvement solutions based on five criteria: business impact, implementation cost, implementation time, solution effectiveness, and technical complexity.

Table 6. AHP Criteria Weights

Rank	Code	Criterion	Weight	Empirical Justification
1	C4	Solution Effectiveness	42.44%	Repeated DVS hotfix loop proved that partial fixes were ineffective
2	C1	Business Impact	30.65%	DVS bugs affected multiple cooperatives and branches
3	C2	Implementation Cost	13.07%	Resource constraints required realistic investment decisions
4	C3	Implementation Time	8.36%	High backlog required quick wins
5	C5	Technical Complexity	5.48%	Complexity was not the main barrier if effectiveness and business impact were achieved

The consistency ratio was 0.024, which is below the 0.10 threshold. Therefore, the pairwise comparison results were considered consistent and valid for ranking the solution alternatives.

Table 7. AHP Ranking of Improvement Solutions

Rank	Code	Solution Alternative	Category	AHP Score	Root Causes Addressed
1	A4	Automated Testing Tools	System/Tool	0.1872	RC-1, RC-2, RC-3
2	A7	Test-Driven Development	Process	0.1412	RC-4, RC-2
3	A8	CI/CD Implementation	Process	0.1372	RC-2, RC-6, RC-9
4	A1	Comprehensive QA Training	Human	0.1143	RC-4, RC-11
5	A9	Test Case Management System	Process	0.0912	RC-6, RC-11
6	A5	Test Environment Upgrade	System/Tool	0.0896	RC-5, RC-7
7	A6	Enhanced Bug Detection Tools	System/Tool	0.0712	RC-9, RC-5
8	A12	System Documentation	Communication	0.0638	RC-4, RC-7
9	A2	Peer Review System	Human	0.0629	RC-11
10	A3	QA Best Practice Guidelines	Human	0.0618	RC-11, RC-4
11	A10	Bug Reporting Template	Communication	0.0588	RC-6, RC-11
12	A11	Cross-functional Meetings	Communication	0.0582	RC-6

The AHP result shows that the three highest-priority solutions are Automated Testing Tools, Test-Driven Development, and CI/CD Implementation. These solutions directly address the most critical technical and process-related root causes, particularly hardcoded query assumptions, missing regression tests, algorithm failures, unstructured triage, and distributed logging issues.

Corrective and Preventive Action Recommendations

Based on the AHP ranking, corrective actions were formulated for the three highest-priority solutions. Preventive actions were also proposed through improvements to the Testing Life Cycle SOP.

Table 8. Corrective Action Based on Top AHP Priorities

Priority	Solution	Key Implementation	Target Metric
1	Automated Testing Framework	Unit tests for balance aggregation, integration tests for journal automation, multi-branch regression suite, and post-deployment smoke tests	DVS recurrence reduced by more than 80%; regression detected within 15 minutes
2	Test-Driven Development	QA involvement in sprint planning, financial edge-case catalogue, and failing tests before implementation	Algorithm failures reduced by more than 50%; edge-case coverage above 90%
3	CI/CD Implementation	Automated test gate, centralized logging, and deployment health check	Zero regression to production; MTTR reduced by more than 50%

These corrective actions indicate that DJOIN should prioritize system-level and process-level improvements instead of relying only on manual bug fixing. The focus should be placed on automated validation, earlier QA involvement, and stronger deployment control.

Table 9. Recommended Improvements to the Testing Life Cycle SOP

STLC Phase	Identified Gap	SOP Recommendation	KPI
Requirement Review	QA was not involved, causing algorithm failures	QA sign-off before sprint and acceptance criteria definition	QA coverage above 95%
Test Case Design	No multi-branch and multi-product scenarios	Add mandatory multi-branch, multi-product, new product onboarding, and financial edge-case scenarios	Coverage per category above 90%
Test Execution	Many uncategorized tickets	Automate post-deployment smoke tests and require complete OpenProject metadata	Metadata completeness above 95%
Bug Triage	Many tickets remained "New" without progress	Maximum one working day for triage and two-hour escalation for severe bugs	SLA compliance above 90%
Regression Testing	Hotfix was performed without regression testing	Every hotfix must include at least one regression test case before production merge	Zero hotfix without regression
Post-Release Monitoring	Bugs were detected reactively through client reports	Automated DVS health check every six hours	Proactive detection above 70%

The SOP recommendations are designed to prevent similar failures from recurring by strengthening QA involvement, test coverage, bug triage discipline, regression testing, and post-release monitoring.

Discussion

The findings show that functional testing failures in DJOIN's system were caused by multiple interacting root causes rather than a single isolated factor. The dominance of Machine and Method categories indicates that the failures were mainly technical-systemic and process-related. Therefore, solving the problem requires improvements in system architecture, regression testing, testing workflow, and monitoring mechanisms.

The most critical root cause was the hardcoded product type assumption in the Balance Validation Dashboard query. This architectural weakness caused the dashboard to fail when new savings products or branch structures were introduced. The KSB case, where one DVS defect generated 38 tickets across nine branches, demonstrates that DJOIN's multi-tenant and multi-branch environment amplified the impact of a single defect. This finding confirms that functional testing in digital banking systems must include multi-product and multi-branch scenarios.

The AHP results support this interpretation. Solution Effectiveness received the highest weight, indicating that DJOIN needs solutions that directly eliminate root causes rather than temporary fixes. Automated Testing Tools ranked first because they directly address the three major root causes: hardcoded DVS logic, absence of regression testing, and algorithm failure. This result shows that technical automation is more strategic than purely managerial solutions when failures are recurring and system-wide.

The integration of RCA and AHP provides a stronger decision-making framework. RCA alone can identify root causes but does not provide implementation priorities. AHP alone can rank solutions but may lack empirical grounding if not supported by root cause evidence. By integrating both methods, this study ensures that each solution priority is based on actual failure evidence, severity assessment, business impact, and technical feasibility.

Overall, the findings indicate that DJOIN should shift from reactive bug fixing to preventive quality assurance. The recommended direction is to implement automated regression testing, adopt Test-Driven Development for financial logic, strengthen CI/CD quality gates, and improve the Testing Life Cycle SOP. These improvements are expected to reduce recurring functional failures, improve system reliability, and support a more sustainable QA process in digital banking.

4. Conclusions

This study identified the principal causes of recurring functional testing failures in PT DJOIN's Coopmax and KOCEK systems and prioritized the proposed improvement solutions using an integrated Root Cause Analysis (RCA) and Analytical Hierarchy Process (AHP) approach. Analysis of 1,759 production bug tickets recorded from 2023 to 2025 identified 12 root causes classified into the Machine, Method, Mother Nature, Material, Man, and Measurement categories. The three root causes with the highest severity scores were the hardcoded product-type assumption in the Balance Validation Dashboard query, with a score of 4.67; the absence of a regression test suite after hotfix implementation, with a score of 4.33; and distributed error logs across multiple services, with a score of 4.33. These findings indicate that the observed failures were predominantly associated with system architecture, testing procedures, regression validation, logging, and monitoring practices.

The AHP results assigned the highest weight to solution effectiveness at 42.44%, followed by business impact at 30.65%, implementation cost at 13.07%, implementation time at 8.36%, and technical complexity at 5.48%. The consistency ratio of 0.024 was below the acceptable threshold of 0.10, indicating that the pairwise-comparison judgments were sufficiently consistent. Among the 12 evaluated alternatives, Automated Testing Tools achieved the highest priority score of 0.1872, followed by Test-Driven Development at 0.1412 and CI/CD Implementation at 0.1372. These alternatives were prioritized because of their alignment with the major technical and procedural root causes identified through RCA.

Based on this prioritization, PT DJOIN may consider automated regression testing as the initial proposed action, followed by earlier QA involvement in development activities, CI/CD quality gates, centralized logging, complete bug-ticket metadata, and a structured bug-triage service-level agreement. Refactoring the Balance Validation Dashboard toward a product-agnostic aggregation mechanism is also proposed to address dependency on manually defined product types. However, this study did not implement or evaluate the proposed solutions. Therefore, no conclusions can yet be drawn regarding their actual effectiveness in reducing recurring bugs or improving system reliability. Future research should conduct a post-implementation evaluation using indicators such as Bug Recurrence Rate, regression-test coverage, mean time to resolution, defect leakage, and triage SLA compliance. Further studies may also extend the analysis to a larger KOCEK dataset and compare AHP with methods such as Fuzzy AHP or TOPSIS.

References

[1] D. G. H. Divayana, A. Adiarta, P. W. A. Suyasa, I. M. Sugiarta, and G. E. B. Darmawan, "Use of the TTA-DIVAYANA Concept to Improve the Weight of Decision Makers in the TOPSIS Method as an Effort to Optimize Evaluation Results," in *Proceedings of the 8th International Conference on Sustainable Information Engineering and Technology*, New York, NY, USA: ACM, Oct. 2023, pp. 76–82. doi: 10.1145/3626641.3627506.

[2] M. Ariansidi, I. M. Candiasa, I. Made, and G. Sunarya, "Analisis Usability Pada Sistem Informasi LAPORBUP Menggunakan Performance Measurement, Retrospective Think Aloud dan User Experience Questionnaire," *KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 3, no. 6, pp. 754–764, 2023, doi: 10.30865/klik.v3i6.807.

[3] I. G. M. S. Dwipayana, I. M. Candiasa, and L. J. E. Dewi, "Usability Evaluation of Lecturer Information System in ITB STIKOM Bali," *sinkron*, vol. 9, no. 1, pp. 149–159, Jan. 2025, doi: 10.33395/sinkron.v9i1.14315.

[4] I. G. P. A. A. Putra, I. M. Candiasa, and I. N. Sukajaya, "Usability Evaluation of the Undiksha Letter Management System Website Using Performance Measurement, Think Aloud, And Mouse Tracking Methods," *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, vol. 14, no. 3, pp. 616–624, Dec. 2025, doi: 10.23887/janapati.v14i3.100659.

[5] I. G. P. Yada Giri, L. J. E. Dewi, and I. M. G. Sunarya, "The Evaluation of Usability and Website Development using Cognitive Walkthrough, Performance Measurement, and System Usability Scale," *Journal of Computer Networks, Architecture and High Performance Computing*, vol. 5, no. 2, pp. 503–514, Jul. 2023, doi: 10.47709/cnahpc.v5i2.2511.

[6] I. N. G. A. Y. Putra, I. M. Candiasa, and L. J. E. Dewi, "Usability Evaluation of SIDUMAS Badung Using Think Aloud, Heuristic Evaluation and SUS," *Sinkron*, vol. 8, no. 1, pp. 368–379, Jan. 2023, doi: 10.33395/sinkron.v8i1.12034.

[7] I. K. A. Diatmika Indra, I. M. G. Sunarya, and I. G. A. Gunadi, "EVALUASI LAYANAN PENGADUAN MASYARAKAT PRO DENPASAR SEBAGAI PLATFORM E-GOVERNMENT MENGGUNAKAN WEBQUAL 4.0, E-GOVQUAL DAN IMPORTANCE PERFORMANCE ANALYSIS," *Jurnal Pendidikan Teknologi dan Kejuruan*, vol. 22, no. 2, 2025, [Online]. Available: https://pengaduan.denpasarkota.go.id/.

[8] I. G. A. S. Nugraha, I. G. A. Gunadi, and L. J. E. Dewi, "Application of Bagging Ensemble Learning on Naïve Bayes Algorithm to Predict Coronary Heart Disease," *Jurnal Teknologi Informasi dan Pendidikan*, vol. 18, no. 2, pp. 943–954, Aug. 2025, doi: 10.24036/jtip.v18i2.981.

[9] N. K. Y. Suartini, I. M. A. Wirawan, and D. G. H. Divayana, "DSS for 'E-Private' Using a Combination of AHP and SAW Methods," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 13, no. 3, p. 251, Jul. 2019, doi: 10.22146/ijccs.46625.

[10] D. G. H. Divayana, "Simulation of the weighted product method integrated with the CSE-UCLA evaluation model," 2024, p. 030004. doi: 10.1063/5.0214095.

[11] "Software Testing and Quality Assurance," in *International Conference on Recent Trends in Computer Science and Information Technology*, ScienceTech Xplore, Oct. 2025. doi: 10.63282/3050-9246/ICRTCSIT-119.

[12] "AI in Software Quality Assurance: A Review of Machine Learning for Testing, Bug Prediction, and Debugging," *International Research Journal of Modernization in Engineering Technology & Science*, Aug. 2025, doi: 10.56726/IRJMETS76291.

[13] T. Ahmad Al-Rawashdeh, F. Al'Azzeh, and F. Al-Tarawneh, "An Analytical Hierarchy Process-Based Technique for Software Requirements Prioritization," *IEEE Access*, vol. 13, pp. 50603–50610, 2025, doi: 10.1109/ACCESS.2025.3552490.

[14] A. M. S. R. H. Attanayake, R. M. Chandima Ratnayake, and T. Markeset, "Functional Failure Criticality Assessment and Control of Power Distribution Systems: Failure Prioritization Using AHP," *IEEE Access*, vol. 12, pp. 72968–72982, 2024, doi: 10.1109/ACCESS.2024.3403030.

[15] S. Feng *et al.*, "ROCAS: Root Cause Analysis of Autonomous Driving Accidents via Cyber-Physical Co-mutation," in *Proceedings - 2024 39th ACM/IEEE International Conference on Automated Software Engineering, ASE 2024*, Association for Computing Machinery, Inc, Oct. 2024, pp. 1620–1632. doi: 10.1145/3691620.3695530.

[16] S. Sarwosri, S. Rochimah, U. Laili Yuhana, and S. Balqis Hidayat, "Software Quality Measurement for Functional Suitability, Performance Efficiency, and Reliability Characteristics

Using Analytical Hierarchy Process," *JOIV : International Journal on Informatics Visualization*, vol. 7, no. 4, p. 2421, Dec. 2023, doi: 10.62527/joiv.7.4.2441.

[17] L. Giamattei, A. Guerriero, R. Pietrantuono, and S. Russo, "Causal reasoning in Software Quality Assurance: A systematic review," *Inf. Softw. Technol.*, vol. 178, p. 107599, Feb. 2025, doi: 10.1016/j.infsof.2024.107599.

[18] S. Zhang *et al.*, "Failure Diagnosis in Microservice Systems: A Comprehensive Survey and Analysis," *ACM Transactions on Software Engineering and Methodology*, vol. 35, no. 1, pp. 1–55, Jan. 2026, doi: 10.1145/3715005.

[19] W. Yue, J. Chai, X. Wan, Y. Xie, X. Chen, and W. Gui, "Root cause analysis for process industry using causal knowledge map under large group environment," *Advanced Engineering Informatics*, vol. 57, p. 102057, Aug. 2023, doi: 10.1016/j.aei.2023.102057.

[20] D. Pietsch, M. Matthes, U. Wieland, S. Ihlenfeldt, and T. Munkelt, "Root Cause Analysis in Industrial Manufacturing: A Scoping Review of Current Research, Challenges and the Promises of AI-Driven Approaches," *Journal of Manufacturing and Materials Processing*, vol. 8, no. 6, p. 277, Dec. 2024, doi: 10.3390/jmmp8060277.

[21] I. Nengah, A. A. Dwijayadi, I. Made, A. Wirawan, D. Gede, and H. Divayana, "PENGEMBANGAN SISTEM PENDUKUNG KEPUTUSAN PENENTUAN HOTEL DI KECAMATAN BULELENG DENGAN METODE ANALYTIC HIERARCHY PROCESS (AHP) DAN TECHNIQUE FOR OTHERS REFERENCE BY SIMILARITY TO IDEAL SOLUTION (TOPSIS)," 2018.

[22] R.-B. Ngoie *et al.*, "A HYBRID APPROACH COMBINING CONJOINT ANALYSIS AND THE ANALYTIC HIERARCHY PROCESS FOR MULTICRITERIA GROUP DECISION-MAKING," *International Journal of the Analytic Hierarchy Process*, vol. 17, no. 1, Jun. 2025, doi: 10.13033/ijahp.v17i1.1308.

[23] "Multi-Criteria Decision Making with Analytical Hierarchy Process for Headphone Recommender System," *Journal of System and Management Sciences*, vol. 14, no. 3, Feb. 2024, doi: 10.33168/JSMS.2024.0311.

[24] A. Perera, B. Turhan, A. Aleti, and M. Böhme, "On the Impact of Lower Recall and Precision in Defect Prediction for Guiding Search-based Software Testing," *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 6, pp. 1–27, Jul. 2024, doi: 10.1145/3655022.

[25] L. Pham, H. Ha, and H. Zhang, "Root Cause Analysis for Microservice System based on Causal Inference: How Far Are We?," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, New York, NY, USA: ACM, Oct. 2024, pp. 706–715. doi: 10.1145/3691620.3695065.