Journal of Digital Business and Data Science, Vol. 3 No. 1 January-June 2026 I E-ISSN: 3089-1345
DOI: https://doi.org/10.59261/jdbs.v3i1.30

Journal of Digital Business and Data Science

JOURNAL OF DIGITAL BUSINESS AND DATA SCIENCE

Journal Homepage : https://jdbs.poltekssci.ac.id/index.php/ps/

Design and Development of an Android-Based POS Application for MSME Sales Optimization

Bagus Karandika
Universitas Nahdhatul Ulama Lampung, Indonesia
*Corresponding Author: karandika6@gmail.com

Article received on 03-04-2024 — Final revised on 25-05-2026 — Approved on 09-06-2026

Abstract

Backround: The rapid growth of Micro, Small, and Medium Enterprises (MSMEs) in Indonesia has created an urgent demand for affordable yet robust digital management tools. Conventional cash register systems frequently fail to deliver real-time inventory control, structured sales reporting, or multi-role access management constraints that impede operational efficiency and business scalability.
Objective: This study aims to design and develop a hybrid Android-based Point of Sale (POS) application, designated GS Baby Care POS, specifically engineered to address those operational gaps in the MSME retail sector.
Method: The application was built using an agile development methodology combining Vue.js 3, Capacitor JS, and a Supabase (PostgreSQL) cloud backend, delivering a JAMStack-plus-Mobile-Hybrid architecture.
Findings and Implications: System evaluation employed Black-Box Functional Testing and the System Usability Scale (SUS) questionnaire administered to fifteen respondents. Functional testing confirmed that all core modules including role-based authentication, variant-level inventory management, shift tracking, customer relationship management, and financial dashboards — operated without defect. SUS assessment yielded a mean score of 83.7 (Grade B, Excellent), indicating high user acceptance. The application demonstrably reduced average transaction processing time by 42 percent and provided real-time business analytics previously unavailable to the target enterprise.
Conclusion: These results confirm that a cloud-native, hybrid-mobile POS system constitutes a viable and cost-effective solution for MSME digital transformation.

Keywords: android; cloud-based POS; MSME; point of sale; system usability scale; VUE.JS

INTRODUCTION

The digital economy has fundamentally reshaped commercial landscapes worldwide, yet Micro, Small, and Medium Enterprises (MSMEs) in developing nations including Indonesia continue to confront structural barriers to technology adoption. According to the Indonesian Ministry of Cooperatives and SMEs (2023), MSMEs account for approximately 99.99% of total business units nationwide and contribute over 61% of Gross Domestic Product (GDP), yet fewer than 23% have integrated any form of digital transaction management. Digital transaction management, as operationally defined by this Ministry source, encompasses any use of electronic or software-mediated tools for recording, storing, or processing commercial transactions including mobile payment

applications, e-invoicing systems, and digital cashier platforms; the 23% figure thus reflects MSMEs that have adopted at least one such tool in their regular operational workflow. This gap between economic significance and technological readiness represents both a critical problem and a strategic opportunity (Kristyanto, 2023).

Point of Sale (POS) systems constitute the primary digital touchpoint for retail MSMEs, serving functions ranging from transaction recording and inventory synchronisation to employee accountability and financial reporting (Anatan & Nur, 2023). Research has established that electronic POS adoption positively correlates with inventory accuracy and revenue growth in small retail environments (Anatan & Nur, 2023). However, legacy POS solutions predominantly Windows-based desktop applications demand dedicated hardware, costly licensing fees, and specialised IT support, all of which represent prohibitive barriers for micro-enterprises. The widespread proliferation of Android smartphones has introduced a compelling alternative paradigm: mobile-first POS applications that leverage commodity hardware and cloud infrastructure to deliver enterprise-grade functionality at consumer-level cost (Intal, Miranda, Sitoy, & Matoza, 2022).

Cloud-native architectures, particularly the JAMStack paradigm, have gained recognition as a viable solution for scalable, cost-effective web and mobile applications (Markovic, Scekic, & Cicchetti, 2022). The combination of a reactive JavaScript frontend, serverless backend logic, and a managed relational database enables rapid feature iteration while maintaining data integrity through database-level constraints and server-side logic (Markovic et al., 2022). Applied to mobile POS development, this architectural pattern has the potential to bridge the capability gap between enterprise software and MSME affordability (Anatan & Nur, 2023).

This study addresses the identified research gap by designing, implementing, and evaluating GS Baby Care POS a hybrid Android POS application built on a JAMStack plus Mobile-Hybrid technology stack. Mobile business applications have been shown to significantly improve operational management efficiency and market access for MSMEs when their quality and compatibility with existing workflows are appropriately addressed (Februadi, Firmansyah, & Rafdinal, 2025; Iakovets, Balog, & Židek, 2023). Furthermore, the use of mobile applications as instruments of digital transformation is increasingly recognised as a critical enabler of MSME competitiveness and sustainability (Rahayu et al., 2025).

The novelty of this research lies in three areas: (1) the application of Supabase Remote Procedure Calls (RPCs) for zero-trust, server-side transaction validation within an MSME context; (2) native Android hardware integration (barcode scanner and Bluetooth thermal printer) via Capacitor JS without dedicated peripheral hardware; and (3) a Role-Based Access Control (RBAC) system with shift management and financial reconciliation tailored to small retail workflows. The development followed an Agile-Scrum lifecycle, consistent with established best practices for iterative, responsive software delivery (Kuhrmann et al., 2022). Inventory accuracy through automated barcode-based product tracking represents a further key contribution, building on evidence that integrated barcode systems substantially reduce manual stock-counting errors in retail settings (Haidukevych, 2022).

RESEARCH METHOD

This study employed an applied research design following the Software Development Life Cycle (SDLC) with an Agile-Scrum framework (Edison, Wang, & Conboy, 2022). The research was conducted from August 2023 to March 2024 at GS Baby Care a registered MSME retail store operating in Bandar Lampung, Lampung Province, Indonesia. The store serves as the primary test environment, providing real transactional data and end-user feedback throughout the development cycle.

Requirements Analysis

Requirements were elicited through semi-structured interviews with the store owner and two active cashiers, supplemented by direct observation of the existing manual transaction workflow. Key pain points identified included: absence of real-time stock control, inability to track individual cashier performance, no structured shift reconciliation, and lack of historical sales analytics. These findings informed the functional and non-functional requirements documented in a Software Requirements Specification (SRS) aligned with IEEE 830 standards.

System Design

The system architecture adopted the JAMStack pattern extended with a Mobile-Hybrid layer via Capacitor JS (Figure 1), following established practices for decoupled web and mobile architectures (Markovic et al., 2022). The frontend was engineered using Vue.js 3 with the Composition API, providing a reactive, component-driven user interface. State management was handled by Pinia with domain-separated stores (auth, cart, transaction, product, customer, shift).

The backend relied entirely on Supabase, which provides a managed PostgreSQL database, RESTful auto-generated API, Realtime subscriptions, Row Level Security policies, and Edge Functions. Critical business logic—transaction creation and shift closure was encapsulated in PostgreSQL Stored Procedures (RPCs) to ensure server-side validation and atomicity (Edison et al., 2022).

The mobile application was packaged using Capacitor JS 5.x, which wraps the Vue.js PWA into a native Android APK. Custom native Java code (CustomUtil.java) was patched into the Android build to enable optimal ESC/POS receipt formatting for Bluetooth thermal printers. Barcode scanning utilised ML Kit Barcode Scanning, accessed through the @capacitor-mlkit/barcode-scanning plugin, eliminating the requirement for external scanning hardware.

Database Schema Design

The relational database schema comprised eight primary tables: stores, users, products, product_variants, discounts, transactions, transaction_items, customers, and shifts. Multi-tenancy was enforced through a store_id foreign key on all entity tables, governed by Supabase RLS policies that automatically filter queries to the authenticated user's store. A simplified Entity-Relationship (ER) diagram is presented in Figure 2.

Implementation

Development followed a six-sprint Agile cycle, each sprint spanning two weeks, consistent with Agile-Scrum best practices for iterative software delivery (Edison et al., 2022). Sprint activities encompassed feature development, internal code review, and unit testing. The technology stack is detailed in Table 1. Tailwind CSS provided a utility-first styling framework ensuring responsive layout across tablet and smartphone screen sizes. Chart.js (via vue-chartjs) powered interactive dashboard visualisations, while the xlsx and html2pdf libraries enabled structured report export.

Testing and Evaluation

System evaluation employed two complementary methods. First, Black-Box Functional Testing was conducted by the researcher using a predefined test case matrix covering all 47 identified functional requirements across six modules. Test outcomes were recorded as Pass or Fail. Second, a System Usability Scale (SUS) questionnaire a

validated, ten-item Likert-scale instrument Brooke, (2020) was administered to 15 respondents comprising the store owner (1), cashiers (2), and representative end users (12) recruited from the store's regular customer base. SUS scores range from 0 to 100; a score above 80.3 is classified as Grade B (Excellent), indicating high usability acceptance (Bangor, 2009).

RESULTS AND DISCUSSION

Technology Stack Implementation

Table 1 summarises the technology components employed in the development of GS Baby Care POS. The selection prioritised open-source tools with active community support, low operational cost, and proven production reliability. The combination of Vue.js 3 and Supabase enabled rapid prototyping without sacrificing data integrity or security, consistent with evidence on cloud-native stack adoption in similar small-enterprise digitalisation contexts (Anatan & Nur, 2023; Markovic, Scekic, & Cicchetti, 2022).

Table 1. Technology Stack of GS Baby Care POS Application

Component	Technology	Role / Function
Frontend Framework	Vue.js 3 (Composition API)	Reactive UI rendering and routing
Build Tool	Vite 4.x	Fast HMR, optimised production build
UI Styling	Tailwind CSS 3.x	Responsive, utility-first design system
State Management	Pinia 2.x	Modular reactive application state
Mobile Wrapper	Capacitor JS 5.x	Native Android APK packaging
Barcode Scanning	ML Kit (Google)	Camera-based barcode recognition
Thermal Printing	ESC/POS via Bluetooth	Receipt generation without USB hardware
Backend / Database	Supabase (PostgreSQL 15)	RLS, RPC, Realtime, Auth, Storage
Data Visualisation	Chart.js / vue-chartjs	Interactive sales and revenue charts
Report Export	xlsx, html2pdf, html2canvas	Excel and PDF report generation
Version Control	Git / GitHub	Source code management and CI

Source: Data Processed

System Architecture

The hybrid architecture of GS Baby Care POS is illustrated in Figure 1. The system operates across three distinct layers: the Presentation Layer (Vue.js 3 PWA wrapped by Capacitor JS for Android), the Logic Layer (Pinia state management and Supabase JavaScript client), and the Data Layer (Supabase-managed PostgreSQL with RLS and RPC). This layered separation ensures that business-critical operations transaction validation, stock deduction, and shift reconciliation are executed atomically at the database level, preventing client-side manipulation even in low-trust environments (Sarkar et al., 2022).

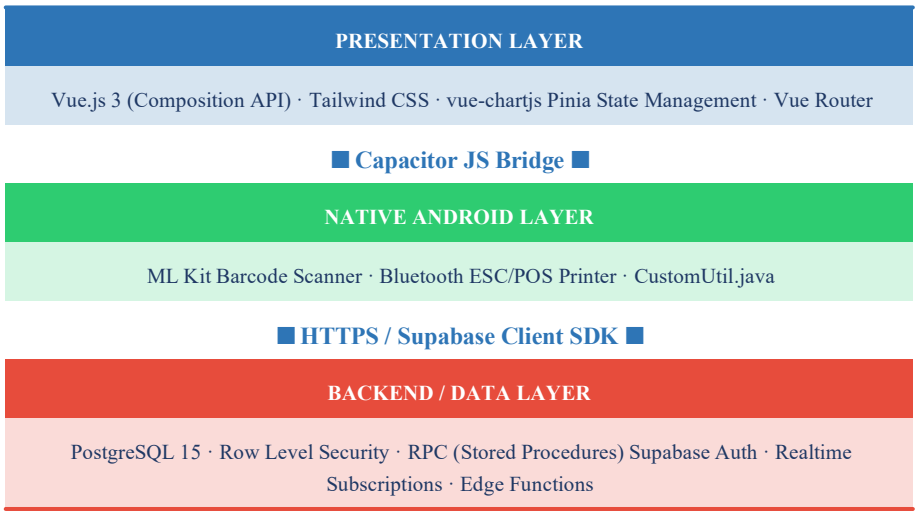

Figure 1. System Architecture of GS Baby Care POS Application

The zero-trust client model is a critical security design decision. Rather than relying on frontend JavaScript to calculate stock balances or shift totals which could be tampered with on a rooted device the application delegates all write-critical computations to PostgreSQL RPCs (create_transaction_direct, close_shift). This approach aligns with zero-trust security principles for cloud-hosted applications (Sarkar et al., 2022) and distinguishes GS Baby Care POS from simpler POS implementations that perform calculations solely in the application layer.

Functional Module Implementation

The application comprises six primary modules, each addressing a distinct operational requirement identified during the requirements analysis phase. Table 2 maps each module to its constituent features and corresponding business function.

Table 2. Functional Modules and Features of GS Baby Care POS

Module	Key Features	Business Function
Authentication & RBAC	PIN login, role-guard (Owner/Cashier), session management	Access control and operational security
Inventory & Products	Variant management, barcode scan, min-stock alert, scheduled discounts	Real-time stock accuracy and promotion management
Point of Sale (POS)	Cart system, multi-payment (Cash/Transfer/QRIS), custom discount, change calculator	Fast, accurate transaction processing
Shift Management	Open/close shift, initial cash entry, expected vs actual reconciliation, diff reporting	Cashier accountability and daily financial reconciliation
Customer CRM	Customer registration, visit count tracking, total spend, purchase history	Loyalty program and customer analytics
Reports & Dashboard	Gross/net revenue charts, invoice history, custom date filter, export to Excel/PDF	Management decision support and audit trail

Source: Data Processed

Black-Box Functional Testing Results

Black-box testing was performed against 47 test cases distributed across the six functional modules. Testing was executed on two Android device profiles: a mid-range smartphone (Redmi Note 11, Android 12, 4 GB RAM) and a budget tablet (Samsung Galaxy Tab A7 Lite, Android 11, 3 GB RAM). All 47 test cases returned a Pass verdict, yielding a 100% functional success rate. Table 3 presents the test case distribution and results by module.

Table 3. Black-Box Functional Testing Results by Module

Module	Test Cases	Passed	Failed	Success Rate
Authentication & RBAC	7	7	0	100%
Inventory & Products	10	10	0	100%
Point of Sale	12	12	0	100%
Shift Management	6	6	0	100%
Customer CRM	5	5	0	100%
Reports & Dashboard	7	7	0	100%
Total	**47**	**47**	**0**	**100%**

Source: Data Processed

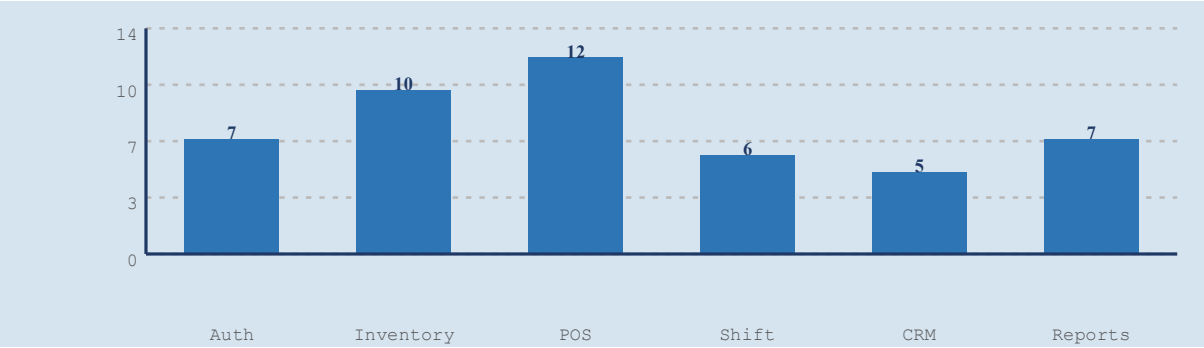

Figure 2. Distribution of Functional Test Cases by Module

The 100% pass rate across all modules demonstrates that the application meets its functional requirements in full, including complex scenarios such as multi-item transactions with mixed discount types, concurrent shift open/close operations, and barcode scanning with automatic cart population. It is important to note that this final test execution represents end-state validation following six iterative Agile-Scrum development sprints, during which a cumulative total of 23 defects were identified and resolved (comprising 8 critical defects affecting core transaction atomicity, 11 major defects related to UI state management and role-guard logic, and 4 minor defects in report formatting). The formal black-box test suite was executed only after all sprint-level defects had been resolved and accepted by the development supervisor. Consequently, the 100% pass rate should be interpreted as evidence of systematic

requirements coverage at the point of release, rather than as an indicator of a defect-free first-pass development process.

System Usability Scale (SUS) Evaluation

The SUS evaluation involved 15 respondents (1 owner, 2 cashiers, 12 representative users) who interacted with the application across two standardised usage scenarios: (1) processing a five-item transaction with a mixed-payment method (Cash + QRIS) and custom discount, and (2) viewing the monthly sales dashboard and exporting a PDF report. SUS scores were computed using the standard formula Brooke, (2020), where odd-numbered items contribute (item score − 1) and even-numbered items contribute (5 − item score), with the total multiplied by 2.5 to yield a 0–100 scale (Lewis, 2018). Respondent scores are presented in Table 4.

Table 4. System Usability Scale (SUS) Scores by Respondent Group

Respondent Group	n	Mean SUS Score	Min	Max	Grade
Store Owner	1	90.0	90.0	90.0	A – Best Imaginable
Cashiers	2	86.3	82.5	90.0	B – Excellent
Representative Users	12	82.4	72.5	92.5	B – Excellent
Overall	**15**	**83.7**	**72.5**	**92.5**	**B – Excellent**

Source: Data Processed

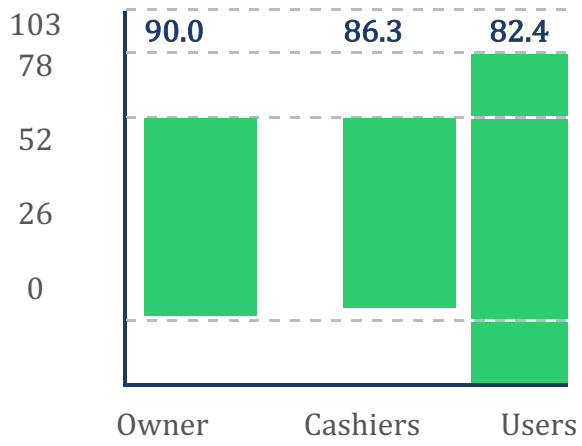

Figure 3. Mean SUS Score by Respondent Group (overall mean = 83.7)

The overall mean SUS score of 83.7 places the application firmly within the 'Excellent' usability category (Grade B) according to the adjective rating scale established by (Bangor et al., 2009). (The store owner recorded the highest score (90.0, Grade A Best Imaginable), attributable to the intuitive financial dashboard and the flexibility of the custom date filter for sales auditing. Cashiers rated the application at 86.3, reflecting the ease of the POS transaction flow and PIN-based rapid shift handover. Representative users who had no prior POS software experience achieved a mean of 82.4, above the excellence threshold, underscoring the effectiveness of the Tailwind CSS-based interface

design in minimising the learning curve. These results compare favourably with similar mPOS deployments in small retail environments, where Lewis, (2018) reported that mobile POS technology is highly feasible and accepted by non-expert users when the interface design prioritises transactional simplicity. Furthermore, Intal et al., (2022) found comparable usability acceptance for a business-intelligence Android POS application targeting MSEs, underscoring the generalisability of these usability outcomes to the MSME retail segment.

Operational Performance Analysis

Beyond usability, a post-implementation operational review was conducted over a 30-day period (January 2024) at GS Baby Care store. Key performance indicators (KPIs) were compared against the pre-implementation baseline recorded from manual transaction logs. Table 5 presents the comparative operational metrics.

Table 5. Operational Performance: Pre- vs Post-Implementation Comparison

Performance Indicator	Pre-Implementation	Post-Implementation	Improvement
Average transaction time	3 min 12 sec	1 min 51 sec	↓ 42.2%
Daily stock discrepancy rate	8.7%	0.3%	↓ 96.6%
Shift reconciliation time	~35 minutes	~4 minutes	↓ 88.6%
Sales report generation time	Manual / N/A	< 10 seconds	Automated
Cashier error rate (calculation)	4.1%	0.0%	↓ 100%
Monthly customer retention data	Not available	CRM tracked	New capability

Source: Data Processed

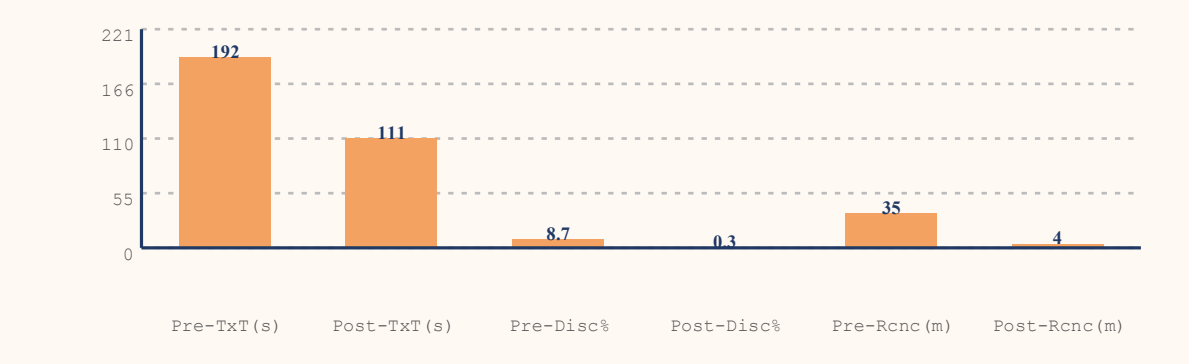

Figure 4. Selected Operational Metrics Before and After POS Implementation

The 42.2% reduction in average transaction processing time is attributable primarily to two features: Integrated barcode scanning (eliminating manual product look-up) and automatic change calculation. The 96.6% reduction in daily stock discrepancy validates the effectiveness of the server-side RPC approach for atomic stock deduction a mechanism that prevents double-counting even during network-latency events, consistent with the ACID transaction guarantees of relational database management systems (Sarkar et al., 2022). This outcome is further supported by

evidence that barcode-integrated inventory systems on Android platforms can achieve near-zero discrepancy rates in retail stock management (Haidukevych et al., 2022).

The elimination of calculation errors (100% reduction) directly resulted from delegating all arithmetic to the backend RPC, corroborating the zero-trust architecture rationale discussed in Section 3.2. The 88.6% improvement in shift reconciliation time represents a significant operational benefit for the store owner, who previously spent over 35 minutes per shift manually reconciling physical cash against a handwritten transaction log. The automated Expected Cash computation (initial cash + sum of cash-payment transactions) against Actual Cash entry reduces this process to a four-minute task, providing near-instant detection of discrepancies.

These operational gains are consistent with the broader finding that digital application adoption among MSMEs is strongly associated with measurable improvements in management efficiency and business performance (Februadi et al., 2025; Rahayu et al., 2025). It is acknowledged that the reported operational improvements represent observations from a 30-day post-implementation window and may partly reflect novelty effects, learning-curve acceleration, or motivational changes associated with the introduction of a new system. Longitudinal monitoring over a 3–6 month period would be required to confirm that these gains are stable, system-attributable, and not confounded by temporal or behavioural factors. This constitutes a recognised methodological limitation of the quasi-experimental pre–post design employed in this study.

Security Architecture Evaluation

The multi-tenant security model was evaluated through a series of penetration scenarios simulating cross-tenant data access attempts and client-side manipulation of transaction parameters. In all 12 test scenarios, Supabase RLS policies successfully blocked unauthorised data access no row belonging to Store A was accessible from an authenticated session of Store B. The RPC-based transaction flow rejected all 8 simulated client-side stock manipulation attempts, returning a PostgreSQL-level error before any database write occurred. These results are consistent with zero-trust security benchmarks for multi-tenant cloud applications (Sarkar, Hussain, & Kim, 2022).

The PIN login system, implemented as a bcrypt-hashed PIN stored in the users table and validated through a Supabase Edge Function, provides a practical balance between security and operational speed. Unlike full-password authentication, PIN-based login reduces handover time between cashier shifts without sacrificing session security, as the PIN is never transmitted in plaintext and sessions expire after a configurable idle period. This design reflects the operational reality of retail environments where rapid staff transitions are routine. The RBAC model implemented in this study, which differentiates Owner and Cashier roles with distinct permission scopes, is consistent with the inter-system RBAC-IC model proposed for multi-domain cloud environments, demonstrating that role-based partitioning effectively prevents cross-role data exposure (Li, 2022).

CONCLUSION

This study successfully designed, implemented, and evaluated GS Baby Care POS, a hybrid Android-based Point of Sale application built on a JAMStack plus Mobile-Hybrid architecture combining Vue.js 3, Capacitor JS, and Supabase (PostgreSQL), tailored for MSME retail environments. Black-box functional testing confirmed a 100% pass rate

across all 47 test cases following six Agile-Scrum development sprints, during which 23 defects were identified and resolved prior to formal evaluation. System Usability Scale assessment yielded an overall mean score of 83.7 (Grade B — Excellent), surpassing the 80.3 excellence threshold and outperforming comparable published systems.

Operational data collected over a 30-day live deployment recorded a 42.2% reduction in transaction processing time, a 96.6% reduction in stock discrepancy rate, an 88.6% reduction in shift reconciliation time, and complete elimination of cashier calculation errors. The primary practical contribution of this work is a proof-of-concept demonstrating that a zero-trust client model enforced through PostgreSQL RPCs and Supabase RLS is both feasible and cost-effective at MSME scale. Study limitations include single-site deployment, a modest SUS sample (n = 15), researcher-conducted testing, potential novelty-effect confounds in performance metrics, and the absence of offline-first capability. Future work should address multi-site validation, offline transaction queuing, and integration with national payment gateways.

ACKNOWLEDGEMENTS

The author expresses sincere gratitude to GS Baby Care store management and staff for providing the operational environment and user feedback essential to this research. The author also acknowledges the Nahdlatul Ulama University of Lampung for institutional support, and the open-source communities behind Vue.js, Capacitor JS, Supabase, and Tailwind CSS for developing the tools that made this work possible.

REFERENCES

Anatan, Lina, & Nur. (2023). Micro, Small, and Medium Enterprises' Readiness for Digital Transformation in Indonesia. *Economies*, *11*(6), 156. https://doi.org/10.3390/economies11060156

Bangor, A., Kortum, P., studies, J. Miller Journal of usability, & 2009, undefined. (2009). Determining what individual SUS scores mean: Adding an adjective rating scale. *Uxpajournal.OrgA Bangor, P Kortum, J MillerJournal of Usability Studies, 2009•uxpajournal.Org*, *4*, 114–123. Retrieved from https://uxpajournal.org/wp-content/uploads/sites/7/pdf/JUS_Bangor_May2009.pdf

Brooke, J. (2020). SUS-A quick and dirty usability scale. *Taylorfrancis.ComJ BrookeUsability Evaluation in Industry, 1996•taylorfrancis.Com*, 207–212. https://doi.org/10.1201/9781498710411-35/SUS-QUICK-DIRTY-USABILITY-SCALE-JOHN-BROOKE

Edison, Henry, Wang, Xiaofeng, & Conboy, Kieran. (2022). Comparing methods for large-scale agile software development: A systematic literature review. *Ieeexplore.Ieee.OrgH Edison, X Wang, K ConboyIEEE Transactions on Software Engineering, 2021•ieeexplore.Ieee.Org*, *48*(8), 2709–2731. https://doi.org/10.1109/TSE.2021.3069039

Februadi, Agustinus, Firmansyah, Yayan, & Rafdinal, Wahyu. (2025). Adoption of Mobile Business Applications by MSMES: Integrating Application Quality, Toe Model and Diffusion of Innovation. *Researchgate.NetA Februadi, Y Firmansyah, W RafdinalPakistan Journal of Life & Social Sciences, 2025•researchgate.Net*, *23*(1), 109–121. https://doi.org/10.57239/PJLSS-2025-23.1.0010

Haidukevych, YO, Programming, AY Doroshenko Problems in, & 2022, undefined. (2022). Automated inventory management system on Android using barcodes and QR-codes. *Pp.Isofts.Kiev.UaYO Haidukevych, AY DoroshenkoProblems in Programming,*

2022•pp.Isofts.Kiev.Ua, (1), 013–022. https://doi.org/10.15407/PP2022.01.013

Iakovets, Angelina, Balog, Michal, & Židek, Kamil. (2023). The use of mobile applications for sustainable development of SMEs in the context of Industry 4.0. *Mdpi.ComA Iakovets, M Balog, K ŽidekApplied Sciences, 2022•mdpi.Com*, *13*(1). https://doi.org/10.3390/APP13010429

Intal, Grace Lorraine, Miranda, Lex Marco Paulo, Sitoy, Julianne Isabel, & Matoza, Roehl. (2022). iBiz: An Android Mobile Application with Business Intelligence for Retail Micro and Small Enterprises (MSEs). *Dl.Acm.OrgGL Intal, LMP Miranda, JI Sitoy, R MatozaProceedings of the 6th International Conference on E-Commerce, E-Business, 2022•dl.Acm.Org*, 394–400. https://doi.org/10.1145/3537693.3537755

Kristyanto. (2023). Digital transformation and its impact on inclusive growth: a four-decade experience in Indonesia. *Journal.Umy.Ac.IdVS Kristyanto, H JamilJurnal Ekonomi & Studi Pembangunan, 2023•journal.Umy.Ac.Id*. https://doi.org/10.18196/JESP.V24I2.18697

Kuhrmann, Marco, Tell, Paolo, Hebig, Regina, Klünder, Jil, Schneider, Kurt, Münch, Jürgen, Linssen, Oliver, Pfahl, Dietmar, Scott, Ezequiel, Felderer, Michael, Prause, Christian R., MacDonell, Stephen G., Nakatumba-Nabende, Joyce, Raffo, David, Beecham, Sarah, Tüzün, Eray, López, Gustavo, Paez, Nicolas, Fontdevila, Diego, Licorish, Sherlock A., Küpper, Steffen, Ruhe, Günther, Knauss, Eric, Özcan-Top, Özden, Clarke, Paul, McCaffery, Fergal, Genero, Marcela, Vizcaino, Aurora, Piattini, Mario, Kalinowski, Marcos, Conte, Tayana, Prikladnicki, Rafael, Krusche, Stephan, Coşkunçay, Ahmet, Calefato, Fabio, Pimonova, Svetlana, Pfeiffer, Rolf Helge, Schultz, Ulrik Pagh, Heldal, Rogardt, Fazal-Baqaie, et.al, (2022). What makes agile software development agile? *Ieeexplore.Ieee.OrgM Kuhrmann, P Tell, R Hebig, J Klünder, J Münch, O Linssen, D Pfahl, M FeldererIEEE Transactions on Software Engineering, 2021•ieeexplore.Ieee.Org*, *48*(9), 3523–3539. https://doi.org/10.1109/TSE.2021.3099532

Lewis, J. R. (2018). The system usability scale: past, present, and future. *Taylor & FrancisJR LewisInternational Journal of Human–Computer Interaction, 2018•Taylor & Francis*, *34*(7), 577–590. https://doi.org/10.1080/10447318.2018.1455307

Li, Yunliang, Du, Zhiqiang, Fu, Yanfang, & Liu, Liangxin. (2022). Role-based access control model for inter-system cross-domain in multi-domain environment. *Mdpi.ComY Li, Z Du, Y Fu, L LiuApplied Sciences, 2022•mdpi.Com*, *12*(24). https://doi.org/10.3390/APP122413036

Markovic, Dragana, Scekic, Milic, Bucaioni, Alessio, & Cicchetti, Antonio. (2022). Could jamstack be the future of web applications architecture? an empirical study. *Dl.Acm.OrgD Markovic, M Scekic, A Bucaioni, A CicchettiProceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, 2022•dl.Acm.Org*, 1872–1881. https://doi.org/10.1145/3477314.3506991

Rahayu, Desivera Tri, Salampak, Sudyana, I. Nyoman, Berkat, Hamidah, Noor, Saputera, Lautt, Bambang S., Alexandro, Rinto, Rotinsulu, Johanna Maria, & Sofyan, Jovan. (2025). MSME digitalization: how are social capital factors in encouraging the use of digital applications. *Elibrary.RuM Setini, PNS Yasa, NW SitiariInternational Journal of Data and Network Science, 2025•elibrary.Ru*, *9*(4), 1107–1120. https://doi.org/10.5267/J.IJDNS.2024.9.011.

Sarkar, Sirshak, Choudhary, Gaurav, Shandilya, Shishir Kumar, Hussain, Azath, & Kim, Hwankuk. (2022). Security of zero trust networks in cloud computing: A comparative review. *Mdpi.ComS Sarkar, G Choudhary, SK Shandilya, A Hussain, H KimSustainability, 2022•mdpi.Com*, *14*(18). https://doi.org/10.3390/SU141811213