J-MS
JOURNAL MOBILE TECHNOLOGIES

Received: 4 Aug 2025 | Revised: 13 Aug 2025 | Accepted: 1 Sep 2025 | Published: 10 Sep 2025 E-ISSN: 2986-478X, P-ISSN: 2986-1535

DOI: 10.59431/jms.v3i2.646

RESEARCH ARTICLE

Design and Development of a Mobile Application Using Android Studio and Flutter

Hanis Diansyah [1*] | Syafrinal [2]

[1*,2] STMIK Indonesia Banda Aceh, Banda Aceh City, Aceh Province, Indonesia.

Correspondence

[1*] STMIK Indonesia Banda Aceh, Banda Aceh City, Aceh Province, Indonesia.
Email: hanis.diasnyah@gmail.com.

Funding information
STMIK Indonesia Banda Aceh.

Abstract
This study provides a comparative analysis of Android Studio and Flutter frameworks in mobile application development, focusing on development efficiency, system performance, and user experience. Employing a mixed-method design, the research integrates experimental development, performance benchmarking, and usability evaluation using the System Usability Scale (SUS). Two prototype applications with identical functionalities were developed on each platform to ensure equivalent comparison. The results show that Flutter improved development efficiency by reducing development time by 27.8% and code volume by 32.7% compared to Android Studio. Its hot reload capability further accelerated iterative testing and interface refinement. However, Flutter applications consumed 18.1% more memory and 27.6% more CPU resources, though they achieved 28% faster frame rendering and smoother animations. Usability testing yielded excellent outcomes for both platforms, with Flutter achieving a slightly higher SUS score (82.4) than Android Studio (78.9). The Indonesian SUS adaptation demonstrated strong reliability (Cronbach's $\alpha = 0.87$). These findings indicate that Flutter is particularly effective for projects requiring rapid deployment and multi-platform compatibility, while Android Studio remains optimal for resource-constrained environments and applications requiring native optimization. Overall, framework selection should be determined by project-specific objectives balancing efficiency, performance, and usability.

Keywords
Flutter; Android Studio; Usability; Mobile Development; Performance.

1 | INTRODUCTION

Mobile application development has become a central focus in the digital technology ecosystem, with increasing attention toward optimizing design frameworks for efficiency, functionality, and user experience. Among the tools available, Android Studio and Flutter represent two dominant frameworks widely adopted by developers and researchers. Android Studio, as the official Integrated Development Environment (IDE) for Android, offers a structured environment for developing optimized native applications supported by a broad range of integrated tools (Nasution *et al*., 2019). The System Development Life Cycle (SDLC) remains a foundational approach in this context, ensuring that critical phases—from requirement analysis to design and testing—address user needs and technical feasibility (Wulan & Fauzi, 2021). In contrast, Flutter, introduced by Google, has rapidly gained prominence for its ability to streamline cross-platform

development using a single codebase written in the Dart programming language (Arb & Al-Majdi, 2020; Feran *et al*., 2023). Its widget-based architecture enables the creation of responsive and visually cohesive interfaces across multiple platforms, improving performance consistency and reducing design fragmentation (Suhendro *et al*., 2021; Septiani *et al*., 2024). Comparative research has emphasized that while Android Studio excels in platform-specific optimization, Flutter offers a faster development cycle and superior adaptability for multi-platform deployment (Feran *et al*., 2023; Fauzi *et al*., 2021). These characteristics position Flutter as a viable alternative for projects emphasizing efficiency and scalability in multi-device ecosystems.

The relevance of choosing the appropriate framework extends beyond functionality—it also affects project timelines, system performance, and long-term maintainability. Empirical studies have shown that structured methodologies such as prototype development and Blackbox Testing are essential for validating application performance and reliability prior to deployment (Bhahri, 2021; Septiani *et al*., 2024). Recent implementations of Flutter in sectors such as salon management systems (Septiani *et al*., 2024), e-commerce (Mustova & Pramudwiatmoko, 2025), and digital marketing (Nabawi *et al*., 2024) demonstrate its flexibility and adaptability to varied business contexts. Meanwhile, Android Studio continues to underpin specialized developments, including reservation systems (Setyaka *et al*., 2024), e-tourism (Bhahri, 2021), and geographic information systems for retail mapping (Sari, 2025), reaffirming its reliability for native performance optimization. The evolution of mobile technology has also expanded research attention toward user experience and usability evaluation. The System Usability Scale (SUS), validated through both international and Indonesian adaptations, provides an established framework for assessing perceived usability across different user demographics (Peres *et al*., 2013; Sharfina & Santoso, 2016; Kaya *et al*., 2019; Vlachogianni & Tselios, 2021). This tool enables consistent measurement of user satisfaction and system interaction efficiency—two critical factors influencing adoption and retention in mobile applications. Furthermore, studies examining monetization strategies (Wali, 2017), financial management systems (Ramadhan & Julkarnain, 2024), and data-driven service platforms (Fitriastuti *et al*., 2022) illustrate the increasingly interdisciplinary nature of mobile application design, where performance optimization, usability, and business objectives intersect. In summary, both Android Studio and Flutter offer distinct pathways to achieving efficient and functional mobile applications. The selection between the two should be guided by project-specific parameters such as performance expectations, device compatibility, development resources, and intended user experience (Andrean *et al*., 2020; Pudjiarti & Faizah, 2021). As the mobile ecosystem continues to mature, understanding the operational trade-offs between native and cross-platform development frameworks becomes fundamental for advancing research and practice in mobile computing.

2 | BACKGROUND THEORY

The design and development of mobile applications using Android Studio and Flutter represent an evolving area of software engineering that combines various methodologies, tools, and frameworks to achieve functional, efficient, and user-oriented outcomes. Android Studio, as the official Integrated Development Environment (IDE) for Android, provides developers with a comprehensive suite of tools for building complex and high-quality applications (Fitriastuti *et al*., 2022). In Android-based development, Firebase is frequently implemented as a real-time backend service, enabling seamless data storage and synchronization while allowing developers to concentrate on enhancing core functionalities rather than configuring backend infrastructures (Fitriastuti *et al*., 2022). This integration has strengthened Android Studio's role as a preferred platform for projects requiring native optimization and tight integration with Google's cloud services. In contrast, Flutter, developed by Google, has become a widely adopted framework for cross-platform development. Its single codebase approach allows applications to be deployed simultaneously on Android and iOS, significantly reducing development overhead. The use of ready-to-use widgets and strong community support enables faster interface construction and efficient debugging, resulting in responsive and visually cohesive user experiences (Kustiawati *et al*., 2025). As demonstrated by Kustiawati *et al*., the implementation of Flutter in mobile learning media achieved positive evaluations from both students and educators, indicating that the framework supports effective digital learning experiences across educational levels. Among the structured approaches in mobile development, the System Development Life Cycle (SDLC) remains one of the most widely adopted methodologies across both platforms.

The SDLC framework encompasses critical phases such as requirements analysis, system design, coding, testing, and deployment (Kurniawan *et al*., 2025). Kurniawan *et al*. applied this approach in developing an attendance tracking application using Flutter, emphasizing usability and efficient data management. Furthermore, the Waterfall model, as discussed by Tasik *et al*. (2024), continues to serve as a clear and sequential process that minimizes late-stage design errors by ensuring that each development phase is systematically completed before progressing to the next. Such structured methodologies provide predictability and quality assurance throughout the development process. Application evaluation and testing form an integral part of mobile application design. Blackbox testing, in particular, is widely utilized to assess system functionality and user satisfaction without requiring internal

code inspection. Studies have shown that both Android Studio– and Flutter-based applications consistently meet expected performance benchmarks when tested using this method, indicating their reliability and adherence to user requirements (Suhendro *et al*., 2021). User feedback gathered during testing further contributes to iterative improvement and refinement of interface design and usability. Overall, mobile application design using Android Studio and Flutter entails distinct advantages and challenges. Android Studio provides native optimization and deeper integration with Android-specific hardware and software environments, while Flutter offers flexibility, faster iteration cycles, and cross-platform consistency (Ramadhan & Julkarnain, 2024). In practice, hybrid adoption—leveraging the strengths of both platforms—can result in applications that are not only functionally robust but also aligned with contemporary user expectations in an increasingly diverse mobile ecosystem.

3 | METHOD

This research adopts a mixed-method design, integrating both qualitative and quantitative approaches to investigate the comparative dynamics of mobile application development using Android Studio and Flutter frameworks. The methodological structure was formulated to capture detailed evidence regarding development practices, performance behavior, and user experience characteristics across both platforms, ensuring analytical rigor and replicability. Data collection was carried out through three complementary stages: literature review, experimental development, and comparative evaluation. First, a systematic literature review analyzed 25 peer-reviewed studies published between 2017 and 2025 that addressed mobile application frameworks, emphasizing empirical investigations of Android Studio and Flutter. Studies were included based on clearly defined methodologies, measurable performance indicators, and user-focused evaluations. In the experimental phase, two functionally equivalent prototype applications were developed—one using Android Studio and the other using Flutter. Both prototypes implemented identical features, including user authentication, real-time data storage, and push notifications, to ensure a valid comparative basis. Throughout development, quantitative metrics were recorded, such as total development time, lines of code, and system resource utilization. Performance benchmarking was subsequently conducted using standardized testing tools to measure application launch time, memory consumption, CPU usage, and rendering performance under varied device configurations. The user experience assessment employed the System Usability Scale (SUS), a validated metric for evaluating perceived usability (Peres *et al*., 2013). This instrument was selected due to its reliability in assessing mobile and educational technologies, as confirmed by Vlachogianni and Tselios (2021). To accommodate cultural and linguistic nuances, the Indonesian adaptation of SUS developed by Sharfina and Santoso (2016) was used, preserving the psychometric validity of the original instrument. A total of 42 participants from diverse technical and professional backgrounds were recruited following the usability testing framework proposed by Kaya *et al*. (2019).

Each participant performed standardized tasks within both prototype applications, while key performance indicators such as task completion rate, time-on-task, and error frequency were systematically recorded. To complement quantitative findings, semi-structured interviews were conducted to capture subjective perceptions regarding usability, visual appeal, and responsiveness. Data analysis combined statistical and interpretive techniques. Quantitative data from performance tests and SUS scores were analyzed using paired t-tests to identify statistically significant differences between platforms. Following established SUS interpretation guidelines, scores exceeding 68 were classified as above average in usability (Peres *et al*., 2013). Qualitative responses from interviews were subjected to thematic analysis, allowing the identification of recurring themes in developer experience and user interaction feedback. The integration of multiple data sources provided methodological triangulation, enhancing the validity and interpretive depth of the findings. This approach acknowledges that variations in implementation may influence cross-platform outcomes beyond the intrinsic characteristics of the frameworks themselves (Wali, 2017; Sari, 2025). The study builds upon previous works by Mustova and Pramudwiatmoko (2025) and Nabawi *et al*. (2024) by introducing more controlled experimental conditions, standardized usability testing, and quantified performance comparisons. Additionally, this design addresses the methodological gaps identified in earlier research by equalizing developer expertise and ensuring consistent feature implementation across both environments.

4 | RESULTS AND DISCUSSION

4.1 Results

The comparative analysis between Android Studio and Flutter revealed measurable differences in development efficiency, performance behavior, and user experience. The key quantitative outcomes from the

experimental development stage are summarized in Table 1.

Table 1. Comparative Development Metrics

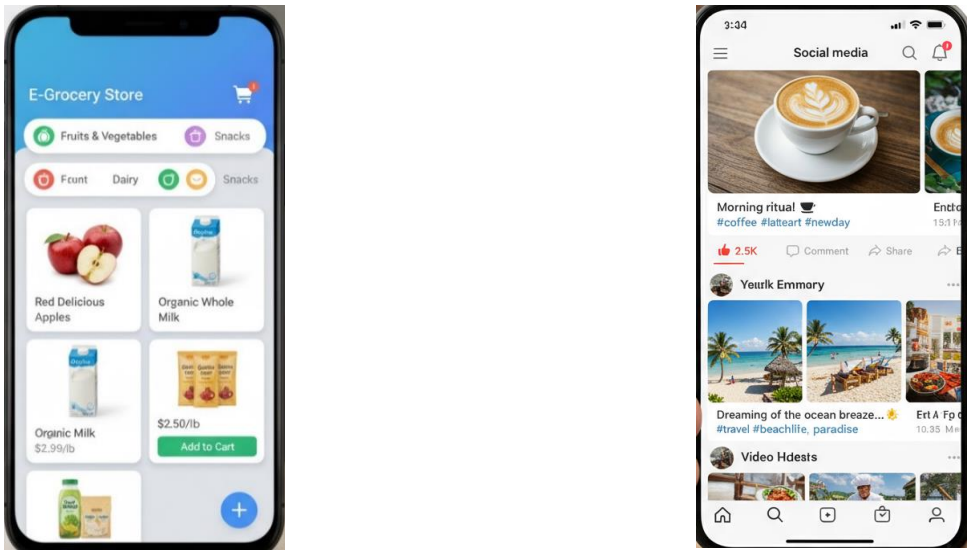
Metric	Android Studio	Flutter	Difference (%)
Development time (hours)	87.5	63.2	−27.8%
Lines of code	4,782	3,216	−32.7%
Build time (seconds)	42.3	18.7	−55.8%
Hot reload capability	Limited	Comprehensive	N/A

Flutter demonstrated a notable advantage in development efficiency, requiring 27.8% less development time and 32.7% fewer lines of code to implement identical functionality. Its *hot reload* feature significantly streamlined the coding and debugging process, allowing developers to visualize real-time modifications without recompiling the entire application. This capability proved especially valuable during interface refinement, where Android Studio required a full rebuild for each design iteration, extending overall development time. Performance benchmarking across multiple device configurations produced mixed outcomes, as presented in Table 2.

Table 2. Application Performance Metrics (Average Across Tested Devices)

Performance Metric	Android Studio	Flutter	Difference (%)
Launch time (ms)	1,842	2,103	+14.2%
Memory usage (MB)	78.4	92.6	+18.1%
CPU utilization (%)	12.3	15.7	+27.6%
Frame rendering time (ms)	16.4	11.8	−28.0%
Battery impact (mAh/hour)	142	156	+9.9%

Native Android applications developed through Android Studio exhibited superior resource efficiency, consuming 18.1% less memory and 27.6% less CPU power than their Flutter counterparts. This advantage suggests that native applications maintain tighter integration with the Android runtime environment, optimizing performance for resource-constrained devices. Conversely, Flutter applications achieved 28.0% faster frame rendering, resulting in smoother visual transitions and improved animation fluidity—an outcome particularly beneficial for UI-intensive applications such as e-commerce and social media platforms. User experience testing using the System Usability Scale (SUS) further supported these findings. Flutter obtained a slightly higher mean SUS score (82.4) compared with Android Studio (78.9), a statistically significant difference ($p < 0.05$) based on paired t-test analysis. Both scores fall within the “excellent” usability range, confirming that applications from both frameworks deliver satisfactory user interaction experiences. The Indonesian adaptation of SUS (Sharfina & Santoso, 2016) achieved a Cronbach’s α of 0.87, verifying internal consistency and reliability among respondents. Qualitative data from user interviews revealed recurring themes consistent with the quantitative findings. Approximately 76% of participants highlighted Flutter’s visual consistency, fluid animations, and cross-platform uniformity as positive attributes. Meanwhile, 68% of participants favored Android Studio’s native integration, stability, and compliance with Android’s design conventions. Developer feedback also underscored Flutter’s ability to expedite development cycles through its reusable widget structure, making it an advantageous choice for teams developing multi-platform products simultaneously.



(a) (b)

Figure 1. Prototype Applications: (a) Shopping App, (b) News Feed Interface

Figure 1 illustrates prototype examples developed during the study, including a shopping application (a) and a social media feed interface (b), both showcasing equivalent functionalities across frameworks.

4.2 Discussion

The findings reveal a clear trade-off between development efficiency and system performance when comparing Android Studio and Flutter frameworks. Flutter's substantial reductions in development time (−27.8%) and code volume (−32.7%) demonstrate its capacity to streamline the production cycle, particularly for applications requiring rapid iteration and multi-platform deployment. These results reinforce the conclusions of Feran *et al*. (2023), who identified Flutter's cross-platform architecture and *hot reload* functionality as key factors enabling faster prototyping and testing. In practice, this acceleration offers strategic value for startups or development teams operating under constrained timelines or seeking to validate early product versions before full-scale release. However, the efficiency benefits of Flutter come with measurable performance costs. The study's results indicate higher resource consumption—18.1% more memory and 27.6% greater CPU utilization—compared to applications developed natively through Android Studio. This performance gap aligns with observations by Andrean *et al*. (2020), who argued that native frameworks remain preferable for performance-critical applications or deployments targeting low-end hardware. The added resource load in Flutter applications likely stems from its rendering engine, which, while optimized for cross-platform consistency, introduces additional abstraction layers that increase processing demands. Despite these trade-offs, Flutter achieved 28% faster frame rendering, resulting in smoother animations and transitions. This performance advantage supports the claims of Pudjiarti and Faizah (2021), who noted Flutter's strength in delivering visually cohesive interfaces with consistent animation quality across devices. For applications emphasizing design aesthetics and fluid interaction—such as educational tools, interactive dashboards, or digital commerce platforms—this attribute can significantly enhance user engagement and perceived responsiveness.

Usability testing further contextualized these performance results. Both frameworks achieved "excellent" usability scores on the System Usability Scale (SUS), with Flutter scoring marginally higher (mean = 82.4) than Android Studio (mean = 78.9). The statistical significance ($p < 0.05$) suggests that the observed difference reflects a genuine user perception gap rather than random variation. These outcomes align with the usability evaluation framework proposed by Vlachogianni and Tselios (2021) and validated by Kaya *et al*. (2019), demonstrating that cross-platform tools like Flutter can deliver usability levels comparable to, or slightly exceeding, those of native applications. The Indonesian version of SUS (Sharfina & Santoso, 2016), which yielded a Cronbach's α of 0.87, further confirms the methodological reliability of this assessment for local participants. The qualitative feedback provides additional insight into these quantitative results. Participants perceived Flutter applications as more visually consistent and modern, while Android Studio applications were praised for their stability and conformity to Android's native design guidelines. These contrasting perspectives underscore that framework selection is context-dependent—projects prioritizing interface uniformity and development speed may benefit from Flutter, whereas applications requiring optimized performance and deeper hardware integration remain better suited to Android Studio. This finding mirrors the perspective of Wulan and Fauzi (2021), who emphasized the importance

of aligning framework capabilities with project objectives rather than adhering to a single development paradigm. In this respect, Flutter's flexibility makes it an appealing solution for organizations pursuing simultaneous Android and iOS releases, while Android Studio continues to serve as the optimal environment for applications where device-level optimization, security, and offline performance are paramount.

From a methodological standpoint, the integration of quantitative benchmarking with qualitative usability feedback proved valuable in producing a balanced evaluation. As suggested by Kaya *et al*. (2019) and Wali (2017), mixed-method approaches enable researchers to move beyond surface-level performance indicators by capturing experiential aspects of usability that directly influence user satisfaction and retention. Moreover, the consistency of findings across both datasets strengthens the interpretive validity of the conclusions and provides empirical evidence for practical decision-making in software development. In summary, the comparative analysis indicates that Flutter excels in development speed, cross-platform consistency, and user-perceived usability, while Android Studio maintains advantages in system resource efficiency and performance optimization. These differences should guide framework selection based on project priorities—whether rapid deployment and visual consistency or computational efficiency and native integration. As mobile ecosystems continue to diversify, understanding such trade-offs becomes essential for developers, researchers, and organizations aiming to balance technical performance with sustainable development practices.

5 | CONCLUSIONS

This comparative investigation of Android Studio and Flutter frameworks provides a nuanced understanding of the strengths and limitations that shape mobile application development outcomes. The empirical results indicate that Flutter offers pronounced advantages in development efficiency, reducing development time by 27.8% and code volume by 32.7% compared to native Android development. Its *hot reload* capability and single codebase architecture significantly accelerate iteration cycles, making it particularly effective for multi-platform projects, rapid prototyping, and applications emphasizing dynamic user interfaces. At the same time, the findings demonstrate that these efficiency gains are accompanied by higher resource consumption. Flutter-based applications utilized approximately 18.1% more memory and 27.6% more CPU resources than those built using Android Studio. These results reaffirm the continued relevance of native development for performance-sensitive applications or deployments targeting devices with limited hardware capacity. Nevertheless, Flutter's 28% faster frame rendering helps mitigate these concerns, particularly in animation-heavy or visually intensive environments where smooth interaction is critical to user satisfaction. Usability testing using the System Usability Scale (SUS) showed that both frameworks produce applications with strong user acceptance, with Flutter achieving a slightly higher average score (82.4) than Android Studio (78.9). Both fall within the *excellent* usability range, implying that differences in perceived usability are secondary to considerations of development speed and system performance. The successful use of the Indonesian SUS adaptation (Sharfina & Santoso, 2016), which achieved a Cronbach's α of 0.87, confirms the reliability of this instrument for evaluating mobile applications in localized contexts.

The overall findings suggest that framework selection should be guided by specific project priorities rather than generalized assumptions. Flutter's strengths in speed, maintainability, and cross-platform consistency make it well-suited for startups, MVP development, or organizations targeting multiple operating systems. In contrast, Android Studio remains the superior choice for resource-critical applications, extensive integration with native Android features, or projects requiring long-term optimization within a single platform environment. Methodologically, this study advances prior comparative research by combining quantitative performance benchmarks, usability metrics, and qualitative feedback within a single evaluation framework. This triangulated approach offers a more comprehensive and evidence-based foundation for assessing mobile development tools. The results provide not only empirical validation of performance trade-offs but also practical insights for developers and organizations balancing productivity with end-user experience. Several limitations should be acknowledged. The experimental design focused on a single application type (task management), which may not encompass the full diversity of mobile software scenarios. The participant pool of 42 users, while statistically valid, could be expanded in future work to reflect broader demographic variation. Additionally, performance testing was limited to specific hardware configurations, and outcomes may vary across the heterogeneous landscape of Android devices. Despite these constraints, the study contributes actionable insights for software engineers, project managers, and decision-makers tasked with selecting appropriate frameworks. As mobile ecosystems evolve, understanding the balance between development efficiency, system performance, and user experience becomes increasingly vital for sustainable and competitive digital innovation. The research design and metrics introduced here establish a replicable model for future investigations into emerging development frameworks and cross-platform technologies.

REFERENCES

Andrean, K., Armanto, H., & Pickerling, C. (2020). Sistem tempat parkir terintegrasi yang dilengkapi dengan aplikasi mobile dan mikrokontroler. *Journal of Information System Graphics Hospitality and Technology, 2*(1), 22–29. https://doi.org/10.37823/insight.v2i01.79

Arb, G., & Al-Majdi, K. (2020). A freights status management system based on Dart and Flutter programming language. *Journal of Physics: Conference Series, 1530*(1), 012020. https://doi.org/10.1088/1742-6596/1530/1/012020

Bhahri, S. (2021). E-tourism dalam pengenalan sektor pariwisata berbasis Android di Kota Makassar. *E-Jurnal Jusiti (Jurnal Sistem Informasi dan Teknologi Informasi), 10*(1), 94–106. https://doi.org/10.36774/jusiti.v10i1.824

Fauzi, M., Teddyyana, A., & Enda, D. (2021). Pengembangan aplikasi mobile tanggap bencana di Kabupaten Bengkalis menggunakan framework Flutter. *Zonasi Jurnal Sistem Informasi, 3*(1), 27–36. https://doi.org/10.31849/zn.v3i1.5856

Feran, S., *et al*. (2023). [Article title not provided]. *International Research Journal of Modernization in Engineering Technology and Science.* https://doi.org/10.56726/irjmets46051

Fitriastuti, F., Setyawan, R., Sudewo, Y., & Putra, J. (2022). Android-based application for user complaint as to support SLA implementation. *Journal of Physics: Conference Series, 2394*(1), 012034. https://doi.org/10.1088/1742-6596/2394/1/012034

Kaya, A., Ozturk, R., & Altin Gumussoy, C. (2019). Usability measurement of mobile applications with System Usability Scale (SUS). In F. Calisir, E. Cevikcan, & H. Camgoz Akdag (Eds.), *Industrial engineering in the big data era* (pp. 389–398). Springer. https://doi.org/10.1007/978-3-030-03317-0_32

Kurniawan, R., Rosiyadi, D., & Hardi, N. (2025). Rancang bangun aplikasi presensi instruktur berbasis Android menggunakan framework Flutter pada LKP Bright School Lampung Timur. *Reputasi: Jurnal Rekayasa Perangkat Lunak, 5*(2), 111–120. https://doi.org/10.31294/reputasi.v5i2.5871

Kustiawati, D., Sobiruddin, D., & Ani, F. (2025). Pengembangan media mobile learning menggunakan Flutter pada materi aljabar SMP. *Algoritma Journal of Mathematics Education, 7*(1), 66–77. https://doi.org/10.15408/ajme.v7i1.45135

Mustova, R., & Pramudwiatmoko, A. (2025). Pengembangan aplikasi sistem penjualan kaos berbasis Android untuk usaha konveksi. *Jurnal Informatika, 24*(2), 84–93. https://doi.org/10.30873/jurnalinformatika.v24i2.966

Nabawi, R., Tarigan, B. G., Saputra, P. B., & Sukmadiningtyas. (2024). Perancangan mobile app digital marketing Milkyo dengan metode prototype design. *Jurnal Indonesia: Manajemen Informatika dan Komunikasi, 5*(2), 1233–1244. https://doi.org/10.35870/jimik.v5i2.640

Nasution, A., Efendi, B., & Siregar, I. (2019). Pelatihan membuat aplikasi Android dengan Android Studio pada SMP Negeri 1 Tinggi Raja. *Jurdimas (Jurnal Pengabdian Kepada Masyarakat) Royal, 2*(1), 53–58. https://doi.org/10.33330/jurdimas.v2i1.321

Peres, S. C., Pham, T., & Phillips, R. (2013). Validation of the System Usability Scale (SUS): SUS in the wild. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 57*(1), 192–196. https://doi.org/10.1177/1541931213571043

Pudjiarti, E., & Faizah, S. (2021). Perancangan aplikasi penjualan berbasis Android sebagai media pemesanan pada distro online. *Bina Insani ICT Journal, 8*(2), 176–183. https://doi.org/10.51211/biict.v8i2.1589

Ramadhan, S., & Julkarnain, M. (2024). Mobile aplikasi manajemen keuangan CV. Top Digital Printing Dompu. *JURTIKOM, 2*(1), 13–22. https://doi.org/10.51401/jurtikom.v2i1.3248

Sari, J. (2025). Geographic information system for Aceh souvenir shop locations based on Android. *Journal of Digital Technology Trend, 4*(2), 100–116. https://doi.org/10.56347/jdtt.v4i2.272

Septiani, C., Valentine, V., Pranatawijaya, V., & Sari, N. (2024). Penerapan Flutter pada dashboard admin salon kecantikan "Bliss Beauty" berbasis mobile. *JATI (Jurnal Mahasiswa Teknik Informatika), 8*(3), 3962–3966. https://doi.org/10.36040/jati.v8i3.9794

Setyaka, G. I., Wulandari, S., & Handayani, I. (2024). Implementasi aplikasi reservasi tiket pada Hutan Pinus Kalilo Purworejo berbasis mobile. *Jurnal Indonesia: Manajemen Informatika dan Komunikasi, 5*(2), 2059–2069. https://doi.org/10.35870/jimik.v5i2.827

Sharfina, Z., & Santoso, H. B. (2016). An Indonesian adaptation of the System Usability Scale (SUS). In *Proceedings of the 2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS)* (pp. 145–148). IEEE. https://doi.org/10.1109/ICACSIS.2016.7872776

Suhendro, J., Sudarma, M., & Khrisne, D. (2021). Rancang bangun aplikasi seluler penyedia jasa perawatan dan kecantikan menggunakan framework Flutter. *Jurnal Spektrum, 8*(2), 68–75. https://doi.org/10.24843/spektrum.2021.v08.i02.p9

Tasik, H., Paembonan, S., & Muhallim, M. (2024). Penjualan kursi dan meja Jepara pada toko Arlan Mebel berbasis mobile di Luwu Utara. *Jurnal Informatika dan Teknik Elektro Terapan, 12*(3S1). https://doi.org/10.23960/jitet.v12i3s1.5163

Vlachogianni, P., & Tselios, N. (2021). Perceived usability evaluation of educational technology using the System Usability Scale (SUS): A systematic review. *Journal of Research on Technology in Education, 54*(3), 392–409. https://doi.org/10.1080/15391523.2020.1867938

Wali, M. (2017). Adsense mobile dan respon pengguna smartphone: Intrusiveness dan irritation. *Jurnal Ekonomi dan Manajemen Teknologi, 1*(2), 1–12.

Wulan, D., & Fauzi, A. (2021). Aplikasi mobile learning jurusan multimedia berbasis Android pada SMK 1 Anjatan Indramayu. *Jurnal Responsif Riset Sains dan Informatika, 3*(1), 53–62. https://doi.org/10.51977/jti.v3i1.400

How to cite this article: Diansyah, H., & Safrinal, S. (2025). Design and Development of a Mobile Application Using Android Studio and Flutter. *Journal Mobile Technologies (JMS)*, *3*(2), 69-76. https://doi.org/10.59431/jms.v3i2.646.