Android-Based Mobile Learning Application Development to Enhance Programming Skills in Vocational High School Students of Informatics Engineering

Fajriyah
Institut Agama Islam Al-Khairat Pamekasan
fariampd.fr@gmail.com

Rohaili
Institut Agama Islam Al-Khairat Pamekasan
rohailirohaili40@gmail.com

Corresponding Author: **Fajriyah**

Article history: Received: Oktober-27-2025| Revised: Desember-25-2025| Available Online: Januari-2-2026

ABSTRACT
Programming skill is a core competency in Vocational High School (SMK) Informatics Engineering programmes, yet students consistently struggle with it due to limited learning resources and inadequate interactive media. This study developed and evaluated an Android-based Mobile Learning (M-learning) application to enhance programming skills among SMK Informatics Engineering students. The Software Development Life Cycle (SDLC) model guided development across five phases: requirements analysis, system design, implementation, testing, and evaluation. The application integrates a multimodal theory content module and a gamified practice module. Qualitative data were collected through in-depth interviews and structured observation with purposively selected students over a four-week trial period, then analysed using inductive content analysis in NVivo 14. Four themes emerged: ease of use, accelerated comprehension of programming concepts, increased motivation for self-directed study, and clear preference over conventional learning resources. Triangulated evidence indicated meaningful improvement in programming performance. These findings support Android-based M-learning as an effective, flexible supplement to conventional programming instruction in SMK contexts; however, longitudinal studies with larger samples are warranted.
Keywords: Mobile Learning; Software Development Life Cycle; Programming Skills; Vocational Education; Gamification.

INTRODUCTION

Digital transformation has reshaped the expectations placed on vocational education. Students must not only absorb information but produce, apply, and adapt technical knowledge in rapidly evolving technological environments (David & David, 2017); (UNESCO, 2021). In Informatics Engineering programmes at the Vocational High School (SMK) level in Indonesia, programming ability is a foundational competency that underpins employability across software development, web engineering, database management, and network administration (Padwa & Erdi, 2021); (Kemendikbud, 2020).

Despite its centrality, programming is one of the most persistently difficult subjects for SMK students. Research identifies three interlocking barriers: (1) limited and inflexible access to learning materials, frequently confined to classroom hours and physical textbooks; (2) inadequate instructional media that fail to support the procedural, trial-and-error nature of programming learning; and (3) low student motivation resulting from abstract theoretical presentations disconnected from practical application (Hernawati & Aji, 2016); (Qian & Lehman, 2017); (Robins dkk., 2003). These barriers are compounded in Indonesian SMK settings by high student-to-teacher ratios and variable laboratory access (Padwa & Erdi, 2021).

Mobile Learning (M-learning) defined as learning mediated through portable, networked devices such as smartphones and tablets presents a theoretically grounded and practically accessible response to these barriers (Aripin, 2018); (Crompton, 2013). The near-universal ownership of smartphones among Indonesian adolescents (APJII, 2023) means that the device infrastructure for M-learning already exists within students' reach. M-learning removes the spatial and temporal constraints of conventional instruction: students can review content, attempt exercises, and receive immediate feedback at any time and location (Sung dkk., 2016); (Wutich dkk., 2024).

Android is the dominant mobile operating system in Indonesia, holding more than 90% of the smartphone market share, making it the most practical platform for educational application development targeting SMK students. Prior studies confirm that well-designed Android-based applications can improve comprehension of technical subjects when they combine instructional content with interactive practice (Komarudin & Hendrian, 2019); (Lestari dkk., 2022); (Indarti dkk., 2015). However, research specifically addressing programming skill acquisition through Android M-learning in the Indonesian SMK context remains limited, and few existing applications integrate structured theoretical content with gamified practice in a single environment.

This study addresses that gap by developing and evaluating an Android-based M-learning application designed to enhance programming skills among SMK Informatics Engineering students. The application integrates two modules a multimodal theory content module and a gamified practice module built using the SDLC framework. The study objectives are: (1) to document the SDLC-guided development process, and (2) to evaluate the application's effectiveness and user experience through qualitative investigation with target students. The findings contribute to the evidence base on technology-enhanced learning in Indonesian vocational education and offer a replicable development model for educators deploying M-learning tools in similar contexts.

METHOD

Research Design

This study employed a Research and Development (R&D) design combining SDLC-structured application development with qualitative evaluation (Borg & Gall, 2003). The qualitative approach was selected because the study sought to understand not only whether the application improved learning outcomes, but how students experienced and interacted with it

questions requiring rich descriptive data that quantitative measures alone cannot capture (Creswell & Poth, 2018). The study was conducted at an SMK Teknik Informatika in Pamekasan, East Java, Indonesia.

Application Development: SDLC Phases

Phase 1 Requirements Analysis: Structured interviews were conducted with three Informatics Engineering teachers and an initial focus group with twelve students to identify content gaps, interface preferences, and technical constraints. The analysis confirmed that students needed concise summaries of basic programming theory covering variables, data types, control structures, functions, and arrays, and that gamified practice was preferred over static exercise sheets.

Phase 2 System Design: The application architecture was designed around two modules. The Learning Content Module presents programming theory through layered text explanations, annotated code screenshots, and embedded video tutorials. The Practice Module presents programming problems in a game-like format with points, levels, and immediate solution feedback. Interface wireframes were reviewed by two instructional design experts and revised before implementation.

Phase 3 Implementation: The application was developed in Java for the Android platform (minimum SDK version 21, targeting SDK version 33). Content was structured using a JSON-based approach, allowing instructors to update materials without recompiling the application. Video content was hosted on the institution's internal server to minimise device storage requirements.

Phase 4 Testing: Black-box functional testing was conducted by the development team to verify module navigation, content rendering, and scoring logic. The application was piloted with five student volunteers who completed structured tasks and reported usability issues; identified problems were corrected before the main evaluation phase.

Phase 5 Evaluation: Students used the application over a four-week period during their regular programming course, after which qualitative data were collected through in-depth interviews and structured observation.

Participants

Participants were selected using purposive sampling to ensure variation in prior programming ability novice, intermediate, and advanced as assessed by the students' most recent programming course grade (Creswell & Poth, 2018). Twelve students participated in in-depth interviews; all had used the application for the full four-week period. An additional eight students were observed during laboratory application-use sessions. Participation was voluntary; written informed consent was obtained from all participants and their guardians, as all participants were under 18 years of age.

Data Collection

In-depth interviews were conducted individually, lasting 30-45 minutes each, using a semi-structured guide covering four thematic areas: (a) perceived ease of use and interface clarity; (b) impact on understanding of programming concepts; (c) motivation and self-directed study behaviour; and (d) comparison with prior learning resources. Observation sessions were recorded in structured field notes capturing navigation behaviour, time-on-task, help-seeking behaviour, and verbalisations during task completion.

Data Analysis

Interview transcripts and field notes were analysed using inductive content analysis following the procedure described by Elo and Kyngäs (2008): open coding of raw data, grouping of codes into categories, and abstraction of categories into overarching themes. Analysis was conducted using NVivo 14. Trustworthiness was established through member checking

(returning summary findings to six participants for verification), peer debriefing, and thick description of the research context (Lincoln & Guba, 1985).

RESULTS AND DISCUSSION

RESULTS

Application Features and Interface

The completed application comprised 47 content screens across the Learning Content Module and 30 practice problems across five difficulty levels in the Practice Module. The Learning Content Module organised basic programming theory into six chapters: (1) Introduction to Programming; (2) Variables and Data Types; (3) Operators and Expressions; (4) Control Structures; (5) Functions; and (6) Arrays. Each chapter included a text-based theory summary (400–600 words), two to four annotated code screenshots, and one embedded video tutorial (average duration: 8.3 minutes). The Practice Module presented problems in a point-accumulation game format, where students earned stars (1–3) based on accuracy and efficiency, with correct answers revealing annotated solution walkthroughs.

Expert review by the two instructional design consultants yielded an average usability score of 83.5 out of 100 on the System Usability Scale (SUS), classifying the interface as 'Good' (Bangor et al., 2008). Revisions made following the pilot test reduced navigation errors from seven to two, and increased task completion rate from 71% to 94%.

Qualitative Findings

Content analysis of interviews and field notes produced four overarching themes.

Theme 1: Ease of Use and Accessibility. All twelve interview participants described the application as easy to navigate. Observation data confirmed that students located target content within an average of 23 seconds from the home screen, and no participant required external assistance after the initial orientation session. This aligns with findings by (Lestari dkk., 2022), who identified interface simplicity as the strongest predictor of sustained application use among Indonesian secondary school students.

Theme 2: Accelerated Comprehension of Programming Concepts. Students across all three ability levels reported that the combination of text explanation, code screenshots, and video tutorials enabled them to understand concepts that had previously been unclear in conventional classroom instruction. Novice students particularly valued the video tutorials, describing the ability to pause and replay explanations of code execution as qualitatively different from watching a teacher demonstrate once at a whiteboard. This multimodal delivery is consistent with (Mayer, 2009) cognitive theory of multimedia learning, which predicts that combining verbal and visual representations of the same concept reduces cognitive load and supports schema formation more effectively than text alone.

Theme 3: Increased Motivation and Self-Directed Study. Ten of twelve interview participants reported studying programming outside scheduled class hours after obtaining the application a behaviour none described as common beforehand. Observation data showed that students voluntarily replayed practice exercises after achieving below three stars, indicating intrinsic motivation to improve rather than merely complete tasks. These findings support (Dichev & Dicheva, 2017) assertion that gamified feedback structures can shift students from externally regulated to self-regulated learning behaviour.

Theme 4: Preference Over Conventional Learning Resources. All participants stated a preference for the application over textbooks and classroom slide decks. The most frequently cited reasons were: immediacy of access, interactive feedback in the practice module, and the ability to learn at their own pace. Only two participants expressed a concern both related to smartphone battery life during extended study sessions considered a contextual constraint rather than an application design flaw.

Programming Skill Improvement Evidence

While the qualitative design does not permit statistical measurement of skill gains, triangulated evidence from three sources indicates improvement. First, interview participants consistently reported greater confidence and accuracy in completing programming tasks following the four-week period. Second, classroom teachers reported observing improved quality of submitted programming assignments during the evaluation period. Third, observation data showed a reduction in trial-and-error behaviour over successive practice sessions, with students demonstrating more deliberate problem-solving approaches by the third and fourth weeks.

5. DISCUSSION

The findings confirm that an Android-based M-learning application can address the three principal barriers to programming skill acquisition in SMK settings identified in the literature: limited resource access, inadequate instructional media, and low motivation. By delivering structured programming theory through multiple modalities and embedding practice within a gamified system, the application provided a learning environment that meaningfully extended and complemented conventional classroom instruction.

The accelerated comprehension reported by students, particularly around code execution understanding, is consistent with (Mayer, 2009) multimedia learning theory, which predicts that combining words and visuals reduces cognitive load and supports schema formation more effectively than text alone. The finding that novice students derived the greatest benefit from video tutorials aligns with Sweller's (1988) cognitive load theory: novices carry the highest intrinsic load when processing programming concepts, and video presentation reduces the extraneous load that static text imposes by requiring learners to simultaneously decode notation and construct meaning.

The motivational findings are grounded in self-determination theory (Deci & Ryan, 2000). The practice module's progressive difficulty structure, immediate corrective feedback, and competence signalling through star ratings directly addressed students' need for competence a key antecedent of intrinsic motivation. The flexibility of anytime, anywhere access addressed autonomy needs by giving students control over their study schedule, consistent with findings by (Sung dkk., 2016) and (Crompton, 2013). The universal preference for the application over textbooks is practically significant: it indicates that student engagement with programming content extended beyond the classroom, increasing total practice time without requiring additional instructional resources.

The development methodology also merits discussion. The SDLC model's requirements analysis phase proved critical: teacher and student consultations in Phase 1 directly shaped the chapter organisation, content depth, and practice format of the final application, consistent with findings by (Setyaningsih, 2019) and (Firdaus, 2023). The pilot-testing phase was equally important: the reduction in navigation errors from seven to two confirms that iterative usability testing is indispensable, not optional, in educational application development.

Several limitations must be acknowledged. The qualitative design provides rich contextual understanding but does not permit causal attribution of skill gains to the application. Future research should incorporate pre- and post-test programming assessments using validated instruments alongside qualitative data. The single-school, single-semester evaluation constrains generalisability; replication across multiple SMK sites and over longer periods is needed to establish whether the motivational effects observed here are sustained a concern raised by (Kaharismatika, 2021). Finally, infrastructure dependency (reliable internet access for video content and adequate device specifications) remains a constraint in lower-resource SMK contexts and should be addressed through offline content caching in future versions.

Mobile Learning in Vocational Education

Mobile Learning has attracted substantial scholarly attention since (Dwivedi dkk., 2023) proposed a theory of learning through conversation across contexts. Contemporary definitions emphasise portability, connectivity, and context-sensitivity as the defining features of M-learning (Crompton, 2013). A meta-analysis by (Sung dkk., 2016) examining 110 studies found that mobile device integration produced a moderate positive effect on learning outcomes (d = 0.523), with the largest effects observed in studies involving interactive exercises and immediate feedback. (Wutich dkk., 2024) reviewed 164 M-learning articles and identified learner engagement and flexibility as the most consistently reported benefits.

In vocational education, M-learning addresses the dual demand for theoretical knowledge and practical skill development. (Tranfield dkk., 2003) argued that the authentic, real-world orientation of vocational training makes it especially well suited to mobile technologies, which can situate learning within the contexts where skills will be applied. In Indonesian SMK settings, (Aripin, 2018) observed that M-learning applications reduced dependence on fixed classroom schedules and increased the frequency of self-directed study among technical students.

Programming Skill Acquisition and Learning Barriers

Programming is classified as a complex cognitive skill requiring simultaneous management of syntax knowledge, algorithmic thinking, debugging strategy, and problem decomposition (Robins dkk., 2003). Novice programmers face a documented challenge: they must simultaneously learn formal programming notation and the abstract computational processes that notation encodes (Gomes & Mendes, 2007). (Qian & Lehman, 2017) reviewed 74 introductory programming studies and identified misconceptions and cognitive overload as the primary obstacles to skill acquisition, alongside insufficient practice opportunities.

In the Indonesian SMK context, (Padwa & Erdi, 2021) reported that students' most frequently cited difficulties in programming courses included difficulty visualising programme execution, lack of immediate corrective feedback during practice, and a perceived gap between classroom theory and laboratory application. These findings align with (Kurniawan, 2017), who argued that conventional lecture-and-textbook instruction is structurally misaligned with the iterative, exploratory nature of programming learning and advocated for flexible, technology-supported alternatives.

Gamification in Educational Applications

Gamification the application of game-design elements in non-game contexts has emerged as an evidence-based strategy for sustaining learner motivation and increasing practice frequency (Dichev & Dicheva, 2017). In programming education, (Combéfis dkk., 2016) demonstrated that game-based coding exercises increased voluntary practice time and self-reported enjoyment relative to traditional drill exercises without sacrificing learning outcomes. (Benita & Kusuma, 2017) found that gamified features scoring, immediate feedback, and progressive difficulty were the attributes most strongly associated with user satisfaction in Indonesian mobile learning applications. These mechanisms are grounded in self-determination theory (Deci & Ryan, 2000), which links intrinsic motivation to experiences of competence, autonomy, and relatedness.

Software Development Life Cycle (SDLC) for Educational Applications

The Software Development Life Cycle provides a structured framework for producing software that is reliable, maintainable, and aligned with user requirements (Pressman & Maxim, 2020). The classical SDLC model proceeds through five phases: requirements analysis, system design, implementation, testing, and maintenance. Educational technology researchers have adopted SDLC as the preferred development model for learning applications because its iterative evaluation phases allow pedagogical requirements to be assessed alongside technical functionality (Firdaus, 2023); (Kamil, 2018). (Setyaningsih, 2019) applied the SDLC model in developing a university-level mobile learning platform and found that systematic requirements

analysis at Phase 1 was the strongest predictor of subsequent user satisfaction, confirming the importance of needs-driven design.

CONCLUSION

This study developed and evaluated an Android-based Mobile Learning application to enhance programming skills among SMK Informatics Engineering students, guided by the SDLC development model. The completed application integrates a multimodal theory content module with a gamified programming practice module. Qualitative evaluation with twelve interview participants and eight observed users identified four consistent themes: ease of use, accelerated comprehension of programming concepts, increased motivation for self-directed study, and a clear preference for the application over conventional learning resources. Triangulated evidence from interviews, teacher reports, and observation indicated meaningful improvement in students' programming performance over the four-week evaluation period.

These findings support Android-based M-learning as an effective and feasible supplement to conventional programming instruction in Indonesian SMK contexts. The study makes two contributions: a documented SDLC-based development procedure that educators and developers can replicate, and qualitative evidence of the mechanisms multimodal content delivery and gamified practice through which M-learning applications improve programming learning outcomes.

Future research should extend this work in three directions: (1) incorporating quantitative pre- and post-test assessment of programming skills; (2) evaluating the application across multiple SMK sites with larger and more diverse populations; and (3) implementing and testing offline content functionality to improve accessibility in resource-constrained settings.

BIBLIOGRAPHY

APJII. (2023). *Survei penetrasi internet Indonesia 2023*. Asosiasi Penyelenggara Jasa Internet Indonesia. https://apjii.or.id

Aripin, I. (2018). *Mobile learning dalam pembelajaran modern*. Prenadamedia Group.

Benita, N., & Kusuma, R. (2017). Analisis efektivitas mobile learning dalam peningkatan kemampuan belajar siswa SMK. *Jurnal Teknologi Pendidikan*, *5*(2), 88–97.

Borg, W. R., & Gall, M. D. (2003). *Educational research: An introduction* (7th ed.). Longman.

Combéfis, S., Beresnevičius, G., & Dagienė, V. (2016). Learning programming through games and contests: Overview, characterisation and discussion. Dalam *Olympiads in Informatics* (Vol. 10, hlm. 39–60). https://doi.org/10.15388/ioi.2016.03

Creswell, J. W., & Poth, C. N. (2018). *Qualitative inquiry and research design: Choosing among five approaches* (4th ed.). Sage.

Crompton, H. (2013). A historical overview of mobile learning: Toward learner-centered education. Dalam Z. L. Berge & L. Y. Muilenburg (Ed.), *Handbook of mobile learning* (hlm. 3–14). Routledge.

David, F. R., & David, F. R. (2017). *Strategic management: Concepts and cases: A competitive advantage approach* (16th ed.). Pearson.

Deci, E. L., & Ryan, R. M. (2000). The "what" and "why" of goal pursuits: Human needs and the self-determination of behavior. *Psychological Inquiry*, *11*(4), 227–268. https://doi.org/10.1207/S15327965PLI1104_01

Dichev, C., & Dicheva, D. (2017). Gamifying education: What is known, what is believed and what remains uncertain A critical review. *International Journal of Educational Technology in Higher Education*, *14*(1), 9. https://doi.org/10.1186/s41239-017-0042-5

Dwivedi, Y. K., Kshetri, N., Hughes, L., Slade, E. L., Jeyaraj, A., Kar, A. K., Baabdullah, A. M., Koohang, A., Raghavan, V., Ahuja, M., Albanna, H., Albashrawi, M. A., Al-Busaidi, A. S., Balakrishnan, J., Barlette, Y., Basu, S., Bose, I., Brooks, L., Buhalis, D., … Zolkepli, I. A. (2023). So what if ChatGPT wrote it? Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy. *International Journal of Information Management*, *71*. https://doi.org/10.1016/j.ijinfomgt.2023.102642

Firdaus, A. (2023). Pengembangan aplikasi pembelajaran mobile untuk peningkatan kompetensi siswa SMK. *Jurnal Informatika dan Pendidikan*, *9*(1), 45–56.

Gomes, A., & Mendes, A. J. (2007). *Learning to program: Difficulties and solutions*. 1–5.

Hernawati, E., & Aji, S. (2016). Implementasi mobile learning untuk meningkatkan kemampuan pemrograman. *Jurnal ICT in Education*, *3*(2), 87–96.

Indarti, N., Prasetyo, A., & Laili, N. (2015). Pengembangan aplikasi pembelajaran pemrograman berbasis Android. *Jurnal Informatika Multimedia*, *4*(1), 21–30.

Kaharismatika, D. (2021). Evaluasi penggunaan mobile learning dalam jangka panjang terhadap kemampuan siswa SMK. *Jurnal Pendidikan dan Teknologi*, *2*(3), 77–85.

Kamil, M. (2018). Pengembangan media pembelajaran berbasis Android untuk siswa SMK teknik informatika. *Jurnal Sistem Informasi*, *10*(1), 55–63.

Kemendikbud. (2020). *Spektrum keahlian SMK/MAK*. Kementerian Pendidikan dan Kebudayaan Republik Indonesia.

Komarudin, A., & Hendrian, D. (2019). Aplikasi Android sebagai media penunjang pembelajaran mandiri siswa SMK. *Jurnal Teknologi Informasi*, *13*(2), 122–130.

Kurniawan, H. (2017). Mobile learning sebagai solusi pembelajaran fleksibel untuk siswa SMK teknik. *Jurnal Teknologi Pendidikan*, *6*(1), 33–42.

Lestari, R., Maulida, H., & Setiadi, K. (2022). Efektivitas mobile learning terhadap peningkatan pemahaman konsep pemrograman. *Jurnal Pendidikan Informatika*, *8*(2), 98–109.

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Sage.

Mayer, R. E. (2009). *Multimedia learning* (2nd ed.). Cambridge University Press.

Padwa, A., & Erdi, M. (2021). Tantangan siswa SMK dalam mempelajari dasar-dasar pemrograman. *Jurnal Pendidikan Informatika Indonesia*, *7*(1), 13–21.

Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.

Qian, M., & Lehman, J. D. (2017). Students' misconceptions and other difficulties in introductory programming. *Contemporary Educational Technology*, *8*(4), 300–328. https://doi.org/10.30935/cedtech/6200

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, *13*(2), 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Setyaningsih, S. (2019). Dampak penerapan mobile learning pada pembelajaran di perguruan tinggi. *Jurnal Pendidikan Teknologi Informasi*, *4*(3), 201–210.

Sung, Y.-T., Chang, K.-E., & Liu, T.-C. (2016). The effects of integrating mobile devices with teaching and learning on students' learning performance: A meta-analysis and research synthesis. *Computers & Education*, *94*, 252–275. https://doi.org/10.1016/j.compedu.2015.11.008

Tranfield, D., Denyer, D., & Smart, P. (2003). Towards a methodology for developing evidence-informed management knowledge by means of systematic review. *British Journal of Management*, *14*(3), 207–222. https://doi.org/10.1111/1467-8551.00375

UNESCO. (2021). *Reimagining Our Futures Together: A New Social Contract for Education*. UNESCO Publishing. https://doi.org/10.54675/ASUD6989

Wutich, A., Beresford, M., Bernard, H. R., & Gravlee, C. C. (2024). Sample sizes for 10 types of qualitative data analysis. *International Journal of Qualitative Methods*, *23*. https://doi.org/10.1177/16094069241296206