Journal of Applied Informatics for Sustainability and Artificial Intelligence
AISAI Journal
https://jurnal.polbeng.ac.id/index.php/AISAI

Volume 1 Issue 1 Year 2026 | Page 16-23 | e-ISSN: xxxx-xxxx | ISSN: xxxx-xxxx
Received: 18-01-2026 | Revised: 25-01-2026| Accepted: 10-02-2026

Developing a Used Goods Trading Application Using MVVM and the Prototype Method

Arik Al-Farabi[1], Ryci Rahmatil Fiska[2], I Gusti Agung Putu Mahendra[3]
[1,2,3] Politeknik Negeri Bengkalis, Bengkalis, Riau, Indonesia, 28711
E-mail: *arikalfarabi123@gmail.com[1]* , *rycirf@polbeng.ac.id[2]*, *agungmahendra@polbeng.ac.id[3]*
*Correspondence: *arikalfarabi123@gmail.com*

Abstrak: *The rapid growth of digital platforms has transformed how secondhand goods are traded, especially among Generation Z. However, existing platforms often fail to address local market needs, particularly in areas like Bengkalis, where users struggle to find relevant items due to poor localization and unintuitive interfaces. This research aims to develop an Android-based secondhand goods trading application by applying the MVVM (Model-View-ViewModel) architecture and the Prototype method. The application includes core features such as product listing, real-time chat, location-based search, category filtering, and user profile management. The development follows five stages of the Prototype method: communication, quick plan, modeling quick design, construction of prototype, and deployment with user feedback. Evaluation results show that the application functions effectively, offers a responsive and intuitive interface, and meets user expectations—especially in terms of performance, stability, and usability. Comparative benchmark testing confirms that the MVVM-based version is 37% faster in loading time, uses 27% less memory, and reduces database requests by 83% compared to a non-MVVM counterpart. This research successfully demonstrates that MVVM architecture, when combined with the Prototype method, yields a maintainable, scalable, and user-friendly application tailored for the local secondhand market in Bengkalis.*

Keywords: *Android, MVVM, Prototype, Secondhand Goods, Application*.

1. Introduction

The rapid growth of digital technology has created significant opportunities to facilitate efficient and sustainable second-hand goods transactions through digital platforms [1]. In the era of the circular economy, such platforms serve not only as transaction channels but also as instruments to reduce waste and extend product lifecycles [2]. However, in regions such as Bengkalis, users often face difficulties in finding locally available items through popular applications such as OLX or Facebook Marketplace due to limited location-based filtering features and overly complex interfaces [3].

Generation Z, as active users of digital platforms, tends to prefer applications that are simple, fast, and tailored to local needs [4]. They prioritize features such as location-based search, direct chat with sellers, clear categories (e.g., electronics, fashion, vehicles), and a streamlined selling process-typically limited to uploading photos and entering the item name, price, and description [5].

To address these challenges, this study develops an Android-based second-hand marketplace application by applying the MVVM (Model-View-ViewModel) architecture and the Prototype development method [6]. MVVM is selected due to its ability to separate business logic from the

user interface, thereby improving code readability, testability, and application performance [7]. Meanwhile, the Prototype method enables rapid iteration based on user feedback, ensuring that the resulting solution closely aligns with local requirements [8].

From a theoretical perspective, this research contributes to the literature by extending the application of MVVM within the context of local e-commerce and validating the effectiveness of the Prototype method [9]. From a practical perspective, the proposed application provides a relevant digital solution for the Bengkalis community and may serve as a reference for developers building similar applications in remote areas [10].

2. Research Methods

This study employs the Prototype method [11], which consists of five main stages: requirements gathering, quick design, prototype development, user evaluation, and system refinement. Each stage is carried out iteratively to ensure that the developed system aligns with user requirements and effectively addresses the identified problems. illustrated in Figure 1.

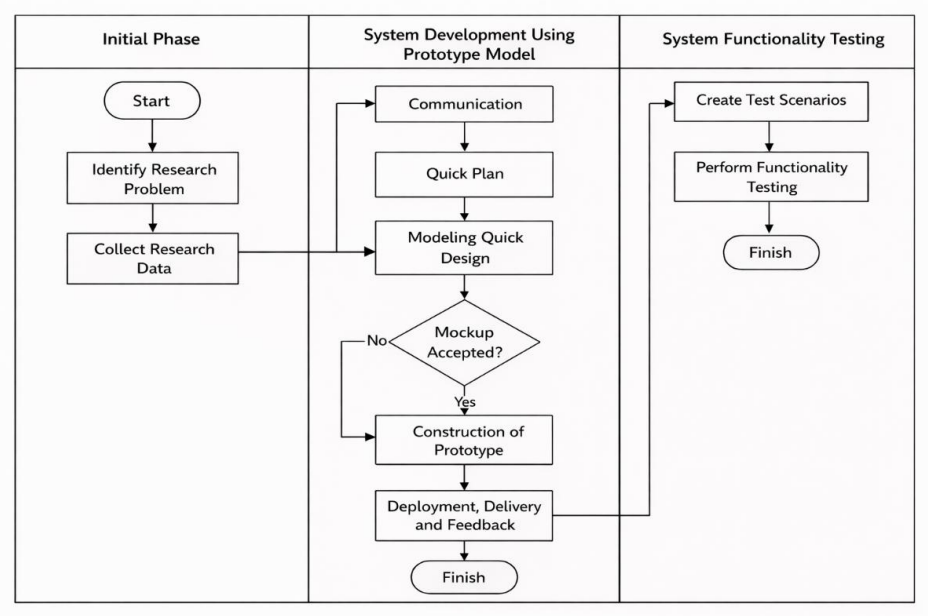

Figure 1. Research Flow

2.1 Prototype Development Stages

This study applies the Prototype method through five iterative stages to ensure that the developed application aligns with local user needs in Bengkalis and can be refined efficiently based on feedback:

1. Communication: Requirements were elicited through interviews and questionnaires with prospective users in Bengkalis to capture user pain points, feature expectations, and usage context.
2. Quick Plan: A rapid project plan was prepared, covering budget estimation, development timeline, and risk identification (e.g., scope changes, data connectivity constraints, and usability risks).
3. Modeling Quick Design: The initial solution was designed by producing UI/UX prototypes in Figma and modeling the system using UML artifacts, particularly Use Case Diagrams and Activity Diagrams, to clarify functional requirements and user interaction flows.

4. Construction of Prototype: The prototype was implemented using Flutter and structured with the MVVM (Model–View–ViewModel) architecture to improve maintainability, modularity, and testability.
5. Deployment, Delivery, and Feedback: The prototype was deployed for user trials. Users conducted hands-on testing, and their feedback was systematically collected and used to guide iterative improvements until the application met functional and usability expectations.

2.2 MVVM Architecture Implementation

The MVVM architecture was applied consistently across the application to enforce separation of concerns and support scalable development can be seen in the Figure 2 and 3.

1. Model: The Model layer manages core data entities such as product listings and chat records. Data persistence and synchronization are handled through Firebase Firestore, ensuring centralized storage and real-time updates.
2. ViewModel: The ViewModel encapsulates the application's business logic, including real-time search processing, data validation, and state management. It acts as an intermediary between the View and the Model, exposing structured data and operations to the UI layer.
3. View: The View layer is responsible for rendering the user interface and presenting information to users. It does not contain business logic; instead, it interacts with the system exclusively through the ViewModel (e.g., observing state changes and invoking ViewModel actions).

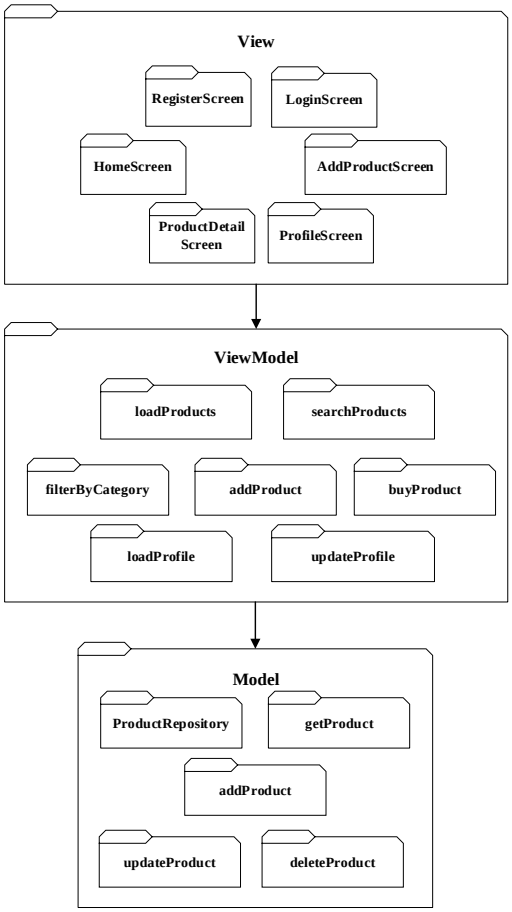

Figure 2. Workflow MVVM

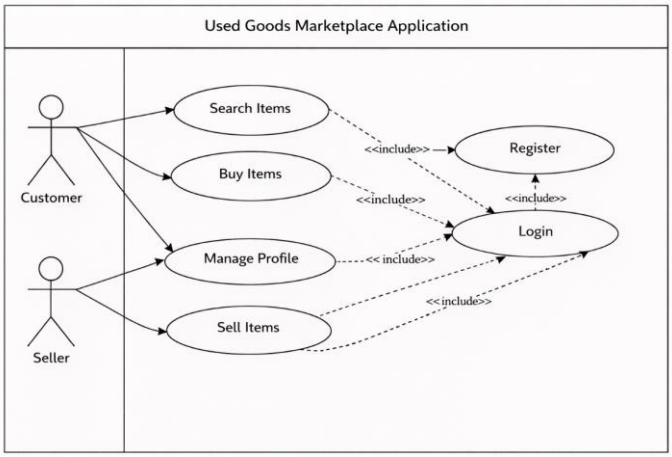

Figure 3. Use Case Diagram

3. Results and Discussion

The following section presents the main interfaces of the developed application.

3.1 Home Page

The Home page displays a list of second-hand items arranged in a grid layout, complemented by a search bar and category filters. This design enables users to quickly browse items and efficiently narrow results based on their preferences can be seen in the Figure 4.

Figure 4. Home Page

3.2 Item Detail Page

This page provides comprehensive product information, including product images, price, description, category, condition, and seller profile. The detailed presentation supports informed purchasing decisions and improves user trust can be seen in the Figure 5.

Figure 5. Item Detail Page

3.3 Seller Chat Page

The Seller Chat page enables real-time communication between buyers and sellers using Firestore Streams. This feature facilitates instant negotiation and information exchange, enhancing transaction efficiency can be seen in the Figure 6.

Figure 6. Seller Chat Page

3.4 Manage Listed Items Page

This page allows sellers to add, edit, and delete their listed items through a simple and intuitive form. The interface is designed to minimize user effort while maintaining full control over product listings can be seen in the Figure 7.

Figure 7. Manage Listed Items Page

3.5 Testing Results

1. Black-Box Testing

Black-box testing results indicate that all functional scenarios were successfully executed, confirming that the system meets the defined functional requirements.

2. User Testing

User evaluation was conducted with 21 respondents, yielding the following results:

- 100% of users reported that the application never crashed during usage.
- 86% of users stated that they were satisfied to very satisfied with the application.
- The primary user recommendation was the addition of push notification features for chat functionality.

3.6 Comparative Benchmark Evaluation

To assess the impact of the MVVM architecture, a comparative benchmark was conducted between MVVM-based and non-MVVM implementations:

1. Page Load Time:
    - MVVM: 0.85 seconds
    - Non-MVVM: 1.35 seconds
2. Memory Usage:
    - The MVVM-based application demonstrated 27% lower memory consumption.
3. Database Requests:
    - MVVM implementation: 6 requests
    - Non-MVVM implementation: 35 requests

These results indicate that the adoption of the MVVM architecture significantly improves application performance, resource efficiency, and system scalability, making it suitable for mobile marketplace applications with real-time interaction requirements.

4. Conclusions

Based on the findings of this study, the following conclusions can be drawn:

1. MVVM Architecture Effectiveness:

   The implementation of the Model–View–ViewModel (MVVM) architecture successfully produced an application that is well-structured, maintainable, and stable. By enforcing separation of concerns between the user interface, business logic, and data layer, MVVM improved code organization and reduced coupling, which supports long-term maintenance and easier feature extension.

2. Prototype Method Suitability:

   The Prototype development method proved effective in ensuring that the application aligns with the needs of local users. Rapid iterations and continuous user feedback enabled the research team to refine features and interface elements early, minimizing requirement mismatches and improving overall usability.

3. User Experience Quality:

   The developed application delivers a user experience that is responsive, intuitive, and efficient. Key interactions such as item browsing, product detail viewing, and real-time seller communication can be performed smoothly with minimal steps, supporting practical use in everyday transactions.

4. Empirical Performance and Code Quality Gains:

   Benchmark results provide empirical evidence that MVVM improves both runtime performance and code quality. The architecture contributes to faster loading times, more efficient memory utilization, and reduced database requests, while also facilitating cleaner, more testable implementation.

To further enhance the system and broaden its applicability, the following improvements are recommended:

1. Real-Time Notifications: Integrate Firebase Cloud Messaging (FCM) to enable real-time push notifications, especially for chat messages and transaction updates.
2. Digital Payment Support: Add digital payment functionality through e-wallet integration to streamline transactions and increase user convenience.
3. iOS Deployment: Extend the application to an iOS version to reach a wider user base and support cross-platform adoption.
4. Long-Term Scalability: Implement Clean Architecture to strengthen scalability, enforce clearer layer boundaries, and support sustainable growth as the application becomes larger and more complex.

References

[1] F. F. Anhar and F. T. Anggraeny, “Implementasi Clean Architecture MVVM dan Repository Pattern Untuk Pengembangan Aplikasi Android Jual Beli Barang Bekas SecondHand,” *Scan : Jurnal Teknologi Informasi dan Komunikasi*, vol. 17, no. 2, Nov. 2022, doi: 10.33005/scan.v17i2.3317.

[2] B. Arfianto, “Analisis Perbandingan Performa Pola Arsitektur Model-View-ViewModel (MVVM) dan Model-View-Presenter (MVP) pada Pengembangan Aplikasi Desa Wisata Berbasis Android,” *DGviuAtXwkNhoZJp2zVewVfh5DpMjucQVKm1dL5csisv*, vol. 6, no. 2, pp. 1–10, 2024.

[3] E. Arribe, Aryanto, and R. Asrianto, “Aplikasi E-Marketplace Menggunakan Arsitektur Mvvm (Model-View-Viewmodel) Berbasis Android,” *JURNAL FASILKOM*, vol. 11, no. 2, pp. 75–78, Aug. 2021, doi: 10.37859/jf.v11i2.2762.

[4] I. M. Riyadhi, Intan Purnamasari, and Kamal Prihandani, “Penerapan Pola Arsitektur Mvvm Pada Perancangan Aplikasi Pengaduan Masyarakat Berbasis Android,” *INFOTECH journal*, vol. 9, no. 1, pp. 147–158, May 2023, doi: 10.31949/infotech.v9i1.5246.

[5] M. R. M. Nanda, “Penerapan Aplikasi Pengingat Jadwal Kegiatan Penggabungan Waktu dan Tanggal Berbasis Android,” *OKTAL: Jurnal Ilmu Komputer dan Sains*, vol. 2, no. 7, pp. 1–10, 2023.

[6] J. R. W. Simanullang, “Aplikasi Penjualan Produk IT dan Jasa Service Berbasis Android Dengan Metode MVVM,” *Jurnal TEKINKOM*, vol. 7, no. 1, pp. 22–27, 2024.

[7] H. Ahsari, “Aplikasi Panduan Budidaya Okra Sistem Pendjawalan Alarm Otomatis Berbasis Android Dengan Thunkable,” *INTECHNO Journal - Information Technology Journal*, vol. 1, no. 2, pp. 5–15, 2020.

[8] Y. Darmi, S. Admira, A. K. Hidayah, and P. Pahrizal, “Aplikasi Kalender Kehamilan dan Perhitungan Masa Usia Kehamilan Berbasis Android Menggunakan Algoritma Naegele,” *JURNAL MEDIA INFOTAMA*, vol. 18, no. 2, pp. 328–336, Oct. 2022, doi: 10.37676/jmi.v18i2.2890.

[9] F. Maulana, “Aplikasi Manajemen Laboratorium Menggunakan Metode MVVM Berbasis Android,” *JITSI*, vol. 3, no. 3, pp. 32–44, 2022.

[10] S. Rosad, A. Yudhana, and A. Fadlil, “Jadwal Sholat Digital Menggunakan Metode Ephemeris Berdasarkan Titik Koordinat Smartphone,” *IT JOURNAL RESEARCH AND DEVELOPMENT*, vol. 3, no. 2, pp. 30–43, Jan. 2020, doi: 10.25299/itjrd.2019.vol3(2).2285.

[11] Tedyyana, A., Ahmad, A. A., Idrus, M. R., Mohd Shabli, A. H., Abu Seman, M. A., Ghazali, O., & Abd Razak, A. H. (2024). Enhance Telecommunication Security Through the Integration of Support Vector Machines. *International Journal of Advanced Computer Science & Applications*, *15*(3).