Int. j. adv. multidisc. res. stud. 2026; 6(1):1251-1253

Received: 02-11-2025
Accepted: 12-01-2026

International Journal of Advanced Multidisciplinary Research and Studies

ISSN: 2583-049X

Design and Development of Village Banking Mobile App

[1] Kabuuta Chisanga, [2] Henry Sinkala

[1, 2] Department of Information and Communication Technology, School of Engineering, Information and Communications University Lusaka, Zambia

Corresponding Author: **Kabuuta Chisanga**

Abstract

Village banking has long provided vital financial services to underserved communities, especially in sub-Saharan Africa. Manual, paper-based record-keeping makes these schemes inefficient, error-prone, and hard to scale. This paper presents the design and development of a Village Banking Mobile App that automates contributions, loan management, and interest calculations; introduces collateral-backed lending to mitigate default risk; and supplies members with real-time statements and notifications. Built with React Native (JavaScript/TypeScript), Expo Router, Remix Run, Metro, Babel, SQLite, and Node.js, the solution improves transparency, accessibility, and operational efficiency, ultimately advancing financial inclusion.

Keywords: Village Banking, Mobile Banking, Microfinance, React Native, Financial Inclusion, Collateral, SQLite

1. Introduction

1.1 Background

Village banking fills a critical financial-access gap in many parts of Zambia and wider sub-Saharan Africa [1-4]. Yet reliance on hand-written ledgers leads to errors, lost data, and slow service [5].

1.2 Problem Statement

Current village banks struggle with inefficiencies, limited reach, transparency issues, and loan-default risk when collateral cannot be enforced [6-9].

1.3 Research Questions

1. How can automation improve record accuracy and transparency?
2. How can a mobile interface remain usable across literacy levels?
3. Which security measures best build trust among first-time digital users?

1.4 Aim and Objectives

The project aims to create a mobile system that (i) digitizes savings, loans, and member contributions; (ii) adds collateral-based lending; (iii) supplies real-time data access; (iv) sends automated reminders; and (v) evaluates usability and impact.

1.5 Justification

Modernizing village banking boosts scalability and member trust, supporting broader economic empowerment.

1.6 Ethical Considerations

The system enforces data privacy, informed consent, transparency, and non-coercive participation while complying with GDPR-like principles.

2. Related Work

Mobile money platforms such as M-Pesa and Kenya's M-Chama demonstrate the power of mobile solutions for group savings [10-12]. Recent advances in AI, blockchain, and cloud infrastructure further lower the barrier for secure, scalable

micro-finance apps [13, 14].

3. Methodology

A Rapid Application Development (RAD) approach was chosen for faster prototyping and iterative user feedback. Planning, analysis, three incremental builds, and continuous testing characterized the process. Purposive sampling involved 15 village-bank participants; data were gathered via interviews, questionnaires, and observation.

4. Requirements Analysis

4.1 User Requirements

Three roles—Group Member, Group Administrator, and System Administrator—define registration, savings, loans, collateral, notifications, and reporting needs.

4.2 System Requirements

Functional requirements include user authentication, group management, savings, loans, collateral, messaging, and analytics; non-functional requirements cover security, usability, performance, compatibility, reliability, and compliance.

4.3 Use-Case & DFD Summary

Key use cases: *Register Member*, *Apply for Loan*, *Make Deposit*, *Loan Approval*, and *System Setup*. A context DFD and Level-0 DFD illustrate data flows among mobile users, the Node.js API, the SQLite store, and SMS/payment gateways

5. System Design

5.1 Database Schema

Eight tables (Users, Groups, Memberships, Contributions, Loans, Repayments, Transactions, and Notifications) capture all core entities with foreign-key enforcement. Refer to "Fig 1"

Table 1: Core database entities

Core database entities			
Entity	**Primary Key**	**Key Attributes (examples)**	**Purpose / Notes**
Users	user_id (INT)	username, phone_number, email, password_hash, role	Stores member login credentials, contact details, and role (member/admin).
Groups	group_id (INT)	group_name, created_by, creation_date, total_balance	Defines each village-bank group, its rules (monthly contribution, status).
Memberships	membership_id (INT)	user_id (FK), group_id (FK), join_date, role, status	Links users to groups; records join date and in-group role (treasurer, etc.).
Contributions	contribution_id (INT)	user_id (FK), group_id (FK), amount, date, is_approved	Logs savings deposits; supports approval workflows.
Loans	loan_id (INT)	user_id (FK), group_id (FK), loan_amount, due_date, status, interest_rate, outstanding_balance	Holds loan details, rates, dates, and balances.
Repayments	repayment_id (INT)	loan_id (FK), user_id (FK), amount, date	Tracks instalment payments against each loan.
Transactions	transaction_id (INT)	user_id (FK), group_id (FK), transaction_type, amount, date	Unified log for all monetary events (deposits, loans, repayments).
Notifications	notification_id (INT)	user_id (FK), message, date, is_read	Stores in-app/SMS alerts (loan approvals, payment reminders).

5.2 Architecture

A three-tier client–server model separates presentation (React Native), business logic (Remix Run & Node.js), and data (SQLite/Firebase). HTTPS, JWT, and role-based access provide security as shown in "Figure 1".

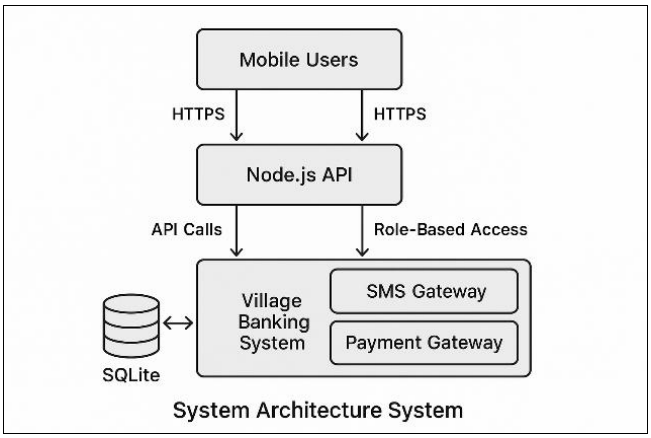

Fig 1: System Architecture

5.3 UI/UX Wireframes

Dashboards for Admin, Member, and Bank Officer emphasizes clarity, consistency, and accessibility, with quick actions for deposits, loans, and reports.

6. Implementation

Development used React Native with Expo CLI and Metro for live reload; Remix Run handled server routes; SQLite stored on-device data with optional Firebase sync. Key modules: Users, Loans, Deposits & Savings, Reports. Code is organized by feature folder with shared services and context providers.

7. Testing and Evaluation

Four test levels—unit (Jest), integration (Detox), system, and acceptance—validated functionality, security, and usability. Pilot users requested UI tweaks (button placement, colour contrast) that were subsequently applied.

8. Conclusion and Recommendations

Most objectives were met, delivering a functional, user-friendly system that digitizes village banking and introduces collateralized lending. Future work should add version control for transactions, expand automated reporting, and conduct broader pilots.

9. Acknowledgment
The author thanks God for guidance; *Chanda Chisanga* for sponsoring his education; *Dorothy Moomba* and *Geoffrey Zimba* for providing a supportive study environment; supervisors and mentors for expert advice; village-bank participants for invaluable feedback; and the Information and Communications University for academic support.

10. References

1. Bakar MA, Jailani N, Zarina S, Mohd YN. Developing financial management systems for community-based banking: Village banking software applications, Procedia Social and Behavioral Sciences, Dec 2011. n/a, pp. n/a [Online]. Available: https://doi.org/10.1016/j.sbspro.2011.05.039
2. Clifton C, Kaczmarczy LC, Mrozek M. Enhancing development efficiency in banking software: Using version control in application lifecycle management. In Proc. 38th SIGCSE Tech. Symp. Comput. Sci. Educ., Covington, KY, USA, 2007 p. n/a [Online]. Available: https://www.researchgate.net/publication/221536923
3. Connolly T, Begg C. Database Systems: A Practical Approach to Design, Implementation, and Management, 4th ed. Boston, MA, USA: Addison-Wesley, 2005. [Online]. Available: https://books.google.com/books?id=5V4hAQAAIAAJ
4. Dennis A, Wixom BH, Roth RM. Systems Analysis and Design, 4th ed. Hoboken, NJ, USA: Wiley, 2009 [Online]. Available: https://rads.wiley.com
5. Deepamala N, Shobha G. Effective project management in software development for financial applications. J. Technol. Sci. Educ., Jul 2018; 8(4). p. n/a [Online]. Available: https://doi.org/10.3926/jotse.427
6. Hassani H, *et al*. Managing village banking applications and their impact on rural communities. Educ. Sci., Dec 2018. vol. n/a, pp. n/a [Online]. Available: https://doi.org/10.3390/educsci8040210
7. Leung C-H, *et al*. Final-year project management in financial software systems development. Commun. Comput. Inf. Sci., Jan 2015. vol. n/a, pp. n/a [Online]. Available: https://doi.org/10.1007/978-3-662-46158-7_9
8. Lynch K, Heinze A, Scott E. International perspectives on financial software team projects in higher education. J. Inf. Technol. Educ. 2007; 6 pp. n/a [Online]. Available: https://doi.org/10.28945/3059
9. Mohamed M, *et al*. A case study on village banking software implementation at rural development institutes. World Appl. Sci. J. 2017; 35. pp. n/a [Online]. Available: https://wasj.org
10. Ozturk L. Financial inclusion and village banking: A theoretical perspective. SSRN Electron. J, Feb 2001. [Online]. Available: https://doi.org/10.2139/ssrn.1137541
11. Mapolisa T, Mafa O. Overcoming challenges in financial application development for open and distance learning programs. Int. J. Asian Social Sci. 2012. vol. n/a, pp. n/a [Online]. Available: https://www.researchgate.net/publication/302026709
12. Maserang S. Project management techniques for banking software development. Univ. Missouri-St. Louis, St. Louis, MO, USA, 2002 [Online]. Available: http://www.umsl.edu/~sauterv/analysis/488_f02_papers/ProjMgmt.html
13. Meta. React Native documentation, Apr 2025 [Online]. Available: https://reactnative.dev/docs/getting-started
14. Expo. Expo Router documentation, Apr 2025 [Online]. Available: https://expo.github.io/router/docs
15. Remix Software Inc. Remix Run documentation, Apr 2025 [Online]. Available: https://remix.run/docs