



# **International Journal of Advanced Research in Education and Technology (IJARETY)**

**Volume 13, Issue 3, May - June 2026**

**Impact Factor: 9.971**



# FinTrac: Design and Implementation of an Offline-First Mobile Personal Finance Management System with AI-Driven Insights

Rohit T Nadageri

Student, Department of Master of Computer Applications, CMR Institute of Technology, Bengaluru, India

**ABSTRACT:** FinTrac is an offline-first cross-platform mobile application for personal finance management designed for the Indian context. Built on React Native with AsyncStorage for on-device persistence, the system supports transaction tracking, budget management, savings goals, and AI-driven insights via the Gemini API—entirely without mandatory internet connectivity. A three-tier architecture ensures modularity and testability. Functional testing across 16 structured test cases yielded a 100% pass rate, validating correctness and robustness.

**KEYWORDS:** Offline-first architecture, React Native, personal finance management, AI financial advisor, AsyncStorage

## I. INTRODUCTION

India's Unified Payments Interface processed over 10 billion transactions per month as of 2024, yet personal finance tracking applications have failed to keep pace with this complexity [1]. Market-leading solutions such as Mint, Walnut, MoneyView, and ET Money introduce significant limitations: mandatory cloud connectivity renders them inaccessible in low-bandwidth environments; centralised data storage raises privacy concerns; subscription-based features exclude economically constrained users; and interfaces calibrated for high financial literacy create adoption barriers among average Indian smartphone users [2].

FinTrac fills this gap. The central hypothesis is that an offline-first, privacy-by-design mobile application—localised for the Indian financial context and enriched with AI-driven advisory capabilities—can deliver superior usability and trust compared to cloud-dependent alternatives without requiring paid infrastructure. Key contributions include: (i) design of an offline-first mobile financial management architecture using React Native and AsyncStorage; (ii) comparative evaluation of frameworks, storage technologies, and AI models to justify technology selection; (iii) integration of Gemini AI for personalised spending insights in a resource-constrained mobile context; (iv) full UML-documented system design; and (v) structured software testing across 16 test cases demonstrating functional correctness.

## II. LITERATURE SURVEY

Kapoor et al. [3] surveyed mobile money management applications and identified internet dependency as the primary barrier to adoption in semi-urban Indian communities, finding that 38% of users abandoned finance apps within the first week due to connectivity requirements—directly motivating FinTrac's offline-first design. Agrawal and Singh [4] demonstrated that natural language financial advisors improve budget adherence by 22% compared to rule-based systems, though their prototype required server-side inference, creating latency and privacy concerns. FinTrac addresses this by routing AI queries through the Gemini API only on explicit user demand, keeping all financial data on-device during normal operation. Patel et al. [5] found that AsyncStorage scales to approximately 50,000 records before noticeable degradation—a threshold sufficient for several years of daily transaction logging—directly informing FinTrac's storage design. Existing literature focuses on either purely local storage without AI advisory, or cloud-native AI without privacy guarantees. FinTrac bridges this gap by combining both capabilities in a single coherent architecture.

## III. METHODOLOGY / APPROACH

### A. System Architecture & Technology Selection

FinTrac adopts a three-tier architecture separating the Presentation Layer, Application Logic Layer, and Data Layer. This modularity ensures that UI changes do not propagate to storage logic. The architecture was selected over monolithic and client-server alternatives based on maintainability and offline capability. React Native was chosen as the frontend

framework for its JavaScript ecosystem compatibility, extensive library support for financial UI components, and proven deployment record in fintech production. For AI advisory, Gemini 1.5 Flash was selected because its free-tier API quota supports expected usage (under 200 queries/user/month), its 1-million-token context window permits embedding months of transaction history, and its response latency under 2 seconds satisfies mobile UX requirements. AsyncStorage was selected for on-device persistence due to its native React Native integration and zero external dependency.

**Table I: Technology Selection Summary**

| Component    | Selection           | Key Rationale                                    |
|--------------|---------------------|--------------------------------------------------|
| Architecture | Three-Tier          | Modular, testable, supports offline-first        |
| Frontend     | React Native + Expo | Cross-platform, large ecosystem, fintech-proven  |
| AI Model     | Gemini 1.5 Flash    | Free tier; high accuracy; <2 s latency           |
| Storage      | AsyncStorage        | Native to React Native; zero external dependency |

**B. Database Design**

FinTrac stores all financial data on-device using a document-oriented key-value model across five collections: User (userId, name, email, passwordHash, dateOfSignup); Account (accountId, userId, accountName, accountType, balance); Transaction (transactionId, accountId, type, category, amount, description, transactionDate); Budget (budgetId, userId, category, budgetLimit, month, year); and Saving Goal (goalId, goalName, targetAmount, savedAmount, targetDate). UUID v4 is used for all primary keys, eliminating collision risk in offline-generated records.

**C. Core Functional Modules**

Six core modules compose the system: (i) User Authentication — guest access, full registration, and credential-based login with locally stored session tokens; (ii) Transaction Management — full CRUD operations with mandatory field validation; (iii) Budget Management — monthly budgets per spending category with colour-coded progress bars (green < 80%, amber 80–100%, red > 100%) and overspend alerts; (iv) Savings Goal Management — named goals with target amounts, deadline dates, contribution tracking, and a celebratory achievement UI; (v) AI Financial Advisor — Gemini 1.5 Flash invoked on explicit user request with summarised transaction and budget context to return personalised insights; (vi) Financial Reports — income-versus-expense summaries, category breakdowns, and monthly trend visualisations rendered entirely from local data.

**IV. RESULTS & DISCUSSION****A. Testing Results**

FinTrac was validated using Black-Box and White-Box testing across four levels: Unit Testing (individual controllers), Integration Testing (cross-module data flows), System Testing (end-to-end user journeys), and User Acceptance Testing. The testing environment used Android Emulator API 33 and a physical Samsung Galaxy A53 (Android 13).

**Table II: Complete Test Case Results (All 16 PASS)**

| Test ID   | Description                   | Expected Result                  | Status |
|-----------|-------------------------------|----------------------------------|--------|
| TM01–TM04 | Transaction CRUD & validation | Save/delete/validate correctly   | PASS   |
| BM01–BM04 | Budget create/edit/validate   | Budget saved, updated, validated | PASS   |
| SG01–SG04 | Savings goal & contributions  | Goal created; progress updated   | PASS   |
| FR01–FR04 | Report generation & filters   | Charts rendered; filters applied | PASS   |

All 16 test cases achieved a PASS result (100% pass rate), validating the functional correctness and robustness of all core modules. Two minor performance defects were identified and resolved: report generation delays exceeding 3 seconds on datasets over 2,000 records were resolved via lazy loading and memoised aggregation (reducing latency to under 800 ms); and budget display recomputation requiring a manual restart was resolved by subscribing budget components to the FinanceContext transaction array.

**B. Non-Functional Requirements**

| Attribute   | Specification                                                                                |
|-------------|----------------------------------------------------------------------------------------------|
| Performance | App launch < 3 s; transaction save < 500 ms; dashboard refresh < 1 s ( $\geq 3$ GB RAM)      |
| Security    | bcrypt password hashing; session tokens purged on logout; no financial data in external logs |
| Usability   | Core tasks within 3 screen interactions; SUS target $\geq 75$                                |
| Reliability | Zero data loss under graceful app closure; error messages for all invalid states             |
| Portability | Android 9.0+, iOS 13+, Chromium-based web browsers via Expo Web                              |
| Scalability | Up to 50,000 transaction records per user without performance degradation [5]                |

**C. Comparative Analysis****Table III: FinTrac vs. Competing Applications**

| Feature              | FinTrac | Mint | Walnut  | ET Money |
|----------------------|---------|------|---------|----------|
| Offline operation    | Full    | None | Partial | None     |
| INR / UPI support    | Native  | No   | Yes     | Yes      |
| AI financial advisor | Yes     | No   | No      | Partial  |
| Data on-device only  | Yes     | No   | No      | No       |
| Free (all features)  | Yes     | No   | Yes     | No       |

FinTrac uniquely combines full offline operation with native Indian financial localisation, an AI advisory layer, and complete on-device data sovereignty. No evaluated competitor achieved all four properties simultaneously.

**V. CONCLUSION**

FinTrac demonstrates that a production-quality financial application can be engineered entirely with free, open-source tooling at zero infrastructure cost. Through systematic comparative evaluation of architectural patterns, frontend frameworks, storage technologies, and AI models, all design decisions were grounded in objective trade-off analysis. The 100% pass rate across 16 structured test cases validates functional correctness. The three-tier architecture provides a foundation for incremental enhancement without architectural rewrites.

Future work includes opt-in cloud sync via Firebase or Supabase with end-to-end encryption; bank API integration using India's Account Aggregator framework; multi-currency support; PDF/CSV export for tax filing; predictive budgeting using time-series analysis; and an admin dashboard for institutional deployments. For the 600+ million smartphone users in India navigating increasingly complex digital payment ecosystems, FinTrac represents a meaningful step toward democratised, privacy-preserving financial self-management.

**REFERENCES**

- [1] National Payments Corporation of India, "UPI Monthly Statistics," NPCI Report, vol. 12, 2024.
- [2] M. Gupta and R. Sharma, "Barriers to mobile financial app adoption in India," J. Fin. Technology, vol. 8, no. 2, pp. 45–62, 2023.
- [3] A. Kapoor, S. Mehta, and D. Iyer, "Internet dependency as an adoption barrier for mobile money management tools in semi-urban India," Int. Conf. Digital Inclusion, Mumbai, 2023, pp. 112–119.
- [4] V. Agrawal and P. Singh, "AI-powered personal finance advisors: Impact on budget adherence in young Indian adults," IEEE Trans. Human-Mach. Syst., vol. 14, no. 3, pp. 201–215, 2024.

- [5] S. Patel, H. Joshi, and N. Rao, "AsyncStorage performance benchmarks in React Native applications," Proc. ACM MobiSys Workshop, 2023, pp. 77–84.
- [6] E. Gamma et al., Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.
- [7] I. Sommerville, Software Engineering, 10th ed. Pearson, 2016.
- [8] R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 8th ed. McGraw-Hill, 2015.
- [9] React Native Documentation. [Online]. Available: <https://reactnative.dev/>
- [10] Google DeepMind, "Gemini 1.5: Unlocking multimodal understanding," Technical Report, 2024.



## International Journal of Advanced Research in Education and Technology

**ISSN: 2394-2975**

**Impact Factor: 9.971**