“Software Requirement Specification”

By
MUHAMMAD AYAAD (2023-KIU-BS4691) (Individual Submission)

Type: FinTrack Personal Expense Tracker (SRS)

Submitted to: Dr. Muazzam Ali Kazmi

Section: Software Engineering (A)

Subject: Software Requirements Engineering

Submission Date: 25/11/2025

DEPARTMENT OF SOFTWARE ENGINEERING (SCET) KIU GILGIT

Contents

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) describes the core requirements for **FinTrack**, a personal expense-tracking application. It outlines the functional and non-functional requirements that will guide the design, development, testing, and validation of the system throughout its lifecycle.

1.2 Project & SRS Repository Link

Github Repo Link (Project + SRS):
https://github.com/Ayaad72/FinTrack.git

1.3 Scope

FinTrack is a cross-platform application developed using Flutter, targeting both **Android** and **Web** platforms. The application enables users to record their daily income and expenses, categorize their spending, review monthly financial summaries, and visualize financial patterns using charts.
The **MVP** includes:

- User authentication
- Transaction management (Add/Edit/Delete)
- Categories (predefined + custom)
- Dashboard and visualization
- Basic export/backup capabilities

1.4 Definitions, Acronyms, and Abbreviations

- **FinTrack** — Personal Expense Tracker Application
- **MVP** — Minimum Viable Product
- **FR** — Functional Requirement
- **NFR** — Non-Functional Requirement
- **DB** — Database (SQLite locally)
- **UI** — User Interface

---

2. Overall Description

2.1 Product Perspective

FinTrack is a standalone application with the following ecosystem components:

- A Flutter-based mobile and web client
- Local storage using SQLite
- Optional cloud synchronization using Firebase
- Visualizations powered by the **fl_chart** library
- Optional future admin dashboard for analytics

2.2 Product Functions

Core functionalities include:

- User account creation, login, and profile management
- Create, edit, and delete transactions (income/expense)
- Categorize transactions using predefined or custom categories
- Monthly summaries and financial overviews
- Visual charts (bar & pie) with filters
- Data export (CSV) and local/optional cloud backup

2.3 User Classes and Characteristics

- **End Users:** Regular users with basic mobile literacy who want to track personal finances.
- **Admin (Future Expansion):** Access for analytics or data review.
- **Testers/Developers:** Users supporting QA and maintenance.

2.4 Operating Environment

- **Mobile:** Android 8.0 and above (iOS planned in future)
- **Web:** Chrome, Edge (Flutter Web supported browsers)
- **Libraries/Tools:** fl_chart, sqflite, shared_preferences, firebase_auth
- **Connectivity:** Works fully offline; sync requires internet

---

3. Specific Requirements

3.1 Functional Requirements (FR)

FR-001 — User Registration

- Users can sign up using email/phone, password or OTP.
- Inputs include: Name, email/phone, password, currency preference.
- **Priority:** High
- **Output:** New verified user account.

FR-002 — User Login

- Secure login using stored credentials.
- Supports “Remember Me.”
- **Priority:** High

FR-003 — Profile Management

- Users can view and update profile details such as name and currency.
- **Priority:** Medium

FR-010 — Add Transaction

- Users can add new transactions with fields: title, amount, date, type (income/expense), category, note.
- **Priority:** High
- **Output:** Transaction saved locally.

FR-011 — Edit Transaction

- Users may update existing transaction details.
- **Priority:** Medium

FR-012 — Delete Transaction

- Users can delete a transaction after confirmation.
- **Priority:** Medium

FR-013 — Transaction History & Search

- Users can view past transactions and filter/sort by date, category, or amount.
- **Priority:** Medium

FR-020 — Predefined Categories

- The system includes built-in categories like Food, Grocery, Travel, Shopping.
- **Priority:** High

FR-021 — Custom Categories

- Users can create their own categories.
- **Priority:** Medium

FR-030 — Monthly Summary

- Displays total income, total expenses, and net balance.
- **Priority:** High

FR-031 — Category-wise Breakdown

- Shows expense distribution by category, including percentages.
- **Priority:** High

FR-040 — Visualizations (Charts)

- Bar charts for monthly totals.
- Pie charts for category distribution.
- **Priority:** Medium

FR-050 — Export & Backup

- Export financial data to CSV.
- Local backup support with optional cloud backup.
- **Priority:** High

FR-060 — Offline Support & Sync

- App remains fully functional offline.
- Synchronization happens automatically when online.
- **Priority:** Medium

FR-070 — Settings & Preferences

- Users can manage preferences like currency, app notifications, and start-of-month settings.
- **Priority:** Medium

3.2 Non-Functional Requirements (NFR)

NFR-001 — Usability

- The application must be easy to use with a clean and intuitive UI.

NFR-002 — Performance

- Dashboard should load within **2–3 seconds** for up to 1000 transactions.

NFR-003 — Reliability & Data Integrity

- All transactions must be saved atomically, ensuring no partial or corrupted entries.

NFR-004 — Security

- Passwords must be securely hashed.
- Authentication tokens stored using device secure storage (Keychain/Keystore).

NFR-005 — Compatibility

- Must run on Android devices and supported modern browsers.

NFR-006 — Localization

- English must be supported.
- Urdu localization may be added in future.

---

4. System Features

4.1 Transaction Flow (Add/Edit/Delete)

1. User opens the **Add Transaction** screen.
2. Inputs details such as title, amount, date, type, category, note.
3. App validates inputs and saves data locally.
4. If cloud sync is enabled & internet is available, data is uploaded.
5. When editing or deleting a transaction, the dashboard auto-updates.

4.2 Dashboard & Visualization Flow

1. Dashboard aggregates all transactions for the current month.
2. Displays total income, expenses, and net balance.
3. Shows category-wise breakdown and percentage distribution.
4. Charts reflect this data; selecting a category filters the transaction list accordingly.

---

5. External Interface Requirements

5.1 User Interfaces

- Simple login and sign-up screens
- Dashboard (summary + charts)
- Transaction list with filters
- Add/Edit transaction forms
- Settings (currency, backups, preferences)

5.2 Hardware Interfaces

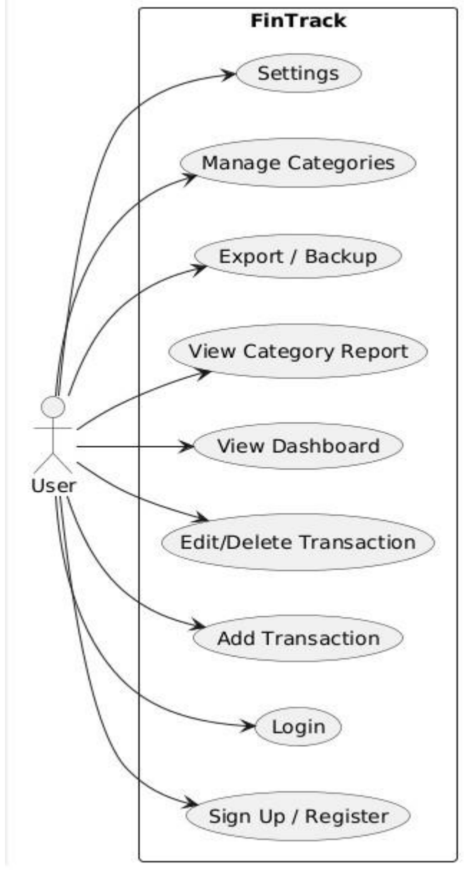
- No hardware required
- Optional: Camera for receipt uploads
- Local device storage for DB

5.3 Software Interfaces

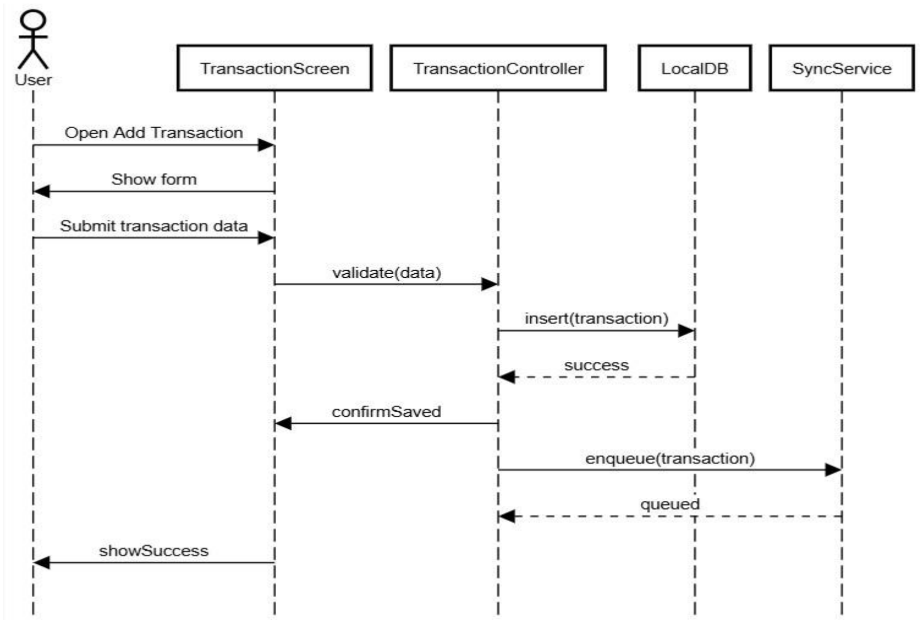
- fl_chart for visualizations
- sqflite or Firebase for storage
- CSV export library
- Optional payment APIs (future wallet features)

5. **UML Diagrams**

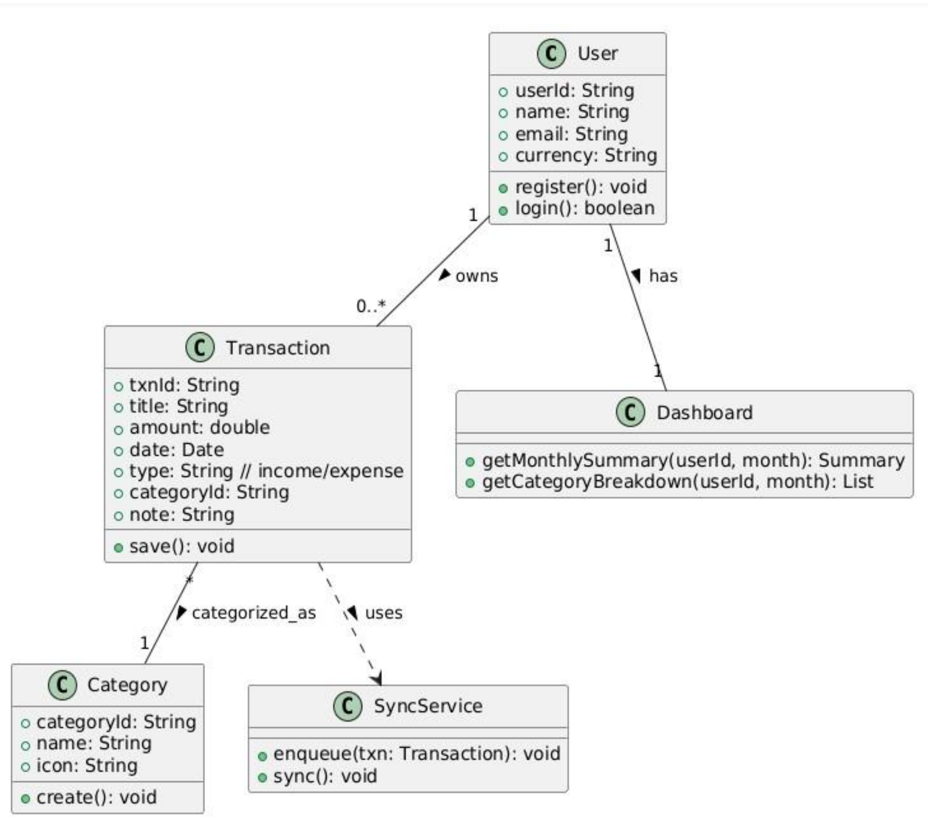
6.1 Use Case Diagram



6.2 Sequence Diagram — Add Transaction



6.3 Class Diagram (Core Entities)



7. Appendices

7.1 Assumptions and Dependencies

- Users must have basic smartphone knowledge.
- Internet is only required for sync and cloud features.
- Certain features depend on third-party libraries (Firebase, fl_chart, sqflite).

7.2 Acceptance Criteria

- Users can register, log in, and manage their transactions reliably.
- Dashboards load within performance limits.
- Charts and summaries accurately reflect stored data.

7.3 Future Enhancements

- iOS support
- AI-based spending insights
- Receipt OCR scanning
- Multi-currency support
- Admin dashboards and analytics