usiness Requirements Document (BRD)

FinSight

Maryam Ali

Dania Osama Bahjat

1. Executive Summary

FinSight is a Financial Transaction Tracker designed to help users monitor transactions near real-time, detect and manage potentially fraudulent activity via rule-based risk scoring, and gain actionable spending insights through dashboards. It also supports subscription detection with due-soon notifications and seeds realistic demo transactions for first-time users so the product is immediately explorable without needing real bank integrations.

2. Business Problem Statement

Individuals and teams who want to track spending and detect suspicious activity often lack:

1. fast visibility into new transactions without manual refresh,
2. clear, explainable fraud indicators,
3. a simple way to categorize/filter transactions and spot patterns,
4. and proactive identification of recurring monthly subscriptions.
5. FinSight addresses this by centralizing transaction tracking, applying deterministic rule-based fraud scoring (0–100 with clear risk levels), surfacing fraud alerts to review/resolve, and providing dashboards and subscription notifications.

3. Business Goals and Objectives

3.1 Goals

1. Provide a single platform to track transactions and analyze spending patterns with dashboards.
2. Detect suspicious transactions automatically using explainable, rule-based scoring and alerts.
3. Improve user awareness of recurring financial commitments via subscription detection and due-soon notifications.

3.2 Objectives

1. Allow manual transaction entry and ensure every transaction is assessed for fraud risk and logged for compliance.
2. Provide comprehensive filtering/sorting/pagination to quickly find transactions and support polling efficiency.
3. Offer a dashboard summary (income, expenses, balance, fraud metrics, category breakdowns, trends).

4. Scope

4.1 In Scope

- Manual transaction creation with validation, fraud scoring, and audit logging.
- Transaction filtering and sorting (type/category/date range/fraudulent flag; sorting by date/amount/fraud score/category; pagination).
- Near real-time viewing via frontend polling toggle (5–10 seconds).
- Rule-based fraud detection, risk levels, and fraud alert creation.
- Fraud alert review and resolution (alerts preserved; filterable by status/severity).
- Insights dashboard (income/expenses/balance and aggregations).
- Subscription detection (25–35 day recurring pattern), ignore capability, due-soon queries, and UI notifications.
- Audit logging (immutability and compliance support).

4.2 Out of Scope

- Machine learning fraud models (explicitly deferred).
- Real-time streaming ingestion (true event streaming).
- External payment gateway/bank integrations and multi-currency.
- Native mobile apps.

5. Stakeholders

5.1 Primary Stakeholders

End Users (Consumers): Track spending, add manual transactions, filter activity, review fraud alerts, manage subscriptions, view dashboards.

5.2 Administrative / Compliance Stakeholders

Compliance Officer / Auditor: Requires immutable audit trails of actions for regulatory/compliance needs.

5.3 Secondary Stakeholders

Developers / Maintainers: Need backward compatibility, predictable error handling, and testability of rules and filters.

6. Business Requirements

ID	Requirement	Description

BR-1	Immediate usability	New users must see realistic demo transactions on first login when no data exists. requirements
BR-2	Transaction capture	Users must be able to add transactions manually and store them reliably. requirements
BR-3	Fraud risk visibility	Transactions must be analyzed by rule-based fraud detection, scored 0–100, categorized into risk levels, and flagged when high risk. requirements
BR-4	Fraud workflow	System must generate fraud alerts for flagged transactions and support review + resolution without deleting alert history. requirements
BR-5	Insights	System must provide dashboards summarizing financial position and spending patterns. requirements
BR-6	Subscription control	System must detect recurring subscriptions and notify users about due-soon payments; users can ignore unwanted detections. requirements
BR-7	Compliance traceability	System must keep an audit trail of key user/system actions to support compliance monitoring. design
BR-8	Near real-time monitoring	Users must be able to see updates automatically via a live-refresh polling mechanism. requirements

7. Functional Requirements (Business View)

ID	Functional Requirement	Description
FR-1	Demo data generation	On login, if user has 0 transactions, create 25–50 demo transactions spread across 60–90 days; include categories (groceries/utilities/entertainment/salary/rent), income+expense, and at least 3 fraud-triggering examples; deterministic by user.
FR-2	Manual transaction creation	User can create a transaction; system validates required fields, computes fraud score, flags if high, creates alert if needed, and logs action.
FR-3	Transaction filtering/sorting	Filter by type/category/date range/fraudulent flag; sort by date/amount/fraud score/category; pagination supported; filters combinable.
FR-4	Near real-time updates	Frontend offers a live refresh toggle; when enabled polls transactions every 5–10 seconds and updates display.
FR-5	Fraud detection scoring	System computes fraud score 0–100 and maps risk levels: LOW (<40), MEDIUM (40–69), HIGH (≥70).
FR-6	Fraud alerts	For fraudScore ≥ 70, mark transaction as fraudulent and create a FraudAlert with severity matching risk level.
FR-7	Fraud alert management	User can list alerts (most recent first), filter by resolved/severity, and resolve alerts (system logs resolution); alerts are never deleted.
FR-8	Dashboard summary	Dashboard returns total income, total expenses, and current balance (income − expenses), plus additional insights as defined.
FR-9	Subscription detection	Identify same-merchant payments occurring 25–35 days apart, create subscription record after 2+ qualifying payments, compute avg amount and next due date, allow ignore, provide due-soon query.

8. Non-Functional Requirements

1. **Compatibility:** Enhancement must not break existing endpoints/contracts.

2. **Performance:** Support efficient pagination and polling to minimize transfer during live refresh.
3. **Maintainability/Testability:** Layered architecture and strong automated test coverage for critical paths (filters, fraud rules, alerts, subscriptions).
4. **Security & Compliance posture:** Audit logging is required for key actions supporting regulatory compliance.

9. High-Level Solution Description

9.1 Solution Architecture

FinSight is implemented as a React frontend + Spring Boot backend, following a layered architecture (Controllers → Services → Repositories) and interacting via HTTP/REST.

Technology Stack (High-level):

- Backend: Spring Boot (Java 17)
- Frontend: React
- DB (dev): H2 in-memory
- Deployment: Docker Compose

9.2 Key Data Objects (Business View)

- **User**: account owner of transactions (concept referenced as core entity).
- **Transaction**: income/expense financial record with category/date/amount and fraud metadata.
- **FraudAlert**: immutable record of suspicious activity that can be reviewed/resolved.
- **Subscription**: detected recurring monthly payment pattern with due dates and ignore status.
- **AuditLog**: compliance record of actions (creation, seeding, resolving, ignoring).

10. Acceptance Criteria (High-Level)

AC-1: First-time user demo readiness
Given a user logs in with zero transactions, the system generates 25–50 demo transactions across the previous 60–90 days, with varied categories and both income and expense types, including examples that trigger fraud detection.

AC-2: Manual transaction → fraud scoring → alert
Given a user creates a valid manual transaction, the system persists it, computes fraud score, flags it if score ≥ 70, creates an alert when flagged, and records the action.

AC-3: Filtering + sorting correctness
Given filters (type/category/date range/fraudulent) and sorting parameters are provided, the returned transactions satisfy all filters simultaneously and are ordered as requested, with pagination metadata.

AC-4: Live refresh behavior
When live refresh is enabled, the UI polls every 5–10 seconds and updates the list when new data is detected; when disabled, polling stops.

AC-5: Subscription due-soon notification basis
Given subscriptions are detected from 25–35 day recurring patterns, the system returns subscriptions due within a specified number of days, excluding ignored subscriptions.

11. Constraints and Assumptions

11.1 Constraints

- No external bank transaction ingestion in this phase; demo seeding + manual entry are required to make the system usable immediately.
- Fraud detection is rule-based (not ML).

11.2 Assumptions

- Users will rely on filtering and dashboard insights to analyze spending patterns.
- A polling-based “near real-time” UX is acceptable for this build.
- requirements

12. Risks and Mitigation

Risk	Impact	Mitigation
False positives from rules	User distrust in fraud alerts	Keep scoring explainable (risk levels) and allow alert resolution workflow.

Demo data feels unrealistic	Poor first impression	Use category variety, realistic ranges, and deterministic generation for consistent demos.
Polling causes unnecessary load	Performance degradation	Use pagination + efficient queries and allow users to toggle live refresh.

13. Success Metrics

FinSight is successful if:

1. New users can explore the product immediately after login due to automatic demo seeding.
2. Fraud detection produces consistent scoring/risk levels and creates alerts for high-risk transactions.
3. Users can efficiently filter/sort transactions and monitor updates with live refresh.
4. Subscription detection identifies recurring payments and supports due-soon awareness.