📘 Product Requirements Document (PRD)

LedgerOne — Unified Financial Tracking & Insight App

---

1. 🧭 Product Overview

LedgerOne is a mobile-first financial assistant that automatically tracks, verifies, and visualizes user transactions across multiple Ethiopian banks and wallets (CBE, Telebirr, Awash, BOA, etc.) using **SMS alerts, app notifications, and receipt links**.

It gives users a **unified financial dashboard**, spending insights, and verified transaction history without requiring bank APIs.

---

2. 🎯 Product Vision

> Enable people in emerging markets to **understand, control, and grow their money** without needing open banking infrastructure.

---

3. 👤 Target Users

Primary Users

- Young professionals in Addis Ababa
- Freelancers receiving payments across multiple banks
- Small business owners

- Remote workers (like you 👀)

Secondary Users

- Students managing allowance
- Families tracking shared expenses

---

4. ❗ Problem Statement

Users in Ethiopia:

- Use **multiple financial apps**
- Have **no unified transaction view**
- Rely on **SMS as proof of payment**
- Cannot easily track spending or savings

Existing banks:

- Do not provide unified dashboards
- Do not offer deep analytics
- Do not provide cross-bank visibility

---

5. 💡 Solution

LedgerOne:

- Detects transactions from **SMS + Notifications**

- Verifies transactions using **bank receipt links**
- Categorizes spending automatically
- Visualizes finances in one place
- Provides insights and alerts

---

6. 🧩 Core Features

6.1 Transaction Ingestion Engine

Sources

1. SMS parsing
2. Notification parsing
3. Receipt link verification
4. Manual entry / OCR (future)

Output

```
{
  amount,
  type: debit|credit,
  bank,
  timestamp,
  txId,
  verificationLevel
}
```

6.2 Receipt Verification System

- Extract receipt URLs from SMS/notifications
- Validate domain (CBE, BOA, Awash, etc.)
- Fetch and parse receipt data
- Cross-check with SMS data
- Assign verification level

Verification Levels

Level	Meaning
Verified	Confirmed by bank receipt
Detected	Parsed from SMS/Notification
Manual	User entered
Flagged	Mismatch

6.3 Unified Dashboard

Elements

- Total balance across banks
- Monthly inflow vs outflow
- Net savings
- Top spending categories
- Recent transactions list

6.4 Transaction History Screen

Each transaction shows:

- Amount
- Date/time
- Bank source
- Category
- Verification badge
- Receipt view (if available)

6.5 Spending Categorization

Auto-classify into:

- Food
- Transport
- Bills
- Transfers
- Income
- Airtime
- Shopping

User can edit → improves ML rules later.

6.6 Analytics & Insights

- Weekly/monthly spending trends
- Category breakdown (pie chart)
- Balance trend line
- Alerts:
    - “You spent 25% more on food this week”
    - “Your balance is decreasing faster than usual”

6.7 Budgeting (Phase 2)

- Set monthly limits per category
- Progress bars
- Overspend alerts

6.8 Privacy & Security System

- All parsing done **on-device**
- No SMS content uploaded
- Only structured transaction data stored
- Optional PIN / biometric lock
- Data export & delete options

---

7. 🔄 User Flows

7.1 Onboarding Flow

1. Welcome screen
2. Value explanation
3. Permission requests:
    - Notification access (first)
    - SMS access (optional upgrade)
4. First transactions auto-imported
5. Dashboard displayed

7.2 SMS Transaction Flow

```
SMS received
→ SMS Receiver triggered
→ Parse amount, bank, date
→ Extract receipt link (if exists)
→ Store as Detected transaction
→ Show in dashboard
```

7.3 Receipt Verification Flow

```
User taps “Verify”
→ Fetch receipt URL
→ Parse receipt data
→ Cross-check with SMS
→ Mark transaction Verified
```

7.4 Notification Flow

```
Bank app sends push notification
→ Notification listener triggered
→ Extract transaction data
→ Store transaction
```

---

8. 🎨 UI/UX Design Requirements

Inspired by modern fintech apps like
Rocker and
Rocket Money

Design Principles

- Clean, minimal, high-trust
- Financial colors (green/red/neutral)
- Card-based layout
- Clear typography hierarchy
- Data-first design

Key Screens

1. Dashboard
2. Transaction list
3. Transaction detail
4. Insights & charts
5. Settings & permissions

---

9. 🧱 Technical Architecture

9.1 Mobile App (Frontend)

- **React Native (Expo)** OR Kotlin native
- State management: Zustand / Redux
- Charts: Victory / Recharts / ECharts

9.2 Parsing Layer

Modules:

- SMS Parser
- Notification Parser
- Receipt Parser

Use:

- Regex
- Bank-specific adapters

9.3 Local Storage

- SQLite / Room DB
- Encrypted storage for sensitive data

9.4 Backend (Optional Phase 2)

- Spring Boot (you already use this 👍)
- User accounts
- Sync across devices
- AI insights engine

---

10. 📱 Permissions Strategy

The permissions model must follow **progressive disclosure**:

> Ask for the **least sensitive permission first**, show value, then request more sensitive permissions only when needed.

📋 Permission Matrix

Permission	Purpose	When Requested	Required for Core?	Fallback if Denied
Notification Access	Detect transactions from bank app notifications	On onboarding (Step 1)	✅ Yes (primary ingestion)	User can enable SMS or manual input
READ_SMS	Full SMS parsing + receipt link extraction	After value is demonstrated (Step 2)	⚠️ Optional but recommended	Notifications + manual receipt input
Internet	Fetch receipt pages for verification	Only when user taps “Verify receipt”	❌ Not required for base tracking	Verification disabled

RECEIVE_SMS (implicit with READ_SMS)	Listen for new SMS only (BroadcastReceiver)	Same time as READ_SMS	⚠️ Required if SMS enabled	N/A
POST_NOTIFICATIONS (Android 13+)	Allow LedgerOne to send alerts (e.g., insights, reminders)	After onboarding or first insight	❌ Optional	No alerts
Biometric / Device Credentials	App lock security	When user enables App Lock	❌ Optional	Use PIN or no lock
Storage / Media (if OCR later)	Import screenshot/PDF receipts	When user uses “Import receipt image”	❌ Optional	Manual entry only

---

🧭 Permission Request Flow (UX)

Step 1 — Onboarding (Low friction)

Ask: Notification Access
Message:

> “LedgerOne can automatically track your transactions from your bank app notifications.”

Buttons:

- **Allow**
- Not now (continue with manual mode)

Step 2 — After first value is shown

Trigger after:

- First transactions detected OR

- User opens dashboard

Ask: SMS Access

Message:

> “Enable SMS access to capture full transaction details and verify receipts automatically. LedgerOne only reads bank transaction messages — not personal messages or OTPs.”

Buttons:

- **Enable SMS Access**
- Skip for now

Step 3 — On demand (contextual)

Ask: Internet permission (implicit) when user taps **“Verify receipt”**

Message:

> “LedgerOne will securely fetch the bank receipt to verify this transaction. Data stays on your device.”

Step 4 — Optional enhancements

Later prompts:

- Notifications for insights
- App lock for privacy
- Import receipt images

🔒 Privacy Safeguards (Must be implemented)

SMS handling rules

- Read **only new incoming SMS**
- Filter by known bank senders
- Ignore OTP patterns (regex)
- **Do not store raw SMS text**
- Immediately extract → then discard

Notification handling rules

- Read only notification text
- Process only bank-related notifications
- Discard non-financial notifications

Receipt handling rules

- Fetch receipt **on-device**
- Parse required fields only
- Do not store HTML
- Do not send to backend

📱 Play Store Compliance Positioning

In Play Console permission declaration:

SMS Access justification

> “LedgerOne automatically records financial transactions from bank SMS alerts to help users track income and expenses across multiple banks. The app only processes bank transaction messages, ignores personal messages and OTPs, and does not store or transmit SMS content.”

Notification access justification

> “LedgerOne reads bank app notifications to detect transaction events and update the user’s financial dashboard in real time.”

🧠 Trust-building UX elements

To increase permission acceptance:

- Show **sample dashboard** before asking SMS
- Display **“No data leaves your phone”** badge
- Add **“Turn off anytime”** toggle in Settings
- Show **banks supported list** (CBE, Telebirr, Awash, BOA)

⚠️ Failure / Denial Handling

If user denies permissions:

Case: Notification denied

- Show:

  > “Enable notifications to auto-detect transactions”

- Offer:
  - Enable SMS

- Manual add

Case: SMS denied

- Continue with notification parsing
- Allow manual receipt verification

Case: Internet denied

- Mark receipts as:

  “Verification unavailable”

11. Monetization Strategy

LedgerOne should follow a **freemium + premium analytics model**.

🎯 Free Tier (Core Value)

- SMS / Notification transaction detection
- Basic transaction list
- Basic categories
- Balance summary
- Monthly totals

This gives immediate value → user adoption.

💎 Premium Tier (Subscription)

Price target (Ethiopia market)

- 49 – 99 ETB/month
- or 499 – 799 ETB/year

Premium features

📊 Advanced Analytics

- Spending trends over time
- Category comparisons
- Monthly reports

🔮 Smart Forecasting

- Balance prediction
- Upcoming bill alerts

📅 Recurring Transaction Detection

- Salary detection
- Rent detection
- Subscriptions

🧾 Receipt Vault

- Verified receipt storage
- Export (PDF/CSV)

📈 **Financial Health Score**

- Savings rate
- Expense ratios
- Risk indicators

🏢 Future B2B Monetization

Later you can add:

- SME accounting lite
- Freelancer income tracking
- Tax/export reports

---

12. Key Metrics (What success looks like)

📊 Activation Metrics

- % users granting Notification access
- % users granting SMS access
- Time to first transaction detected

📈 Engagement Metrics

- Daily Active Users (DAU)

- Weekly Active Users (WAU)
- Sessions per day
- Transactions viewed per session

💰 Revenue Metrics

- Free → Premium conversion rate
- Monthly recurring revenue (MRR)
- Customer lifetime value (LTV)

🎯 Retention Metrics

- Day 1 retention
- Day 7 retention
- Day 30 retention

Goal:

D30 retention > 25%

---

13. Privacy & Compliance (CRITICAL for approval)

🔐 Data Principles

LedgerOne must be:

- **On-device first**
- **Zero SMS storage**
- **No cloud sync by default**
- **No third-party sharing**

📜 Permissions Policy

SMS

Used only for:

- Transaction detection

Never for:

- OTP
- Personal messages

Notifications

Used only for:

- Bank app alerts

🧾 Data Stored

Allowed:

```
amount
timestamp
bank_name
transaction_id
category
verification_status
```

Not allowed:

```
full_sms_text
receipt_html
account_number_full
```

🔏 Security Measures

- Local database encryption (SQLCipher or Android Encrypted DB)
- Biometric lock (optional)
- App PIN lock

🧾 Privacy Policy must clearly say

> “LedgerOne processes financial SMS and bank notifications locally on your device.
> No personal messages, OTPs, or SMS content are uploaded or stored externally.”

---

14. Risk Analysis

⚠️ Risk 1: Play Store rejection

Mitigation:

- Narrow SMS usage
- BroadcastReceiver only
- Strong privacy policy

⚠️ Risk 2: Bank format changes

Mitigation:

- Regex versioning
- Remote pattern updates (no SMS upload)

⚠️ Risk 3: User privacy concerns

Mitigation:

- Transparent onboarding
- On-device processing
- Manual mode fallback

⚠️ Risk 4: Incomplete data

Mitigation:

- Multi-source ingestion:
    - SMS
    - Notifications

- Receipt links
- Manual input

---

15. MVP Scope (Version 1)

You should **NOT build everything at once**.

🎯 MVP Features

Core

- Notification ingestion
- SMS ingestion (basic)
- Transaction list
- Category auto-detection
- Basic dashboard

Verification

- Receipt URL extraction
- Manual verification button
- Verified badge

UI

- Dashboard
- Transaction list
- Settings
- Permission onboarding

Security

- On-device DB
- Basic privacy policy

---

16. Post-MVP Roadmap

Phase 2 (2–3 months)

- Budgeting
- Category editing
- Export CSV
- Better charts

Phase 3 (3–6 months)

- Forecasting
- Recurring detection

- Smart insights
- Financial score

Phase 4 (future)

- Bank API integrations (if Ethiopia opens APIs)
- SME mode
- Multi-device sync

---

17. Technical Architecture (High-Level)

📱 Frontend

- React Native (Expo) or Kotlin
- Tailwind / NativeWind UI
- Chart library

⚙️ Backend

- Optional (Firebase / Supabase)
- For premium features only

📦 Local Storage

- SQLite (Room / WatermelonDB)

🔄 Data Flow

```
SMS / Notification / Receipt
        ↓
Parsing Engine
        ↓
Transaction Normalizer
        ↓
Local Database
        ↓
UI Dashboard + Analytics
```

---

18. UX Principles

LedgerOne must feel:

✔ Clean
✔ Trustworthy
✔ Professional
✔ Financial-grade

Design rules

- Large balance numbers
- Clear debit/credit colors
- Minimal clutter
- Card-based layout

- Clear verification badges

---

19. Go-To-Market Strategy (Ethiopia)

🎯 Target users

1. Young professionals
2. Freelancers
3. Remote workers (like you)
4. Small business owners
5. Students

📢 Acquisition channels

- Telegram fintech groups
- TikTok short demos
- YouTube demo
- University communities
- Tech Twitter / X

🎁 Early adoption hook

> “Track all your money across CBE, Telebirr, Awash and BOA in one app.”

---

20. Launch Plan

Pre-launch

- Build MVP
- Test with 20–50 users
- Fix parsing issues

Soft launch

- Release APK privately
- Collect feedback

Public launch

- Play Store submission
- Marketing push
- Influencer demos

---

21. Vision (Long-term)

LedgerOne becomes:

> “The financial operating system for individuals in markets without open banking.”

Future:

- AI financial assistant
- Investment insights
- Loan readiness score
- Credit history builder