## **Smart Ledger - AI Financial Insights Module Implementation Guide (LLM-Centric)**

### **📌 Objective**

Build an AI-powered analytics module for "Smart Ledger" that delivers high-value insights using GPT and supporting APIs. The system must assist users in smart financial decision-making through:

* Categorization
* Budget tracking
* Spending insights
* Recurring detection
* Savings suggestions
* Digest reports

**Target:** Designed for implementation using GPT or Claude (without human developers), this document serves as a complete build spec with workflows, prompts, schema updates, and UX descriptions.

# **Product Requirement Document (PRD)**

### **Module: Real-Time Purchase Insight & Savings Advisor**

**Target Release:** v1  
 **Part of:** Smart Ledger – Financial Analytics Suite  
 **Owner:** CPO / Product Manager  
 **Collaborators:** Engineering, Data Science, UX/UI

## **🧭 Overview**

This module transforms raw receipt data into real-time insights, helping users track spending, detect recurring expenses, and find opportunities to save money — all powered by AI and enhanced by live pricing data and historical behavior.

## **🎯 Objectives**

|  |  |
| --- | --- |
| **Goal** | **Description** |
| 💡 Increase financial awareness | Detect spending patterns and overspending behavior |
| 💸 Maximize savings | Offer live price comparisons and smarter purchasing suggestions |
| 🔁 Track recurring expenses | Identify repeat purchases and recommend efficient alternatives |
| 📈 Stay on budget | Alert users when they approach or exceed category-specific budgets |
| 📬 Provide digestible insights | Deliver weekly summaries with key learnings and opportunities |

## **🧩 Final Feature List (v1)**

|  |  |
| --- | --- |
| **Feature** | **Description** |
| ✅ **Smart Categorization** | Auto-tag items from receipts into categories like groceries, dining, utilities, etc. |
| ✅ **Budget Tracking & Alerts** | Users can define monthly category budgets. App notifies when nearing/exceeding them. |
| ✅ **Recurring Expense Detection** | System identifies repeated purchases and suggests optimizations (bulk buy, reduce frequency, etc.) |
| ✅ **Savings Suggestions** | GPT + external price source compares past purchase price to market prices and recommends cheaper options. |
| ✅ **Weekly Smart Digest** | Summarizes overspending, top categories, and missed savings in a concise, shareable format. |
| ✅ **Real-Time Purchase Insight** | Instantly analyzes new purchases and offers feedback on how the user could have spent smarter. |

## **📐 Functional Requirements**

|  |  |  |
| --- | --- | --- |
| **ID** | **Feature** | **Description** |
| FR1 | **Receipt Insight Trigger** | Triggered post-OCR, each receipt generates itemized insights. |
| FR2 | **Categorization via GPT** | GPT assigns category to each item (stored in category field). |
| FR3 | **Budget Configuration** | Each user can set monthly budgets per category (stored in new budgetConfig model). |
| FR4 | **Budget Breach Alerts** | Alert sent when spending in a category nears or crosses budget threshold. |
| FR5 | **Recurring Purchase Engine** | If same item is found ≥2 times in 30 days, flag as recurring and suggest savings. |
| FR6 | **Live Price Comparison** | Use static/mock or API data to compare current vs. market prices and compute savings. |
| FR7 | **GPT Insight Generator** | Craft clear summary of price insight: “You paid ₹X, now it’s ₹Y, you could have saved ₹Z.” |
| FR8 | **Digest Aggregator** | Weekly cron job collects key insights and pushes them to user digest. |

## **🧱 Data Models**

### **1. insightItems**

js

CopyEdit

{  
 userId: ObjectId,  
 itemName: String,  
 category: String,  
 detectedPrice: Number,  
 matchedMarketPrice: Number,  
 receiptId: ObjectId,  
 dateDetected: Date,  
 savings: Number,  
 insightText: String,  
 isRecurring: Boolean,  
 isTracked: Boolean  
}

### **2. budgetConfig**

js

CopyEdit

{  
 userId: ObjectId,  
 categoryBudgets: [  
 {  
 category: String,  
 monthlyLimit: Number,  
 currentSpend: Number  
 }  
 ],  
 lastResetDate: Date  
}

## **📊 Weekly Digest Format (Sample)**

json

CopyEdit

{  
 userId: ObjectId,  
 weekStart: Date,  
 topCategories: ["Dining", "Groceries"],  
 overspentCategories: ["Dining"],  
 missedSavings: [  
 {  
 item: "Shampoo",  
 paid: 350,  
 couldHavePaid: 270,  
 saved: 80  
 }  
 ],  
 recurringAlerts: ["Coffee", "Food Delivery"]  
}

## **🔄 Backend Integration Plan**

|  |  |
| --- | --- |
| **Folder** | **Action** |
| controllers/insights.js | New logic for category tagging, budget checking, savings calculation |
| controllers/budget.js | For CRUD on budgetConfig |
| routes/insights.js | POST /insights/from-receipt |
| routes/budget.js | POST /budget, GET /budget |
| models/insightItem.js | Schema for insights |
| models/budgetConfig.js | Schema for budget settings |
| utils/gptInsightPrompt.js | Templates for GPT-based insight generation |
| scripts/priceLookup.js | Price matching logic (mock API or scraper) |
| scripts/recurringDetector.js | Looks at item history to detect repeat behavior |

## **🔐 Non-Functional Requirements**

|  |  |
| --- | --- |
| **NFR** | **Description** |
| ⚡ Performance | All insights must process within 3–5 seconds after receipt upload |
| 🔒 Security | JWT-based auth applies; insight data is user-scoped and encrypted |
| 📱 UX/UI | Fully mobile-optimized; visually highlights savings and over-budget alerts |

### **✅ Features Covered**

1. Smart Categorization
2. Budget Tracking & Alerts
3. Recurring Expense Detection
4. Savings Suggestions
5. Weekly Smart Digest
6. Real-Time Purchase Insight

## **✅ Feature 1: Smart Categorization**

### **🎯 Purpose**

Automatically assign categories (e.g., groceries, dining, utilities) to items extracted from receipt OCR.

### **🧠 GPT Prompt Example**

{"prompt": "Categorize: 'Kellogg's Chocos 375g' into one of these: groceries, dining, utilities, travel, personal care."}

### **🧭 User Flow**

1. OCR extracts items → trigger categorization
2. GPT assigns each item a category
3. System saves it to DB
4. Category is shown as label/chip in UI

### **⚙️ Backend Logic**

* On receipt upload, categorize all items
* Add category field to each item object

#### **MongoDB Schema (Receipt Model Update)**

items: [  
 {  
 name: String,  
 price: Number,  
 category: String // e.g., groceries  
 }  
]

### **🖼️ Wireframe**

Receipt Item List:  
---------------------------------------------------------  
| Kellogg’s Chocos 375g ₹175 [Groceries ✅ 95%] |  
| Dominos Pizza Medium ₹450 [Dining ✅ 91%] |  
| Tide Detergent 3kg ₹345 [Groceries ✏️ 88%] |  
---------------------------------------------------------  
[+] Add Custom Category

## **✅ Feature 2: Budget Tracking & Alerts**

### **🎯 Purpose**

Allow users to set monthly budgets per category and get alerts when nearing/exceeding those budgets.

### **👥 User Actions**

* Add/edit budgets per category
* View total spend vs budget
* Get notification when near limit

### **🧭 User Flow**

1. User configures budgets per category
2. Each new expense adds to monthly total
3. If total > 80% of budget → warning alert
4. If > 100% → critical alert

#### **MongoDB Schema (New budgetConfig model)**

userId: ObjectId,  
category: String,  
limit: Number,  
month: String // e.g., '2025-04'

#### **Alert Conditions**

if (totalSpend >= 0.8 \* limit) → warn  
if (totalSpend >= limit) → alert

### **🖼️ Wireframe**

Your Budgets - April 2025  
-------------------------------------------------  
| Groceries ₹3,400 / ₹4,000 🟡 85% Used |  
| Dining ₹2,100 / ₹2,000 🔴 105% Used |  
| Utilities ₹700 / ₹1,500 🟢 47% Used |  
-------------------------------------------------  
[Set Budget] [Enable Alerts ✓]

## **✅ Feature 3: Recurring Expense Detection**

### **🎯 Purpose**

Automatically detect repeated purchases (e.g., monthly Netflix, groceries) and offer optimization suggestions.

### **🧠 GPT Prompt Example**

{"prompt": "Is this item likely recurring? 'Netflix Subscription ₹499'"}

### **🧭 Logic Flow**

1. Search for same item bought ≥2 times in 30 days
2. Mark item as recurring: true
3. UI can display suggestion: “This item is recurring. Consider annual plan to save.”

#### **Schema Update (Receipt Model)**

recurring: Boolean // flag at item level

#### **Cron Job**

* Run daily
* Check receipts for repeat items
* Set recurring = true if condition met

### **🖼️ Wireframe**

Recurring Purchases Detected:  
------------------------------------------------------------  
| Netflix Subscription ₹499/month 🔁 Recurring |  
| Suggestion: Save ₹1,000/year with annual plan [Optimize] |  
------------------------------------------------------------

## **✅ Feature 4: Savings Suggestions**

### **🎯 Purpose**

Identify cheaper alternatives or bulk-buy suggestions for previously bought items.

### **🧠 GPT Prompt Example**

{"prompt": "Suggest cheaper or bulk alternative for: 'Surf Excel Matic 2kg ₹498'"}

### **🧭 User Flow**

1. Post-OCR → compare item price with market
2. If savings found → GPT crafts suggestion
3. Display below item in receipt detail or savings tab

#### **Example Output**

“You paid ₹498 for Surf Excel. Buying a 4kg pack at ₹900 saves ₹96.”

### **🔁 Source Data**

* Use mock price DB or scraping tools
* Optionally use PriceBefore, ScrapeHero, or cached Flipkart/Amazon data

### **🖼️ Wireframe**

Smart Suggestions:  
------------------------------------------------------------  
| Surf Excel Matic 2kg ₹498  
| 🟢 Cheaper Option: ₹450 (Amazon)  
| 💸 You could’ve saved ₹48  
| [🛒 View Deal] [Why?]  
------------------------------------------------------------

## **✅ Feature 5: Weekly Smart Digest**

### **🎯 Purpose**

Provide a weekly summary of user’s financial activity with actionable insights.

### **🧭 Digest Content**

* Total spend
* Overspent categories
* Top 3 expenses
* Missed savings (sum of gptInsight deltas)
* Tip of the week (from GPT)

### **🧠 GPT Prompt Example**

{"prompt": "Summarize this week’s spend pattern and give 1 saving tip."}

### **🕒 Cron Job (Every Sunday 6PM)**

* Aggregate user receipts from the past 7 days
* Summarize and send digest

#### **Email Content Example**

🧾 Weekly Spend: ₹9,800  
📊 Overspent in: Dining (₹2,300/₹2,000)  
💸 You missed savings worth ₹315  
💡 Tip: Consider preparing meals 2x per week to reduce dining cost.

### **🖼️ Wireframe**

🧾 Your Weekly Financial Digest (Apr 1 - Apr 7)  
  
- Total Spend: ₹9,800  
- Top Category: Dining (₹2,300)  
- Missed Savings: ₹315  
- Recurring Alerts: 2 Items  
- Tip of the Week: 🍳 Cook 2 meals to save ₹600  
  
[📄 View Details] [📤 Email to Self]

## **✅ Feature 6: Real-Time Purchase Insight**

### **🎯 Purpose**

Automatically analyze a new receipt post-OCR and provide a smart purchase insight, such as price comparison and better buying suggestions.

### **🧠 GPT Prompt Example**

{"prompt": "Analyze this item: 'Surf Excel Matic 2kg - ₹498', and compare with online prices. Return savings insight if possible."}

### **🧭 User Flow**

1. User uploads receipt → OCR extracts items
2. System sends GPT prompt per item (or batch)
3. GPT returns insight
4. UI displays insight below item

#### **Conditions:**

* Max 100 GPT calls/min → batch items or defer

20 receipts queued → cron job handles next batch

### **🖼️ UI Description**

* Show smart tag below each item
* Collapse summary for 3+ savings: “You missed ₹135 in savings on this receipt”

### **⚙️ Backend Logic**

#### **MongoDB Schema Update (Receipt Model)**

items: [  
 {  
 name: String,  
 price: Number,  
 gptInsight: String,  
 category: String,  
 recurring: Boolean  
 }  
]

#### **New Service**

* generateInsightsForReceipt(receiptId)
* Call GPT for each item
* Save insight back

### **🕒 Cron Job (Every 4 hours)**

* Job: processPendingReceipts
* Find receipts with items[].gptInsight = null
* Retry GPT-based insight generation

### **🔌 Free APIs for Price Comparison**

* [PriceBefore](https://www.pricebefore.com/)
* [WebScrapingAPI](https://www.webscrapingapi.com/)
* Flipkart/Amazon scraping (static prices)

### **🖼️ Wireframe**

[Receipt Upload Successful ✅]  
  
Itemized Insight:  
-------------------------------------------------  
| Dominos Pizza ₹450 🍽️ Dining |  
| 🔎 Suggestion: 20% coupon available next time! |  
-------------------------------------------------  
| Surf Excel 2kg ₹498 🧺 Groceries |  
| 🔎 You could save ₹48 via 4kg pack on Amazon |  
-------------------------------------------------

Reference wireframe

