# Opportunity Agent MVP – Engineering Requirements

## 1. Introduction

The Opportunity Agent MVP detects procurement anomalies across seven predefined detector types without per-detector thresholds, applying a single global minimum financial impact filter to ensure actionable results. This document contains complete functional, UI, non-functional, and acceptance requirements for the engineering team to build the MVP.

## 2. Scope

The MVP will implement seven fixed detectors, ingest procurement datasets, calculate potential savings, and output results to both Excel and dashboard feeds. No configuration beyond the global minimum financial impact is required for MVP.

## 3. Functional Requirements

|  |  |  |
| --- | --- | --- |
| FR ID | Requirement | Description |
| FR-1.0 | Data Ingestion | Ingest data from purchase\_orders, invoices, contracts, price\_benchmarks, indices, shipments, and supplier\_master tables. |
| FR-1.1 | Data Validation | Validate field completeness, data types, referential integrity, and currency codes. |
| FR-2.0 | Currency Normalisation | Convert all financials to GBP using FX rates from indices at transaction date. |
| FR-2.1 | Index Adjustment | Adjust contract prices using relevant index when index\_link is defined. |
| FR-3.0 | Opportunity Detection | Implement the 7 detectors: Unit Price vs Benchmark, Contract Price Drift, PO↔Invoice Discrepancy, Early Payment Discount Missed, Demand Aggregation, Logistics Cost Outliers, Supplier Consolidation. |
| FR-3.1 | Financial Impact Calculation | For each finding, calculate potential savings in GBP. |
| FR-4.0 | Global Financial Filter | Exclude opportunities where financial\_impact\_gbp < min\_financial\_impact (default: 100 GBP). |
| FR-5.0 | Output – Excel | Generate Excel with summary + per-detector sheets sorted by savings descending. |
| FR-5.1 | Output – Dashboard Feed | Write results to opportunity\_findings table and/or JSON API feed matching schema. |
| FR-6.0 | Auditability | Include original transaction IDs in source\_records field for all findings. |
| FR-7.0 | Repeatable Runs | Same dataset must produce identical outputs on repeated runs. |

## 4. Data Model

The following entities and relationships form the core MVP data model. All data must be structured to support detector queries efficiently.

|  |  |
| --- | --- |
| Table | Key Fields |
| purchase\_orders | po\_id, supplier\_id, category\_id, item\_id, unit\_price, quantity, currency, order\_date, contract\_id |
| invoices | invoice\_id, po\_id, supplier\_id, item\_id, unit\_price, quantity, currency, invoice\_date, payment\_terms |
| contracts | contract\_id, supplier\_id, category\_id, item\_id, agreed\_price, currency, start\_date, end\_date, index\_link |
| price\_benchmarks | item\_id, benchmark\_price, currency, source, effective\_date |
| indices | index\_name, value, effective\_date, currency |
| shipments | shipment\_id, po\_id, logistics\_cost, currency, delivery\_date |
| supplier\_master | supplier\_id, supplier\_name, region, risk\_rating |

## 5. Detection Logic

Each detector applies SQL or equivalent logic over the dataset:

|  |  |
| --- | --- |
| Detector | Logic |
| Unit Price vs Benchmark | Identify POs where unit\_price > benchmark\_price by > X% (no threshold per detector; global financial impact filter applies). |
| Contract Price Drift | Detect invoices where price differs from agreed\_price after applying index adjustment. |
| PO↔Invoice Discrepancy | Find mismatches in unit\_price or quantity between PO and corresponding invoice. |
| Early Payment Discount Missed | Identify invoices with discount terms not taken despite early payment potential. |
| Demand Aggregation | Find multiple small POs for same supplier/item/category in short period that could be consolidated. |
| Logistics Cost Outliers | Highlight shipments where logistics\_cost deviates significantly from category average. |
| Supplier Consolidation | Find categories with multiple suppliers where volume could be shifted to fewer suppliers for better rates. |

## 6. Processing Flow

1. Data ingestion from source tables.  
2. Data validation and normalisation (currency, date formats, referential integrity).  
3. Apply index adjustments where applicable.  
4. Run all 7 detectors.  
5. Apply global financial impact filter.  
6. Calculate savings and enrich with category/supplier metadata.  
7. Write results to Excel and dashboard feed.  
8. Log run metadata for audit.

## 7. Output Schema

Findings table schema for dashboard/API feed:

|  |  |
| --- | --- |
| Field | Description |
| opportunity\_id | Unique identifier for the finding |
| detector\_type | One of 7 fixed detectors |
| supplier\_id | Link to supplier\_master |
| category\_id | Category of spend |
| item\_id | Item/part identifier |
| financial\_impact\_gbp | Savings potential in GBP |
| calculation\_details | JSON breakdown of calculation |
| source\_records | List of related PO/invoice/contract IDs |
| detected\_on | Timestamp of detection |

## 8. Non-Functional Requirements

- Process ≥100k rows in under 3 minutes.  
- Use indexed queries for detector logic.  
- Support RBAC (role-based access control).  
- Encrypt all data at rest and in transit.  
- Log all detections for audit with user who initiated run.

## 9. Revised Acceptance Criteria

1. All 7 detectors run successfully without manual configuration.  
2. ≥3 varied findings per detector type in MVP demo dataset.  
3. Findings vary by supplier, SKU/category, and savings amount.  
4. Savings calculations match expected formulas within ±0.1% tolerance.  
5. Outputs match defined Excel and dashboard schema exactly.  
6. Identical dataset yields identical results.  
7. End-to-end run (ingestion → output) completes within 3 minutes for 100k rows.