Product Requirements Document (PRD)

Scout Analytics MVP – Retail Intelligence Dashboard

Version: 1.1 **Date:** June 21, 2025 **Owner:** TBWA Philippines **Status:** Implementation

---

1. Executive Summary

Scout Analytics MVP is a four-page retail intelligence dashboard that ingests 18,000+ Philippine transactions in real time. It delivers an L0 executive pulse, L1 focus pages (trends, mix, consumer), L2 drawer details and L3 raw-data modals—complete with global filters, click-to-drill-down and AI-powered recommendations.

---

2. Business Objectives

- **Primary:** Ship a production-ready analytics dashboard for TBWA client decks
- **Secondary:** Drive data-led retail optimizations across Philippine segments
- **Tertiary:** Lay groundwork for a SaaS analytics platform

---

3. Success Metrics

Category	Metric	Target
Performance	Initial load time	< 2 s
	Filter response	< 200 ms
	Uptime	> 99.5 %
Engagement	Session duration	> 5 min
	Pages/session	> 3
	Filter usage rate	> 70 %
Business Impact	AI insights/session	⩾ 5
	Client presentation success	> 90 %
	Decision implementation rate	> 40 %

4. Architecture Overview

```yaml
frontend:
  framework: React 18
  language: TypeScript
  styling: Tailwind CSS
  state: Zustand + URL sync + localStorage
backend:
  db: Supabase (PostgreSQL)
  rls: enabled
  realtime: Supabase subscriptions
ai:
  provider: Azure OpenAI (GPT-4, GPT-3.5, embeddings)
deploy:
  hosting: Vercel
  ci_cd: GitHub Actions
monitoring:
  metrics: Prometheus + Grafana
  errors: Sentry
```

5. Data Architecture & Backend Requirements

5.1 Core Tables

Table	Columns
`transactions`	`id` (UUID PK), `created_at` (timestamptz, idx), `store_id` (FK), `customer_id` (FK), `total_amount`
`transaction_items`	`id` (UUID PK), `transaction_id` (FK), `product_id` (FK), `quantity` (int), `price` (decimal)
`products`	`id`, `name`, `category` (idx), `brand_id` (FK)
`brands`	`id`, `name`
`stores`	`id`, `name`, `barangay` (idx), `region`, `location` (geography)
`demographics`	`transaction_id` (FK), `gender`, `age_bracket`
`behavior_signals`	`transaction_id` (FK), `request_type`, `suggestion_accepted` (bool)

5.2 Materialized Views

- `mv_hourly_volume`: (hour, count, sum_amount)
- `mv_category_mix`: (category, count, share)
- `mv_substitution_flow`: (from_brand, to_brand, count)

Refreshed every 5 minutes via scheduled SQL jobs.

5.3 ETL & Real-Time

- **Batch ingestion:** Supabase `.range()` loop in 1,000-row chunks, with retry up to 3×
- **Enrichment:** demographic & behavior signals via audio/video inference
- **Realtime:** Supabase subscriptions to push inserted rows to connected clients
- **Cache:** Redis (5 min TTL, stale-while-revalidate) for heavy endpoints

6. API Layer

All endpoints must accept the full global filter set: `from`, `to`, `barangays[]`, `stores[]`, `categories[]`, `brands[]`, plus optional `hour`, `gender`, etc.

GET /api/transactions:

- **params:**
  - `from`: date
  - `to`: date
  - ...global filters...
  - `limit`: int (default 1000)
  - `offset`: int (default 0)
- **returns:**
  - `headers`: `X-Total-Count`
  - `body`: `[ { transaction… }, … ]`

GET /api/volume:

- **params:** same global filters
- **returns:** `{ data: [ { hour, count, sum_amount }, … ], count }`

GET /api/category-mix:

- **params:** same
- **returns:** `{ data: [ { category, count, share }, … ], count }`

GET /api/substitution: GET /api/demographics: GET /api/behavior-signals:

- **params:** same
- **returns:** similar shape: `data` + `count`

POST /api/ai-insights:

- **body:** `{ filters }`

- **returns:** `{ insights: [ { insight, confidence, category, action_items }, … ] }`
- **Indexes:** on all filterable columns (`created_at`, `category`, `brand_id`, `store_id`)
- **Pagination:** exact first‐call `count` head query → loop pages of 1000

---

7. Feature Requirements

7.1 Global Filter System (P0)

- **Types:** date range, barangay/region, category, brand, store
- **Persistence:**
    1. Zustand in-memory
    2. URL sync (`?from=…&categories=…`)
    3. localStorage fallback
- **UI:** top-bar with pill badges + reset icon + active‐count badge

7.2 Page & Visual Matrix

Page	Key Visuals	Toggles & Drill-downs
L0: Overview	4 KPI cards, sparkline, top 5 products bar	KPI‐card clicks → `/transaction-trends?from…&to…`
L1: Transaction Trends	line (daily), box, heatmap (hour×day)	time-slot click → add `hour=`, navigate `/transaction-trends`
L1: Product Mix	pie (category), stacked bar, Sankey (sub)	slice click → `categories=`; flow edge click → `brands=`; navigate `/product-mix`
L1: Consumer Insights	donut (gender), funnel, geo-heatmap	segment click → `gender=`; map region click → `barangays=`; navigate `/consumer-insights`
AI Insights (L0)	text panel of ⩾ 3 insights + confidence bar	auto‐refresh on any filter change → `POST /api/ai-insights`

7.3 Drill-down Interaction Contract

1. User click on a chart element
2. `FilterStore: .setFilter(key,value)`
3. `Router: navigate("/target-page?"+currentQueryString)`
4. Page hook re-fetches via matching `/api/...` endpoint

---

8. Security & Compliance

- **Row‐Level Security:** enforced on `transactions` & related tables
- **Auth:** JWT issued by Auth0/Azure AD; validate on each request
- **CORS:** allow only frontend domain
- **Rate Limiting:** AI endpoint capped at 60 req/min per user

---

9. Monitoring & Alerting

- **APM:** Prometheus + Grafana for latency & error SLOs
- **Error Logging:** Sentry for 5xx & auth failures
- **Health:** `/api/health` ping every 60 s

---

10. Quality Assurance

- **Unit Tests:** Jest ⩾ 80 % coverage on hooks, utils, API mocks
- **E2E Tests:** Playwright for filter persistence & drill-down flows
- **Accessibility:** Axe CI (no violations; charts with ARIA)
- **Performance:** Lighthouse CI (perf ⩾ 90, a11y ⩾ 90)

---

11. Implementation Timeline

Phase	Deliverables	ETA
Phase 1	Core data model, batch fetch hook, scaffold pages	Jun 17 AM
Phase 2	Global filters + URL sync + localStorage	Jun 18 PM
Phase 3	Drill-down interactions & navigation	Jun 19 AM
Phase 4	AI panel integration + `/api/ai-insights`	Jun 20 PM
Phase 5	Performance & accessibility tuning	Jun 22 AM
Phase 6	QA, full test suite, CI/CD	Jun 23 PM

---

12. Risk Assessment

Risk	Mitigation
Large-batch timeouts	Retry logic + server-side RPC count queries
AI downtime / rate limits	Cache last insights + fallback messaging
Mobile slow networks	Code-split charts, lazy load non-critical comps