FINTECH COMPETITIVE INTELLIGENCE

Neobank Competitive Intelligence & Strategy Engine

Turning noisy neobank competitor feeds into sprint-ready specs — with an architecture built so it cannot invent a fact.

Live Radar Platform

TypeScript · Zod · Vercel

Public Repo

Mereke Dadabayeva

Technical Product Owner · mereke.me · github.com/MerekeDadabayeva

THE HOOK

A pricing page changes. The system diffs it, classifies it, and ships a sprint-ready ticket — with zero invented facts.

100%	100%	0%	2.8s
Precision	**Recall**	**Hallucination Rate**	**Synthesis Latency**
n=20 benchmark	n=20 benchmark	by architecture, not just measurement	avg. per verified signal

Measured with the project's own automated test suite — every claim on this slide is a reproducible test result, not marketing copy.

CONTEXT & MARKET PROBLEM

Two failure modes of generic competitive monitoring

European neobanks — **Trade Republic, N26, Revolut, Scalable Capital, Bitpanda —** ship pricing and product changes continuously. Product teams need to know what changed and why it matters, fast — without a system that quietly guesses.

1. Generic Feed Syndrome

Aggregating press releases and marketing pages without ever answering “what does this mean for our roadmap?”

2. Hallucinated Synthesis

LLM-based tools pointed at scraped pages fill gaps with plausible but unverified claims — a real liability when the claim is a fee or an interest rate.

Why it matters for a PM

- A fabricated fee number in a Monday brief is a real liability in a regulated fintech context.
- Engineering sprint capacity is finite — noisy feeds burn it on non-issues.
- Executive trust is fragile: one wrong claim and the whole feed gets ignored.
- Competitive response needs to be same-day, not “next sprint after we double-check.”

A 4-stage verified intelligence pipeline

1

Validated Ingestion

Real HTTP fetch with bot-block detection, anchor-keyword assertion, and a >40% content-drop check to catch silent site redesigns.

→

2

Deterministic Diff

Line-level diff (jsdiff) between the current and prior snapshot. If it's not in the diff, it does not exist to the system.

→

3

Schema-Enforced Synthesis

Rule-based classifier tags category, JTBD pillar, and impact — every output validated against a strict Zod schema before it reaches the UI.

→

4

Execution Bridge

Auto-generates a mini-PRD, sprint-ready Jira Gherkin story, and weekly/daily digest markdown from each verified signal.

13 monitored sources across N26, Revolut, Scalable Capital & Bitpanda · 5 JTBD pillars · auto-routed to publish or human review

THE KEY DESIGN DECISION

Why there is no LLM in the synthesis loop

Zero hallucination isn't a benchmark score here — it's a property of the architecture.

Generative summary (common approach)

- ✗ LLM reads scraped page, asked to describe what changed
- ✗ Fills gaps with plausible-sounding but unverified claims
- ✗ Risk scales with every source added
- ✗ Fabricated fee or rate = real liability in fintech

Deterministic rule classifier (this system)

- ✓ Classifier only reads the diff's added/removed lines
- ✓ No generative step exists to invent a claim
- ✓ Every output validated against a strict Zod schema
- ✓ Trade-off: needs structured domains (pricing, changelogs) to work

Benchmarked against 20 realistic fixtures, tested on 10 more it never saw

Metric	Target	Benchmark (n=20)	Held-out (n=10)
Precision	≥ 90%	100.0%	100.0%
Recall	100%	100.0%	100.0%
Hallucination rate	0%	0.0%	0.0%
Synthesis latency	< 5.0s	2.8s	3.1s

Being precise about what this proves

These are hand-authored fixtures modeled on each competitor's real, current pricing pages — not a months-long live crawl. What they validate is that the diff → classify → schema-validate logic behaves correctly and consistently across a realistic range of inputs, including a held-out set the classifier never saw during development. Proving it holds up against the live internet over time is the natural next step (see Roadmap).

PRODUCT SURFACE

What the Trade Republic PM actually sees

1 Live Signal Feed

13 monitored sources across 4 competitors, filterable by JTBD pillar and triage state (Moat Lead / Defensive / Noise).

2 Executive Brief & Matrix

Auto-generated weekly digest in Visual, Slack, and Email formats — one click to copy into a stakeholder update.

3 Data Integrity Inspector

Every signal links back to its raw ingestion payload, timestamp, and unified diff — nothing is asserted without a receipt.

4 What-If Simulator

Models yield-spread sensitivity and transfer-bounty economics against the Trade Republic baseline in real time.

HOW THE CLASSIFIER DECIDES

Business impact drives triage — not keyword-matching alone

Three real rules from the classifier, applied to the signals shown on Slide 2 & 3:

NOISE (LOW ROI)

P3 – Ignore

IF a competitor move is a pure luxury/status tier (metal card, lounge access, concierge)

→ *Negligible retail volume impact. Preserves developer sprint focus for higher-leverage work.*

DEFENSIVE NEED (PARITY)

P1 – Next Sprint

IF a competitor's cash rate reaches parity with the Trade Republic baseline (3.75%)

→ *Retention messaging required for high-balance accounts before churn risk builds.*

DIFFERENTIATOR (MOAT)

P0 – Immediate Response

IF a competitor's onboarding shows a verified KYC failure loop (e.g. app rating drop)

→ *Prime acquisition window: contrast against Trade Republic's 3-minute frictionless signup.*

WHAT I DID — THE EXECUTION BRIDGE

From verified signal to sprint-ready ticket

Actual output from data/signals.json for the N26 signal on Slide 2 — not a mockup.

```
Epic: [COMP-INTEL] Strategic Response to N26 PRICING

Scenario: User views comparative product advantages
  Given a user has an active uninvested cash balance > 0 EUR
  When they view the cash interest / account overview
  Then they see their compounded 3.75% p.a. payout
  and accrued 1% Saveback total clearly displayed

Acceptance:
  ✓ Tracking events emitted for banner impressions
  ✓ No latency regression (<200ms p95 render)
```

V1 Scope Discipline

Auto-generated for every mini-PRD:

- ✗ Do NOT subsidize unsustainable promotional CAC spikes or match temporary referral bounties
- ✗ Do NOT introduce complex tiered loyalty points that degrade fee transparency
- ✗ Do NOT alter the core €1.00 execution fee without Pricing Committee sign-off

KEY SIGNALS & IMPACT

What this demonstrates about how I work

CS Fluency

Real HTTP fetch layer, jsdiff-based diff engine, and Zod-validated schemas — not a mocked UI over static JSON.

V1 Scope Discipline

Every generated mini-PRD ships explicit out-of-scope boundaries, protecting engineering bandwidth from creep.

Impact-Based Triage

Signals are prioritized by business impact, not just keyword match — a fee-parity move outranks a UI rebrand.

Resilient by Design

Static GitHub Pages deployment embeds a full fallback dataset — the UI never breaks even with no backend.

WHAT I'D BUILD NEXT

From a curated demo to a continuously live radar

1

Scheduled live crawl

Wire the existing fetcher to a GitHub Actions cron against the real source URLs already defined in config — turning the demo dataset into a genuinely live feed.

2

Narrow, schema-constrained LLM fallback

Add a generative step only for diff lines the rule classifier can't confidently tag — keeping the zero-hallucination guarantee for the common case.

3

Source health triage view

A lightweight admin panel over bot-blocked or drifted sources, so they get fixed instead of silently logged.

LET'S CONNECT

Built to demonstrate end-to-end product-engineering judgment

Source validation, deterministic data pipelines, schema enforcement, and translating machine-verified signals into artifacts a product team can act on the same day.

View Repo	github.com/MerekeDadabayeva/Competitive-Intelligence-Neobank-Sector
Launch Live Radar	merekedadabayeva.github.io/Competitive-Intelligence-Neobank-Sector
Portfolio	mereke.me

Mereke Dadabayeva · Technical Product Owner · Berlin, Germany