COGNITRA - Governed Market Intelligence for Automotive

Student: Sandrine Lepesqueux | Course: MSIS 549 – Generative AI Technologies

COGNITRA is an AI-powered market intelligence system for automotive closure systems, built for Tier-1 supplier Apex Mobility. It converts unstructured documents into validated intelligence records using a minimal-AI pipeline: LLMs handle extraction and synthesis, deterministic Python rules handle scoring and quality checks, and a human approval gate governs executive brief generation. Built with Streamlit, Gemini Flash-Lite/Flash, Python, and JSONL storage.

Key skills developed include schema-first API design with Gemini's response_schema, prompt engineering with explicit numbered rules and competitor context blocks, AI-assisted development workflows using Claude Code and Codex inside a full repository context (VSCode), GitHub versioning, and enforcing clear boundaries between LLM-extracted and computed fields with runtime guardrails. I also developed a practical understanding of spec-to-code consolidation — managing specification drift during iterative AI-assisted development and resolving it by embedding domain guidance directly into the runtime prompt.

Executive Summary

Market intelligence for automotive closure systems faces a structural bottleneck: premium sources produce more relevant content than any analyst can process, resulting in missed signals, inconsistent outputs, and no mechanism to detect patterns across documents. For a Tier-1 closure systems supplier, these gaps translate into delayed awareness of OEM strategy shifts, overlooked technology changes, and reactive competitive positioning.

COGNITRA addresses this by converting unstructured documents — PDFs and pasted text from sources such as S&P Global, Bloomberg, and Automotive News — into structured, validated intelligence records using a minimal-AI pipeline. A single LLM call per document (Google Gemini Flash-Lite or Flash) extracts strict-schema JSON with controlled taxonomy topics, region classifications, and evidence bullets. All downstream processing — schema validation, region normalization, priority and confidence scoring, macro-theme detection, and duplicate suppression — runs deterministically in Python with no model involvement. A human approval gate governs which records surface in executive briefs, and a read-only KPI pipeline tracks quality trends without mutating production records.

This architecture produces five reliability properties: (1) **structured record reliability** — all inputs converted to schema-validated JSON with deterministic postprocessing; (2) **governance by design** — a human approval gate with full mutation and provenance trail; (3) **quality-calibrated scoring** — confidence and priority computed from observable signals, not model self-assessment; (4) **duplicate suppression and canonicalization** — story-level deduplication with source-quality ranking; and (5) **cost and runtime predictability** — meta-based model routing and quota tracking that reduced API consumption by an estimated 30–50%.

Evaluation across three documented test cases and approximately ten source-verified documents demonstrated consistent schema validation on clean PDFs, accurate geographic classification after postprocess hardening, and reliable priority escalation for business-critical signals. Five of six identified failure modes — including government entity hallucination, publisher confusion, and geo signal distortion — were resolved through targeted extraction rules and deterministic guards. One residual risk remains: entities spanning chunk boundaries. User evaluation at the GenAI Fair reinforced these results: all six participants perceived clear differentiation from standard AI summarization, and five of six confirmed the tool addresses a real problem. The system was demonstrated end-to-end in a walkthrough video, ingesting real licensed articles and producing executive-ready outputs with traceable evidence.

1. Business Problem & Significance

This project started from a personal pain point. As a one-person market intelligence function covering automotive closure systems and car entry, the challenge was not finding good information — it was having no time to process it. Premium sources (Bloomberg, S&P Global, Automotive News, MarkLines) produce a continuous stream of relevant content. In practice, most of it never got processed: some articles were forwarded with a quick comment, many were never shared at all. The intelligence was there; the bandwidth to convert it into something useful was not.

This bottleneck scales with team size but never disappears. It combines four compounding problems:

- **Volume exceeds processing capacity.** Entire categories of signal are routinely missed — not because they were unavailable, but because they were never reached.
- **Manual processing produces inconsistent, non-reusable outputs.** Topics and regions are tagged differently across analysts. Summaries live in email chains that are not searchable or comparable over time.
- **No reliable middle ground between over-sharing and silence.** Without a structured triage layer, most workflows default to silence for anything requiring more than two minutes of attention.
- **No cross-document signal detection.** Documents are processed and shared individually. There is no mechanism to detect when multiple sources processed days or weeks apart collectively point to the same emerging trend — a pattern visible across five records is invisible when each lives in a separate email.

In practice, this often results in PDFs being stored in folders with inconsistent naming conventions, irregular executive updates, and a loss of institutional knowledge when someone leaves who managed the files. For a Tier-1 closure systems supplier, the tangible costs of missed signals include incorrect assumptions about OEM strategy, overlooked technology changes (such as smart entry, digital keys, and cinch systems), and reactive rather than proactive competitive positioning —to name just a few critical consequences. The solution must be effective for an individual today and scalable for a team in the future.

An AI-powered solution is well-suited to this problem because the core bottleneck is not judgment — it is volume and consistency. **Extracting structured metadata from a news article, classifying it against a fixed taxonomy, flagging it for relevance, and synthesizing signals across multiple records are tasks a language model can perform reliably and at scale.** COGNITRA implements exactly this division: LLMs handle extraction, deterministic rules enforce consistency, and the analyst retains control over what reaches executive stakeholders.

2. Solution Approach & Design Process

2.1 Concept

COGNITRA ingests an automotive source document (PDF with selectable text or pasted text; OCR not yet integrated) and converts it into **structured, validated intelligence records** using a minimal-AI pipeline:

- **One LLM call per document** for strict-schema JSON (Gemini 2.5 Flash-Lite primary, Flash fallback).
- **Deterministic postprocessing and validation** to normalize fields, compute priority/confidence, and enforce governance rules.
- **Human review gate** (Pending, Approved, Disapproved) with deterministic auto-approve heuristics.
- **Deterministic rendering** of a single-record Intelligence Brief from stored JSON.
- **Executive Brief Synthesis** (LLM) generated only from approved JSON records (currently limited to API quota of 20 calls/day).
- **Insights analytics** from validated, stored records (topic momentum, company mentions, region-topic matrix, quality trend).

The core principle is **structure before synthesis**: extraction first, validation second, synthesis last.

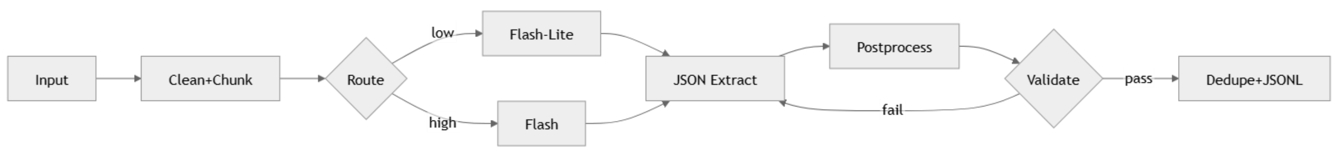
2.2 Why Minimal AI

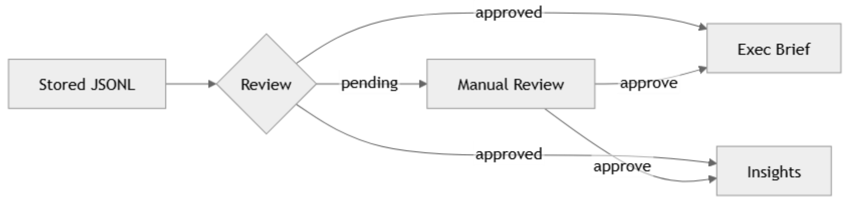
Minimal AI is an architectural constraint rather than a product claim. AI is used only where deterministic logic cannot reliably perform the task: schema-controlled structured extraction, controlled taxonomy classification, and cross-record synthesis after human approval. Everything else is deterministic, including PDF extraction, cleaning, chunking, schema validation, deduplication, scoring, macro-theme detection, and brief formatting, which keeps outputs auditable, comparable across documents, and resilient to model variability.

Full LLM summarization per document was evaluated and rejected because it produces non-comparable outputs with weak query ability and higher hallucination exposure without a strict schema boundary. A RAG-first conversational architecture was also considered and rejected because it optimizes retrieval and Q&A rather than producing governance-ready records that can be validated, deduplicated, filtered, and trended over time. The selected approach, strict-schema extraction plus deterministic validation plus human approval, aligns with COGNITRA's goal: traceable, repeatable intelligence generation.

2.3 System Architecture

The pipeline implementation is detailed in §3.3; module-level breakdown in §4.0.





The following tools and libraries implement each stage of the pipeline:

Tool	Role
Google Gemini 2.5 Flash-Lite / Flash	LLM structured JSON extraction + executive brief synthesis
Streamlit	Multipage web app (5 pages + Home)
PyMuPDF + pdfplumber	PDF text extraction with fallback
Altair + pandas	Interactive trend analytics
Python 3.9+ / pytest	Language and test framework (102 tests, 5 modules)
JSONL (flat files)	Record storage — no database required

Detailed diagrams covering model routing decision logic, spec evolution phases, LLM vs. deterministic data contract, and the full ingest sequence are provided in *Appendix G*.

3. Data & Methodology

3.1 Data Sources

All documents ingested during development and testing were real licensed articles from active subscriptions. Sources used:

- S&P Global Mobility: monthly registration reports, OEM strategy notes, supply chain alerts.
- Bloomberg: OEM strategy articles, technology partnership announcements, financial results.
- Automotive News: tested via the copy-paste text input mode (no PDF upload).
- MarkLines: automotive production and sales data reports.
- OEM Press releases and others.

COGNITRA stores structured intelligence records derived from licensed or publicly available sources. Original pdf documents are not included in the public GitHub repository due to copyright restrictions on licensed intelligence content.

Test cases in §5 reference real documents by publisher and headline description only.

3.2 Model and Tool Selection Rationale

Google Gemini 2.5 Flash-Lite and Flash were chosen for their free tier access (20 requests/day each), unlike OpenAI and Anthropic APIs, which lacked sufficient free options for iterative development. Flash-Lite is used for straightforward metadata extraction from clean text, preserving Flash quota for more complex tasks. Flash serves as a fallback for noisy documents and schema repair, and is the primary model for executive brief synthesis requiring cross-record reasoning. This Flash-Lite-first approach efficiently manages quota.

Gemini's response_schema parameter was also a key factor in the design: it enforces strict JSON output at the API level, meaning the model is constrained to return exactly the structure you define rather than just being instructed to do so via prompt.

See Appendix E for full output schema.

3.3 Data Processing Pipeline

Raw PDF text is extracted using PyMuPDF as the primary extractor, with pdfplumber as fallback for documents where PyMuPDF returns sparse or malformed output. Extracted text goes through a deterministic cleanup stage (src/text_clean_chunk.py) that removes headers, footers, legal boilerplate, and repeated navigation elements, then chunks the result into bounded segments with 800-character overlap to preserve cross-boundary entities. A noise classification step examines removal ratios and patterns to route each document to the appropriate model tier before the LLM call.

Postprocessing runs deterministically after extraction: region normalization against data/new_country_mapping.csv (~90 country mappings), priority boosting, confidence scoring, macro-theme detection, duplicate detection, and schema validation. No LLM is involved in postprocessing. All field-level changes are recorded in a _mutations audit trail per record.

4. Technical Implementation

4.1 System Overview

COGNITRA is a Streamlit multipage application (5 pages + Home) converting automotive PDFs and pasted text into governed intelligence records and executive briefs. Design decisions and iterations are in §4.1–4.4. *See Appendix B for implementation timeline.*

The system comprises 14 modules from PDF extraction through quality monitoring; *see Appendix E* for the full module breakdown.

Auto-approve conditions: confidence not Low AND publish_date present AND source_type not Other AND 2+ evidence bullets. Records failing any condition go to Pending.

Repair logic: one repair attempt on schema validation failure — repair prompt instructs the model to fix specific violations without re-extracting the full record.

Where to Start: README.md for setup and run commands; AGENTS.md for system invariants and pipeline rules. Key source files: model_router.py, postprocess.py, schema_validate.py. Streamlit pages: 01_Ingest.py through 04_Insights.py. Tests and quality monitoring: docs/quality/, run_quality.py, and tests/.

4.2 Key Iteration: From External Specs to Code-Embedded LLM Guidance

The project began with five Markdown files — master spec, topic taxonomy, competitor watchlist, prompts, and brief template — which forced clarity upfront and were useful for early alignment. As the codebase evolved, those documents fell behind. App behavior changed, docs did not always keep up, and critical logic had no enforcement mechanism. This created four concrete problems: priority, confidence, and macro themes needed to be computed the same way every time (determinism); schema and computed-field boundaries needed to be enforced at import and runtime, not just described (guardrails); pytest can validate Python functions, not prose (testability); and Streamlit pages and the pipeline needed to read one canonical implementation, not a document that might be stale (single source of truth).

A dedicated refactoring session moved all policy logic from documentation into code: topic tagging rules into src/constants.py and the extraction prompt; competitor context injected directly into the prompt; AGENTS.md established as the single operator manual for humans and AI agents. Data-driven configuration (new_country_mapping.csv) was preserved where appropriate. Impact: before consolidation, the LLM confused "OEM Strategy" with "OEM Programs" and misapplied "Closure Technology & Innovation" to general vehicle electronics. After consolidation, with explicit use/don't-use rules per topic and competitor names in the prompt, topic classification accuracy improved measurably.

See Appendix G3 for a visual summary of the three phases.

4.3 Extraction Prompt Design (13 Rules)

The extraction prompt started with 5 general extraction rules. Each iteration of the ChatGPT source-verification workflow (see §5.1) exposed specific failure modes — publisher confusion, geo signal distortion, government entity hallucination — which were resolved by adding targeted rules and corresponding regression tests. The 13-rule prompt and the 102-test suite are both products of this iterative cycle.

The most persistent issue was publisher-vs-cited-source classification (rules 1–2): as an example an S&P Global article cited Reuters in the body, the model consistently set source_type to "Reuters" — a plausible but incorrect extraction, since Reuters was the cited wire source, not the publisher. This required multiple iterations to resolve reliably, because the model's behavior was defensible from a surface reading of the text; the fix needed explicit instructions prioritizing document header signals over in-body citations.

Other important rules: Rule 5 (evidence bullets capped at 25 words, verbatim facts only); Rule 7 (at least one numeric fact required when present); Rule 8 (government entities must be explicitly named, no country-context inference); Rule 10 (country_mentions restricted to operational markets only, excluding geopolitical backdrop countries). *See Full verbatim prompt in Appendix A.*

4.4 Deterministic Scoring: Priority, Confidence, and Macro Themes

All scoring runs in postprocess with no model call involved. Computed confidence replaced LLM self-assessment after discovering the model rated confidence as High regardless of quality. The fix uses observable signals: +2 for publish_date present, +2 for known source_type, +1 to +2 for evidence bullet count, +1 for key_insights, +1 for non-empty regions_relevant, -1 per 3 postprocess corrections, -1 if publish_date was backfilled by regex. Thresholds: >= 7 → High, >= 4 → Medium, < 4 → Low. Full audit trail stored in _confidence_detail per record.

Priority boosting (_boost_priority()) applies domain-specific escalation after LLM assessment. Key supplier escalation: a Tier 1 or Tier 2 competitor with a qualifying signal — confirmed footprint region (footprint_and_key_supplier), closure-relevant topic (key_supplier_and_closure_topic), or closure-relevant keyword (key_supplier_and_closure_keyword) — escalates to High. A competitor mention alone without a qualifying signal remains Medium, preventing noise escalation from passing references.

4.5 Meta-Based Model Routing and Two-Phase Repair

With 20 requests per day on both Gemini Flash-Lite and Flash under the free tier, the original per-chunk two-pass strategy wasted calls: Flash-Lite failed on every noisy-document chunk before escalating, burning 2 calls per chunk. The redesign uses cleanup metadata (removed_ratio, removed_line_count) to choose the model upfront. High-noise documents route directly to Flash. Phase 1 runs all chunks with the chosen model; Phase 2 retries only failed chunks with the stronger model — successful chunks are never re-extracted. Selective escalation (_should_retry_strong()) ensures only schema or structural failures trigger retry. Estimated impact: 30-50% reduction in API call consumption.

5. Results & Evaluation

5.1 Evaluation Approach

The executive brief is only as reliable as the upstream extraction. Brief generation depends on structured fields produced during PDF-to-JSON extraction — source_type, publish_date, companies_mentioned, topics, evidence_bullets — and if extraction is incorrect or incomplete, errors propagate into prioritization, trend analytics, and the final brief narrative. This dependency is why extraction quality controls (schema validation, deterministic postprocessing, confidence scoring, and human review gating) are treated as **critical safeguards,** not optional steps.

Evaluation combines three methods: (1) automated schema and postprocess validation on every ingested record, backed by 102 tests across 5 modules (*see Appendix C*); (2) three documented test cases with manual source verification; and (3) post-hoc quality monitoring via src/quality.py with 10 KPIs *(see Appendix D for definitions*). Approximately 10 documents were verified by attaching the source PDF to a separate AI chat (ChatGPT) and requesting a critical review of the extracted JSON against the original — checking for hallucinations, wrong field values, and unsupported claims. Multiple extraction rules (publish date handling, region guards, geo signal distortion fix) were directly derived from issues caught in those sessions (*see Appendix H for examples*).

Six participants evaluated COGNITRA during the GenAI and Agentic Fair via a structured feedback form. Five of six confirmed the tool addresses a real problem; all six perceived clear differentiation from standard AI summarization, citing governance, traceability, and cross-document pattern detection as standout features. Five of six indicated they would use the tool actively or occasionally. The most valued capabilities were filtering signal from volume and cross-record pattern detection — both core to COGNITRA's design. Respondents were primarily technical (4 engineers/data professionals, 2 students); executive-level feedback was not captured.

A 0–3 rubric across five dimensions — schema validity, evidence grounding, geo determinism, publisher classification, and priority accuracy — was applied to all test cases. *See Appendix F for rubric definitions and per-test-case scores.*

5.2 Test Case A — Market Note (Western European Registrations, S&P Global)

Document type: S&P Global monthly registration report covering Germany, France, Spain, Italy, and the UK, with a section on Chinese-owned brands gaining UK market share (BYD, Chery, SAIC MG, Great Wall Motor).

Two failure modes were caught and fixed. First, government entity hallucination: before rule 8, the pipeline returned government_entities = ["EU", "European Commission"] despite neither being named — the model inferred EU membership from France/Spain/Germany context. After the no-inference rule, government_entities = []. Second, false China region and macro theme escalation: keywords from the Chinese brands paragraph caused the postprocess region hint to inject China into regions_mentioned, triggering the China EV Competitive Acceleration macro theme and falsely escalating priority to High on a European report. A deterministic China guard resolved this. Verified result: regions_mentioned = ["West Europe"], macro_themes_detected = [], priority = Medium.

5.3 Test Case B — OEM Press Release with Date and Region Errors (Toyota Argentina)

Document type: Toyota Argentina press release (published Feb 7, 2026) announcing 2025 results. The article body references a Feb 4, 2026 internal release date. Two errors on initial ingest: publish_date extracted as 2026-02-04 (internal event date) instead of 2026-02-07 (document header date); regions_mentioned set to ["United States"] from a keyword-triggered postprocess hint despite the article covering Argentina, Chile, and Peru. Both fixed with targeted rules — publisher header parser now extracts dates from document headers first, preserving internal dates as event_date; US guard added mirroring the China guard. Regression test validates: publish_date = 2026-02-07, event_date = 2026-02-04, regions_mentioned includes Mercosul (not United States).

5.4 Test Case C — OEM Strategy with Macro Theme Detection (Mercedes-Benz / Bloomberg)

Document type: Bloomberg article on Mercedes-Benz unveiling the CLA EV with AI voice controls, with a header timestamp of February 1, 2026. Three behaviors tested: (1) Publisher date override — LLM extracted 2026-01-29 (Berlin event date in body); postprocess Bloomberg header parser correctly overrode to 2026-02-01, storing event date separately with diagnostic flags. (2) mentions_our_company false positive — LLM set this True; postprocess corrected to False after verifying Apex Mobility did not appear in source-grounded fields. (3) Macro theme detection — SDV and Luxury OEM Stress themes correctly fired; Germany derived West Europe in regions_relevant, triggering High priority via key OEM + footprint boost. Confidence = 7 → High.

See Appendix F for scores.

5.5 Aggregate Results

Metric	Target	Observed
Schema validation (first-pass)	>= 80%	Consistent on clean PDFs; repair path triggered on noisy documents (S&P legal footers, multi-publisher headers)
Evidence grounding	>= 90%	Bullets traceable to source after rule 5 enforced; quality.py hard-miss threshold < 45% overlap → High severity

Geo determinism	>= 98%	Achieved after China/US postprocess guards; Toyota Argentina and Western EU regressions resolved and regression-tested
Publisher classification	>= 95%	S&P/Bloomberg/Reuters correct after rules 1-2; edge cases remain for whitelabel PDFs
Test suite	All pass	102 tests, 5 modules, all passing (.venv\Scripts\python -m pytest)
Token usage	<= 5,000/doc	Flash-Lite typically < 3,000 tokens; Flash escalation adds ~1,500-2,000
Processing time	<= 30s	Clean single-chunk docs < 15s; noisy chunked docs 20-40s

5.6 Failure Analysis

Across all test cases and development runs, six failure modes were identified. Five were resolved through targeted rules or architectural changes; one remains a known residual risk.

- Government entity hallucination (resolved): model inferred entities from country context before rule 8. Residual risk for uncommon government bodies not in the prompt examples.
- Publisher confusion (resolved): source_type set to cited wire source instead of publisher. Fixed by rules 1-2; residual risk on whitelabel PDFs.
- Geo signal distortion (resolved): geopolitical backdrop countries distorted region classification. Fixed by rule 10 and US/China postprocess guards.
- LLM overconfidence bias (resolved): model self-rated confidence as High regardless of quality. Fixed by deterministic computed score.
- Spec drift (resolved): see §4.1.
- Chunk boundary effects (residual): entities or dates spanning chunk boundaries can be missed. 800-character overlap mitigates but does not eliminate this risk.

6. Limitations & Future Work

6.1 Current Limitations

No authenticated source integration requires manual PDF upload for each document — production would connect to source APIs or monitored document libraries directly. Storage is flat-file JSONL with no concurrent writers, no indexing, and performance that degrades past approximately 5,000 records. The free-tier quota of 20 requests per day per model constrains batch processing. The system was designed and validated for a single analyst — multi-user environments need role management and concurrent write handling. Input is English-only; non-English sources will produce degraded extractions, limiting coverage for Apex Mobility's European and Asian footprint. Finally, trend detection is limited to a predefined pattern set — emerging signals not matching an existing theme are silently unclassified rather than surfaced.

Some extraction rules in the extraction prompt *(appendix A)* remain tightly coupled to specific publisher signatures (e.g., rule 2 explicitly lists WSJ, FT, CNBC, Nikkei as "Financial News"); sources not matching a known pattern default silently to "Other," requiring ongoing rule maintenance as the source base expands.

6.2 Future Work (Prioritized)

For production deployment at Apex Mobility, two foundational tracks are required. First, externalizing hardcoded business logic (competitor list, taxonomy, priority rules, region mapping) into YAML/CSV configuration files loaded at runtime — the prerequisite for all downstream integration. Second, Microsoft environment integration: Azure hosting to resolve the free-tier quota, SharePoint document libraries as the PDF source with Power Automate triggering extraction and delivering briefs on schedule, and Azure SQL replacing JSONL for concurrent multi-analyst access.

Priority	Improvement	Rationale
High	Configurable architecture	Externalize business logic to YAML/CSV; prerequisite for production
High	Microsoft/Azure integration	Azure hosting, SharePoint ingest, Power Automate triggers, Azure SQL — resolves quota, storage, and multi-analyst limits
High	Multi-language support	French, German, Japanese, Spanish for European and Asian footprint
Medium	Emerging Signal Detection	Flag unrecognized terms appearing in 3+ records within 30 days
Medium	Re-ingest capability	Apply updated rules to existing records without re-running the model
Medium	Embedding-based duplicate detection	Better paraphrase matching than current SequenceMatcher
Low	Power BI Insights	Broader audience analytics without application access

7. Ethical Considerations

Three ethical considerations are worth noting. First, the publisher ranking hierarchy and topic taxonomy both encode Apex Mobility's perspective — Western, English-language sources rank higher, and topics outside the nine canonical buckets (labor relations, environmental compliance) may be miscategorized; both should be reviewed and expanded for any broader deployment. Second, PDFs from premium sources contain licensed content — stored evidence excerpts and JSONL records should be reviewed by legal counsel before production deployment. Third, the analyst controls which records appear in executive briefs, creating selective briefing risk; the human review gate is a core architectural safeguard, not an optional step — deploying in full-automation mode removes the principal check against systematic extraction errors reaching executive stakeholders.

8. References & Code

- Project repository: https://github.com/SandrineLpx/COGNITRA (original pdf source documents not redistributed; see §3.1.)
- Streamlit App: https://cognitra.streamlit.app/
- Demo walkthrough video: Cognitra Demo Recording
- Libraries: Google Gemini API (ai.google.dev) · Streamlit (streamlit.io) · PyMuPDF (pymupdf.readthedocs.io) · pdfplumber (github.com/jsvine/pdfplumber) · Altair (altair-viz.github.io)

9. AI Disclosure

Code Development and Project Architecture Claude Code (Opus/Sonnet 4.6) and Codex were used throughout development inside VS Code with full repository context — more effective than isolated prompt sessions because the AI could cross-reference behavior across files. Claude Code handled deeper repo-wide consistency checks; Codex handled rapid iteration.

The most instructive friction was with UI design. Codex consistently confirmed that Streamlit could support custom colors, animations, and layout positioning, suggesting HTML/CSS injections. I trusted it and lost hours on approaches that were not feasible — the AI continued proposing fixes rather than acknowledging the framework's constraints. The lesson: verify capability limits independently before investing implementation time. The UI was still improved *(see Appendix I)* but not to the original target, which motivated the Lovable mockup as a separate UX reference (see §8).

Domain judgment drove the decisions that mattered most: schema structure, evidence format, review workflow, quality bar, and the decision to default to Flash-Lite over Flash. The AI generated implementations efficiently; the decisions themselves required domain knowledge the AI did not have.

Report Writing Iteration logs and technical notes accumulated to over 90 pages. Claude Sonnet 4.6 condensed these into an initial structured draft. The author then iterated extensively — strengthening arguments, adding context not captured during development, and cutting content that did not serve the engineer-reader. AI assisted with condensing and review; all factual claims were verified against the codebase, and problem framing, design rationale, and evaluation criteria reflect the author's own analysis.

Prompt Design The author specified all extraction, repair, and synthesis rules; AI assistance was used to draft and refine prompt language.

Appendix — Additional Technical Detail

The following material provides verbatim artifacts needed to replicate exact system behavior.

A. Full Extraction Prompt (verbatim from src/model_router.py)

13 numbered rules plus CLOSURE SYSTEMS COMPETITORS and TOPIC CLASSIFICATION blocks. Injected into every Gemini API call at runtime.

```
You are extracting structured intelligence for Apex Mobility, an automotive closure
systems supplier (door latches, strikers, handles, smart entry, cinch systems).
Return JSON only matching the schema. Follow these rules strictly:

1) source_type is the PUBLISHER of the document. If 'S&P Global', 'S&P Global
Mobility', 'AutoIntelligence | Headline Analysis', or '(c) S&P Global' appears,
set source_type='S&P'. If MarkLines is the publisher, set source_type='MarkLines'.

2) If Reuters or Bloomberg is only cited inside the article, do NOT set source_type
to Reuters/Bloomberg unless they are clearly the publisher. Use 'Financial News'
for financial publications (WSJ, FT, CNBC, Nikkei) and 'Industry Publication' for
automotive trade press that are not Automotive News. Use 'Other' only when no
specific type fits.

3) actor_type must be one of: oem, supplier, technology, industry, other.
Use 'oem' for vehicle manufacturers; 'supplier' for closure system competitors;
'technology' for tech companies; 'industry' for broad market/sector items.

CLOSURE SYSTEMS COMPETITORS — set actor_type='supplier':
Tier 1: Hi-Lex, Aisin, Brose, Huf, Magna (Magna Closures/Mechatronics),
        Inteva, Mitsui Kinzoku
Tier 2: Ushin, Witte, Mitsuba, Fudi (BYD subsidiary), PHA, Cebi, Tri-Circle
Our company: Apex Mobility (set mentions_our_company=true if mentioned)

4) publish_date: normalize to YYYY-MM-DD. Handle '4 Feb 2026', 'Feb. 4, 2026'.
   Else return null.

TOPIC CLASSIFICATION — pick 1-4 topics:
- 'OEM Strategy & Powertrain Shifts': broad pivots (BEV/ICE, vertical integration).
  NOT single program updates.
```

```
- 'Closure Technology & Innovation': ONLY when latch/door/handle/digital key/
  smart entry/cinch appears explicitly. NOT general vehicle electronics.
- 'OEM Programs & Vehicle Platforms': specific program announcements. NOT strategy.
- 'Regulatory & Safety': regulations, standards, recalls. NOT general politics.
- 'Supply Chain & Manufacturing': plant openings/closures, disruptions, tariffs.
  NOT pure financial performance.
- 'Technology Partnerships & Components': where tech is central. NOT purely
  commercial alliances.
- 'Market & Competition': demand, registrations, share shifts. NOT exec changes.
- 'Financial & Business Performance': earnings, M&A, insolvency. NOT exec churn.
- 'Executive & Organizational': leadership changes, governance.

5) evidence_bullets: 2-4 bullets, each <= 25 words. Verbatim facts only.
6) key_insights: 2-4 analytical bullets. Do NOT repeat evidence; add analysis.
7) If numeric facts are present, at least one bullet must include a specific number.
8) government_entities: ONLY explicitly named bodies. Return [] if none named.
9) If article mentions AI/SDV features, include at least one evidence_bullet on it.
10) country_mentions: ONLY operational market data. NOT geopolitical backdrop.
    regions_mentioned: canonical values only (West Europe, NAFTA, ASEAN, Mercosul,
    etc. or individual Apex Mobility countries).
11) keywords: key topics, technologies, material company names. NOT country/region
    names, publisher names, or generic measurement phrases.
12) Deduplicate all lists. Normalize US->United States, UK->United Kingdom.
13) notes: leave empty unless important context missing elsewhere.

Use only the provided text.
```

B. Implementation Changelog

Date	Milestone	Key Changes
Feb 11	Foundation	Initial extraction pipeline, geography processing, schema validation, centralized constants
Feb 12	Core System	End-to-end pipeline (PDF → Gemini → JSONL), duplicate detection with publisher ranking, executive briefing module, bulk deduplication CLI, schema hardening, 25+ tests
Feb 13	Extraction Pipeline Hardening	Deterministic text cleanup and chunking, context pack assembly, two-pass model strategy with selective escalation, chunked extraction with repair and cross-chunk merge, bulk PDF ingest, auto-approve model
Feb 14	Analytics & Quality	Trend dashboards, API quota tracker, quality monitoring pipeline (scripts/run_quality.py) with R1-R5 and B1-B5 KPIs
Feb 16	Deterministic Scoring & Macro Themes	Priority boosting with key supplier escalation rules, computed confidence scoring, macro-theme detection engine, publisher header date parsing with diagnostics, meta-based model routing
Feb 17	Region Architecture & Doc Consolidation	Two-tier region model from data/new_country_mapping.csv (~90 countries), AGENTS.md as single operator manual, topic guidance and competitor list embedded in prompt, runtime context-path audit, repo hygiene
Feb 19	UX Flow Upgrade	Nine-stage pipeline stepper, queue-centric Review with tabbed detail panel and score diagnostics, Brief version comparison, Insights executive snapshot tiles with record traceability, Admin macro theme viewer — no schema break
Feb 21	Alignment & Documentation Pass	README rewritten for GitHub, Lovable UX mockup link, 'weekly brief' → 'executive brief' rename, 4 failing tests fixed, prompt snapshot tests moved to scripts/one_off/

C. Test Suite

Total: 102 tests across 5 modules, all passing. Run via: `.\.venv\Scripts\python -m pytest`

test_scenarios.py (30 tests) Duplicate detection (exact, whitespace, case), fuzzy story detection, source quality scoring (S&P > Bloomberg > Reuters > Other), executive brief candidate selection, brief and email rendering, title normalization, company canonicalization.

test_macro_themes.py (42 tests) Macro theme firing, anti-keyword suppression, region gating (Tariff requires US/China/Europe), premium company gate, strength scoring, rollup clusters, audit trail, backward compatibility, priority escalation, and key supplier escalation (alone → Medium; + footprint/closure topic/keyword → High).

test_regions_bucketed.py (24 tests) Two-tier region architecture, individual Apex Mobility countries by name, non-individual countries collapse to buckets (Canada → NAFTA, Vietnam → ASEAN, Sweden → West Europe), legacy alias normalization, city hints, schema relaxation, source type normalization.

test_publish_date_pdf.py (3 tests) PDF header date preferred over metadata, metadata fallback, Toyota Argentina regression (publish_date = 2026-02-07, event_date = 2026-02-04).

test_brief_qc.py (3 tests) Topic label bullets not flagged as ungrounded, uncited claim bullets correctly flagged, invalid REC IDs flagged.

D. Quality Monitoring KPIs (verbatim from docs/quality/QUALITY_KPIS.md)

Two-level scorecard: record extraction quality (R1-R5) and executive brief quality (B1-B5). Run via scripts/run_quality.py — read-only, does not mutate production records.

Record KPIs (weekly)

- KPI-R1: High-severity defects per record (target: 0). High = hallucination, wrong numeric, wrong company/OEM, wrong geo signal, wrong publisher.
- KPI-R2: Medium-severity defects per record (target: <= 1.0). Medium = duplicates, missing secondary signals, phrasing that alters meaning.
- KPI-R3: Evidence grounding coverage (target: >= 90%). % of records where all evidence bullets are source-verifiable. Hard-miss threshold: < 45% word overlap → High severity. Near-miss: 45-60% → Medium.
- KPI-R4: Canonicalization stability (target: >= 95%). % of records where canonical forms match reference tables.
- KPI-R5: Geo determinism pass rate (target: >= 98%). % of records where regions_relevant equals the deterministic result of country_mentions, with no display buckets leaking into footprint fields.

Brief KPIs (weekly)

- KPI-B1: Ungrounded claims count (target: 0).
- KPI-B2: Overreach count (target: <= 2). Overreach = stronger certainty than source supports, or Apex Mobility implication asserted without explicit support.
- KPI-B3: Uncertainty compliance (target: 100%). Brief must include >= 1 uncertainty item if any source record contains uncertainty language.

- KPI-B4: Synthesis density (target: >= 2 cross-record themes supported by >= 2 REC IDs).
- KPI-B5: Action specificity score (target: >= 4/5). 1 = generic advice; 5 = owner + timeframe + concrete task + trigger + artifact output.

Weighted Quality Score — Record (0-100): subtract 25 per high issue, 10 per medium, 2 per low. Brief (0-100): subtract 25 per ungrounded claim, 20 per wrong signal/certainty, 10 per overreach, 20 for missing uncertainty, 15 for missing cross-record synthesis. Cadence: weekly KPIs; monthly trendline + top recurring defects + rules calibration.

E. System Architecture Detail

Module Breakdown

Module	File	Role
PDF extraction	pdf_extract.py	PyMuPDF primary, pdfplumber fallback
Clean & chunk	text_clean_chunk.py	Remove noise, split into 800-char overlap segments
Context pack	context_pack.py	Keyword hits, watchlist matches, country scoring
Model routing	model_router.py	Flash-Lite primary, Flash for high-noise + repair
Postprocess	postprocess.py	Priority, confidence, macro themes, region normalization, _mutations audit trail
Validation	schema_validate.py	Pipeline invariant — no invalid record reaches storage
Deduplication	dedupe.py	Exact (dedupe_key) + fuzzy (SequenceMatcher 0.85); publisher ranking determines canonical
Storage	storage.py	Append-only records.jsonl
Review	02_Review.py	Pending / Approved / Disapproved lifecycle; auto-approve on quality threshold
Record brief	render_brief.py	Deterministic rendering from JSON — no LLM call
Executive brief	briefing.py	One Gemini Flash call, up to 20 approved records, full REC ID citation
Insights	04_Insights.py	Topic momentum, company mentions, region-topic matrix, QC trend
Quality	quality.py / run_quality.py	Evidence grounding (hard miss < 45%, near miss 45–60%), geo determinism, KPI trend tracking
Tests	pytest (5 modules)	102 tests — extraction, macro themes, regions, publish date, brief QC

Field Lineage (LLM vs. Deterministic)

Field	Source	Notes
title, evidence_bullets, key_insights, topics	LLM	Strict schema output via response_schema
publish_date	LLM + deterministic override	Header/PDF metadata overrides low-confidence dates
source_type	LLM + deterministic normalization	Publisher markers correct misclassification
country_mentions, regions_mentioned	LLM + deterministic canonicalization	Aliases normalized, countries collapsed to region buckets
priority	Deterministic only	Rule-based _boost_priority()
confidence	Deterministic only	_compute_confidence() from observable signals
macro_themes_detected	Deterministic only	Rule engine in postprocess

Output Schema (verbatim from src/model_router.py)

18 LLM-extracted fields and 18 computed fields added by postprocess (5 primary: priority, confidence, macro_themes_detected, priority_final, priority_reason; 13 diagnostic detail fields prefixed with _). The guardrail raises RuntimeError at import time if any required key is missing from both the LLM schema properties and the known computed fields set.

```
properties = {
    "title": {"type": "STRING"},
    "source_type": {"type": "STRING", "enum": sorted(ALLOWED_SOURCE_TYPES)},
    "publish_date": _nullable({"type": "STRING", "pattern": r"^\d{4}-\d{2}-\d{2}$"}),
    "publish_date_confidence": {"type": "STRING", "enum": sorted(ALLOWED_CONF)},
    "original_url": _nullable({"type": "STRING"}),
    "actor_type": {"type": "STRING", "enum": sorted(ALLOWED_ACTOR_TYPES)},
    "government_entities": {"type": "ARRAY", "items": {"type": "STRING"}},
    "companies_mentioned": {"type": "ARRAY", "items": {"type": "STRING"}},
    "mentions_our_company": {"type": "BOOLEAN"},
    "topics": {"type": "ARRAY", "items": {"type": "STRING", "enum": CANON_TOPICS},
            "minItems": 1, "maxItems": 4},
    "keywords": {"type": "ARRAY", "items": {"type": "STRING"},
              "minItems": 3, "maxItems": 15},
    "country_mentions": {"type": "ARRAY", "items": {"type": "STRING"}},
    "regions_mentioned": {"type": "ARRAY", "items": {"type": "STRING"}, "maxItems": 15},
    "regions_relevant_to_apex_mobility": {
        "type": "ARRAY",
        "items": {"type": "STRING", "enum": FOOTPRINT_REGIONS},
```

```
    },
    "evidence_bullets": {"type": "ARRAY", "items": {"type": "STRING"},
                         "minItems": 2, "maxItems": 4},
    "key_insights": {"type": "ARRAY", "items": {"type": "STRING"},
                     "minItems": 2, "maxItems": 4},
    "review_status": {"type": "STRING", "enum": sorted(ALLOWED_REVIEW)},
    "notes": {"type": "STRING"},
}
# Computed fields added by postprocess — not extracted by LLM.
_COMPUTED_FIELDS = {
    "priority", "confidence", "macro_themes_detected",
    "priority_llm", "priority_final", "priority_reason",
    "_publisher_date_override_applied", "_publisher_date_override_source",
    "_confidence_detail",
    "_macro_theme_detail", "_macro_theme_strength", "_macro_theme_rollups",
    "_region_migrations", "_region_ambiguity", "_region_validation_flags",
    "_provenance", "_mutations", "_rule_impact",
}
# Guardrail: fail loud at import time if required key is missing from both.
unexpected = set(REQUIRED_KEYS) - set(properties.keys()) - _COMPUTED_FIELDS
if unexpected:
    raise RuntimeError(f"Schema misalignment: {sorted(unexpected)}")
```

F. Test Case Rubric Scores

Detailed dimension-by-dimension scores for the three test cases in §5, using the 0-3 rubric.

Dimension	0 — Fail	1 — Partial	2 — Acceptable	3 — Strong
Schema validity	Invalid JSON or missing required fields	Schema passes with repair step	First-pass schema valid	First-pass valid + all enum constraints satisfied
Evidence grounding	Bullet not traceable to source (overlap < 45%)	Near miss: partial support (45-60% overlap)	All bullets traceable to source text	All bullets verbatim facts with numeric data where present
Geo determinism	Wrong region derived from non-operational country	Region present but includes incorrect extra entry	Correct regions, no spurious entries	Correct regions + correct footprint derivation
Publisher classification	Wrong source_type (cited source vs publisher)	source_type = Other when specific type available	Correct publisher identified	Correct publisher + publish_date accurate
Priority accuracy	Wrong priority tier (High when should be Low)	Priority off by one tier	Correct priority tier	Correct tier + correct reason key in audit trail

Test Case A — Western European Registrations (S&P Global)

Dimension	Score	Notes
Schema validity	3	First-pass valid after rule fixes
Evidence grounding	3	All bullets traceable; numeric registration data present
Geo determinism	3	West Europe only after China guard applied
Publisher classification	3	S&P correctly identified from document header
Priority accuracy	3	Medium — no footprint region, no macro theme fired

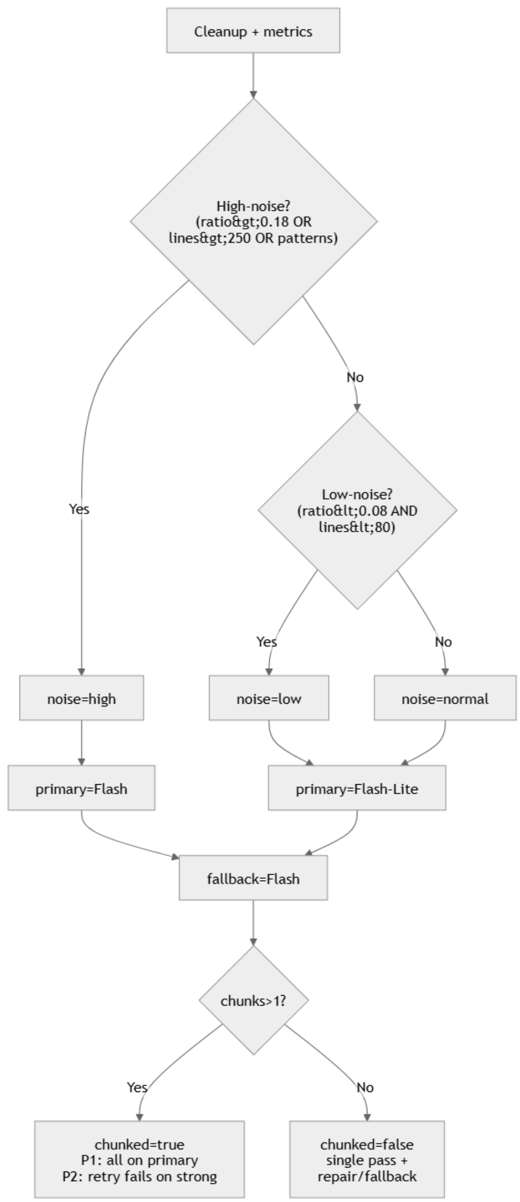
Test Case B — Toyota Argentina Press Release

Dimension	Score	Notes
Schema validity	3	First-pass valid after repair
Evidence grounding	2	Bullets traceable; numeric production/sales data present
Geo determinism	0 → 3	Initial: US false positive from keyword hint. After fix: Mercosul correct, US removed
Publisher classification	3	Press release correctly classified; header date override applied
Priority accuracy	2	Medium — Mercosul not in Apex Mobility primary footprint

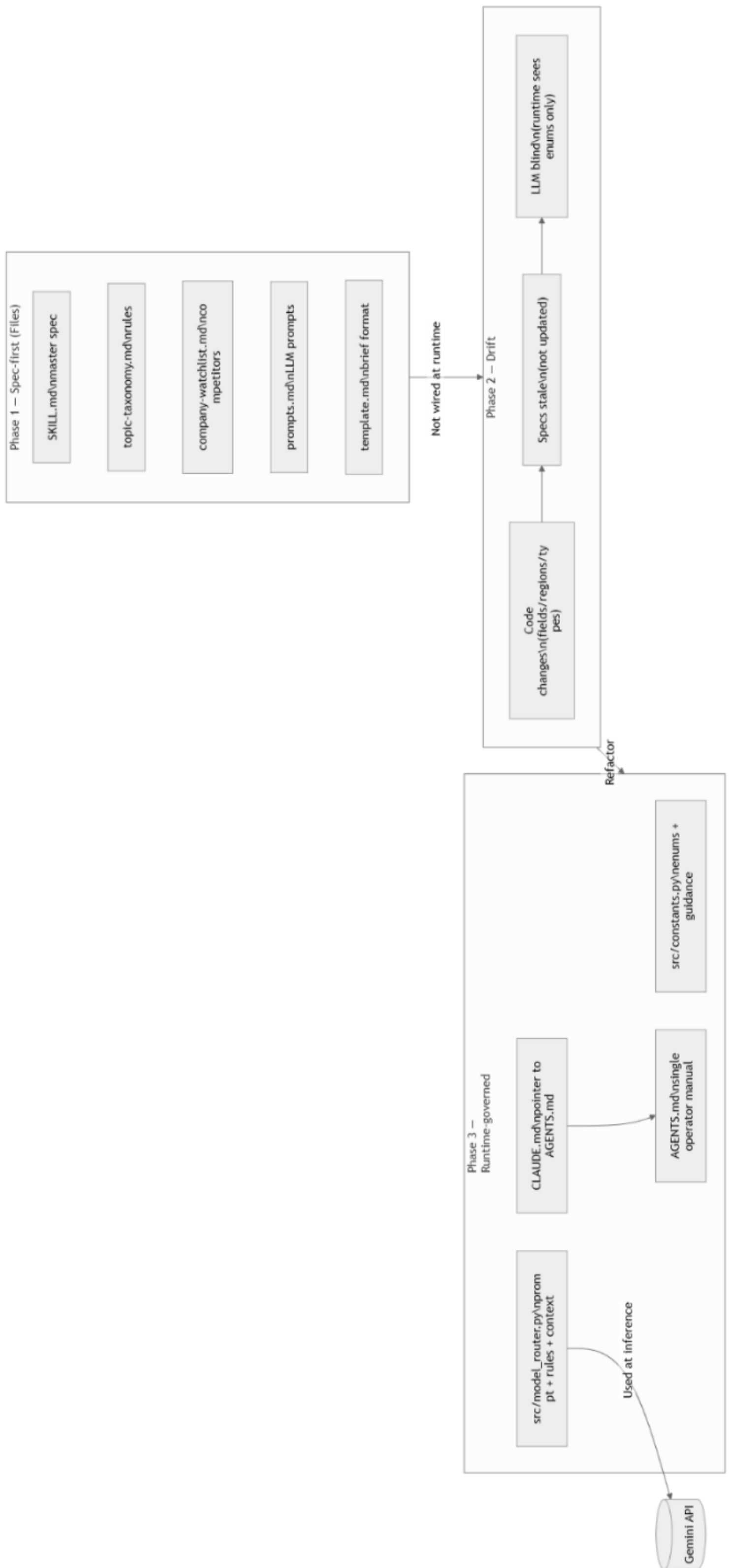
Test Case C — Mercedes-Benz CLA EV (Bloomberg)

Dimension	Score	Notes
Schema validity	3	First-pass valid
Evidence grounding	3	All bullets traceable; AI voice control feature captured per rule 9
Geo determinism	3	West Europe correctly derived from Germany in country_mentions
Publisher classification	3	Bloomberg header timestamp correctly overrode body event date
Priority accuracy	3	High — macro theme + footprint region, correct reason key in audit trail

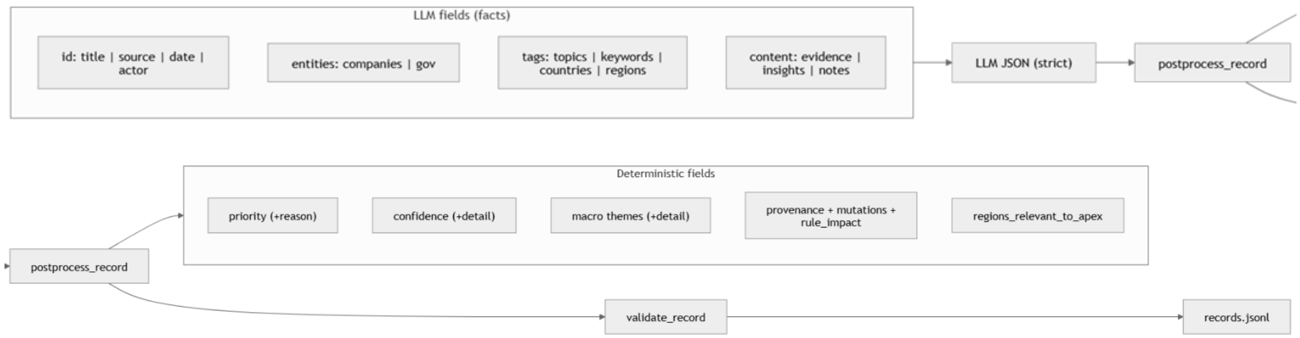
G1. Model Routing Decision



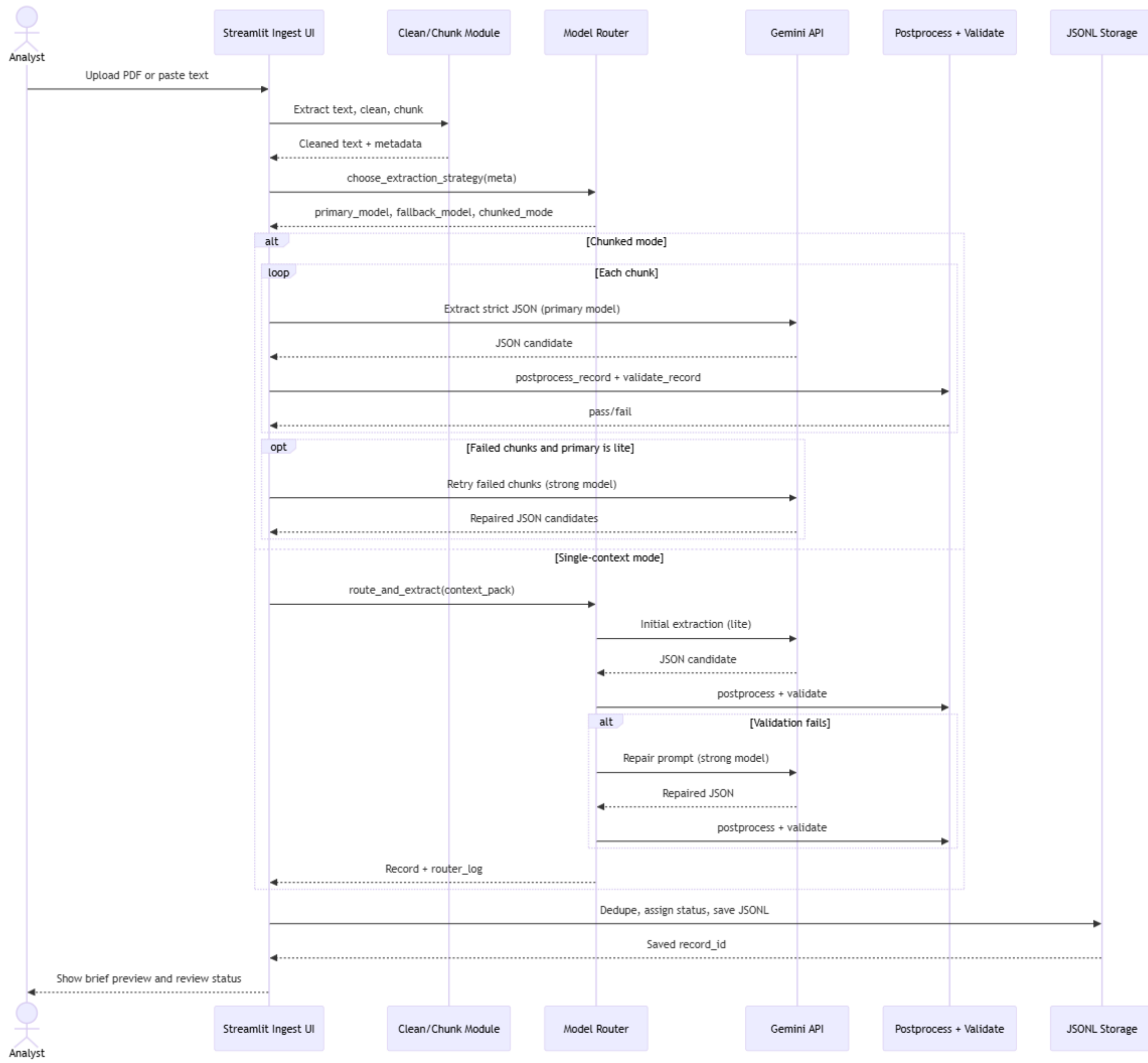
G2. Spec Evolution (Phase 1 → 2 → 3)



G3. Data Contract (LLM vs Deterministic boundary)



G4. Ingest Sequence (swimlane)



H. ChatGPT Quality Review Example

Approximately 10 documents were validated using a "second-model review" workflow. For each document, the original source PDF and the extracted JSON record were provided to an independent ChatGPT session and the model was asked to perform a critical, field-by-field comparison between the JSON and the PDF text.

The review focused on three primary failure modes using QUALITY_CHECKLIST.md (used as a manual/operational standard in AGENTS.md):

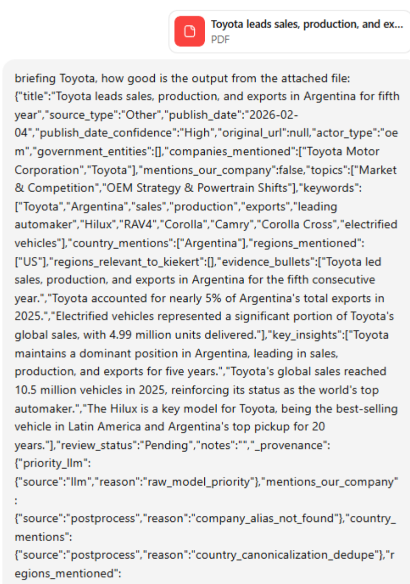
- Hallucinations: facts present in the JSON but not supported anywhere in the PDF
- Wrong extracted values: incorrect numbers, dates, entities, or classifications compared to the PDF

- Unsupported interpretation: key insights or implications written with a higher level of certainty than the source supports.

Screenshots below show 2 examples where the original PDF and the corresponding extracted JSON were sent to ChatGPT, followed by an explicit validation request. ChatGPT returned a structured list of discrepancies and improvement suggestions, which guided iteration on extraction and postprocessing rules.

This verification process was repeated until discrepancies were resolved (between 1 to 4 times). The most persistent issue was publication date extraction, because the publisher's header formatting and timestamp placement caused conflicting date candidates during parsing.

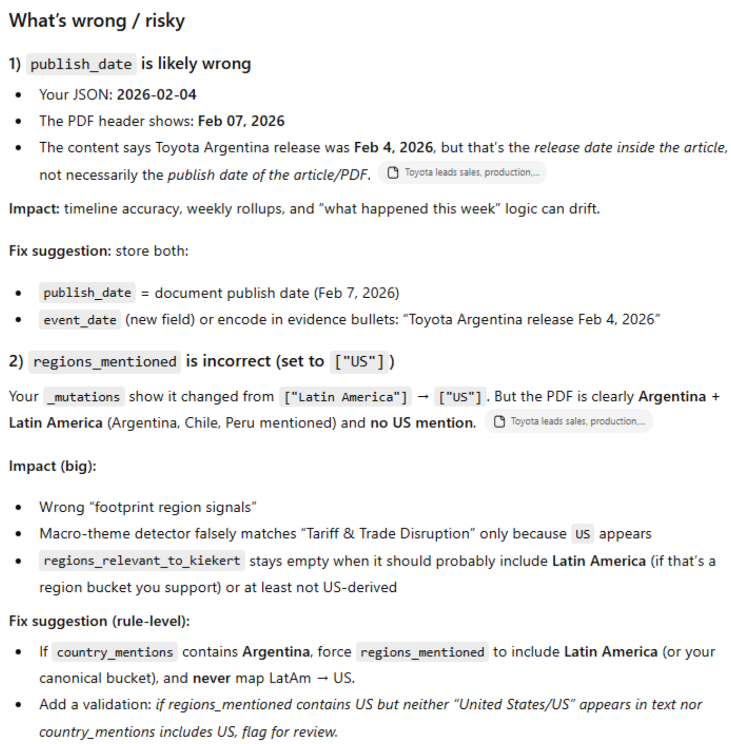
Example 1: Input / ChatGPT Output:



Toyota leads sales, production, and ex...
PDF

briefing Toyota, how good is the output from the attached file:

{"title":"Toyota leads sales, production, and exports in Argentina for fifth year","source_type":"Other","publish_date":"2026-02-04","publish_date_confidence":"High","original_url":null,"actor_type":"oem","government_entities":[],"companies_mentioned":["Toyota Motor Corporation","Toyota"],"mentions_our_company":false,"topics":["Market & Competition","OEM Strategy & Powertrain Shifts"],"keywords":["Toyota","Argentina","sales","production","exports","leading automaker","Hilux","RAV4","Corolla","Camry","Corolla Cross","electrified vehicles"],"country_mentions":["Argentina"],"regions_mentioned":["US"],"regions_relevant_to_kiekert":[],"evidence_bullets":["Toyota led sales, production, and exports in Argentina for the fifth consecutive year.","Toyota accounted for nearly 5% of Argentina's total exports in 2025.","Electrified vehicles represented a significant portion of Toyota's global sales, with 4.99 million units delivered."],"key_insights":["Toyota maintains a dominant position in Argentina, leading in sales, production, and exports for five years.","Toyota's global sales reached 10.5 million vehicles in 2025, reinforcing its status as the world's top automaker.","The Hilux is a key model for Toyota, being the best-selling vehicle in Latin America and Argentina's top pickup for 20 years."],"review_status":"Pending","notes":"","_provenance":{"priority_llm":{"source":"llm","reason":"raw_model_priority"},"mentions_our_company":{"source":"postprocess","reason":"company_alias_not_found"},"country_mentions":{"source":"postprocess","reason":"country_canonicalization_dedupe"},"regions_mentioned":{"source":"postprocess","reason":"regions_bucketed_deduped"},"regions



What's wrong / risky

1) `publish_date` is likely wrong

- Your JSON: **2026-02-04**
- The PDF header shows: **Feb 07, 2026**
- The content says Toyota Argentina release was **Feb 4, 2026**, but that's the *release date inside the article*, not necessarily the *publish date of the article/PDF.* Toyota leads sales, production,...

Impact: timeline accuracy, weekly rollups, and "what happened this week" logic can drift.

Fix suggestion: store both:

- `publish_date` = document publish date (Feb 7, 2026)
- `event_date` (new field) or encode in evidence bullets: "Toyota Argentina release Feb 4, 2026"

2) `regions_mentioned` is incorrect (set to `["US"]`)

Your `_mutations` show it changed from `["Latin America"]` → `["US"]`. But the PDF is clearly **Argentina + Latin America** (Argentina, Chile, Peru mentioned) and **no US mention.** Toyota leads sales, production,...

Impact (big):

- Wrong "footprint region signals"
- Macro-theme detector falsely matches "Tariff & Trade Disruption" only because `US` appears
- `regions_relevant_to_kiekert` stays empty when it should probably include **Latin America** (if that's a region bucket you support) or at least not US-derived

Fix suggestion (rule-level):

- If `country_mentions` contains **Argentina**, force `regions_mentioned` to include **Latin America** (or your canonical bucket), and **never** map LatAm → US.
- Add a validation: *if regions_mentioned contains US but neither "United States/US" appears in text nor country_mentions includes US, flag for review.*

Market Intelligence

2 files

Western_European_passenger_car_...
PDF

how good is the JSPN extraction below versus original pdg attached. Follow the quality_checklist attached in this folder.

{"title":"Western European passenger car registrations fall 2.7% year over year in January","source_type":"S&P","publish_date":"2026-02-06","publish_date_confidence":"High","original_url":null,"actor_type":"industry","government_entities":["EU"],"companies_mentioned":["S&P Global Mobility"],"mentions_our_company":false,"topics":["Market & Competition","Regulatory & Safety","Supply Chain & Manufacturing"],"keywords":["Western Europe","passenger car","registrations","market","January","forecast","ICE vehicles","BEV","malus payments","weight-based penalty","GDP growth","geopolitical environment","protectionism","tariffs","supply chains"],"country_mentions":["Germany","France","Spain","Italy","United Kingdom","United States"],"regions_mentioned":["Western Europe","US"],"regions_relevant_to_kiekert":["Western Europe","US"],"evidence_bullets":["Western

+

Input:

ChatGPT output:

Below is a QC-style comparison of your JSON extraction vs. the original S&P PDF, using your **Cognitra Weekly Quality Checklist** criteria. QUALITY_CHECKLIST Western_European_passenger_car_...

A1. High-severity blockers (must be ZERO)

1) Numbers / facts mismatch (BLOCKER)

- **Title mismatch:** JSON title says *"fall 2.7%"* but the PDF headline says *"fall 2.8%"*. Western_European_passenger_car_...
- **Registrations mismatch (minor but still "wrong value" risk):** JSON bullet says **860,319**; the PDF shows **860,319** in the "Implications" line but also **860,318** in the narrative. You should pick one and stay consistent (ideally mirror the headline + main narrative figure). Western_European_passenger_car_...

2) Wrong entities (BLOCKER)

- **government_entities = ["EU"]** is **not explicitly stated** in the visible PDF text. The PDF references "the government" in France/Spain contexts, not "EU" explicitly. This fails "No wrong entities." Western_European_passenger_car_...

3) Geo signal distortion (BLOCKER per your policy)

- **country_mentions includes "United States"** even though the US appears only as **context** ("US tariff conflicts"), not as an operational market being reported with registrations. The checklist says country_mentions should contain **only explicit operational countries.** QUALITY_CHECKLIST Western_European_passenger_car_...
- Because you included US in `country_mentions`, your deterministic derivation also pushes **regions_relevant_to_kiekert = ["Western Europe","US"]**, which becomes a footprint distortion for this record. QUALITY_CHECKLIST Western_European_passenger_car_...

In the previous example, ChatGPT correctly identified the high blockers. When I reviewed the original PDF, it showed values like 2.7% and 2.8%, as well as 860,319 and 860,318—so items 1) and 2) were not actual errors; these discrepancies stemmed from the source file quality. This was the correct behavior. I was surprised to find such issues in the S&P file.

For item 3), the geo signal distortion was fixed and captured during extraction.

"regions_mentioned" : [
0 : "Western Europe"
1 : "US"
]
"regions_relevant_to_kiekert" : [
0 : "Western Europe"
1 : "US"

From source file:

slight growth in 2025 and 1.0% in 2026. The unsettled geopolitical environment is expected to exert a restraining effect on global trade and German economic growth. Rising protectionism, particularly influenced by the US tariff conflicts and structural shifts in global supply chains due to the pandemic and geopolitical tensions, is likely to exacerbate these challenges.

Fix with claude code:

Geo Signal Distortion Fix — Documentation Summary

Issue detected: `regions_mentioned: ["Western Europe", "US"]` for an article about Western European passenger car registrations. The US appeared in the source text only as backdrop context: *"Rising protectionism, particularly influenced by the US tariff conflicts..."* — no US market data was reported.

Root cause chain

```
Source text: "US tariff conflicts"
    ↓
LLM → country_mentions: ["United States"]      ← no extraction rule existed
LLM → regions_mentioned: ["Western Europe", "US"]  ← no extraction rule existed
    ↓
postprocess: derive_regions_relevant_to_kiekert(["United States"]) → ["US"]
    → regions_relevant_to_kiekert = ["US"]   ← wrong
    ↓
postprocess safety-net check _has_explicit_us_signal():
    → scans full raw PDF text → finds "US tariff conflicts" → returns True
    → "US" stays in regions_mentioned            ← wrong
```

Three gaps: no prompt rule for `country_mentions`, no prompt rule for `regions_mentioned`, and the postprocess safety-net confirming US from any mention in the source PDF (including tariff context).

Fix 1 — Extraction prompt rule (rule 10) [`src/model_router.py`]

Added rule 10 to `extraction_prompt()` covering both fields:

> `country_mentions` and `regions_mentioned`: list ONLY countries/regions where the article explicitly reports operational market data (production volumes, vehicle registrations, plant locations, sales, revenue). Do NOT include a country/region when it appears only as geopolitical backdrop, tariff reference, or macro context. Example: a Western European registrations article that mentions 'tariff conflicts' as background — the backdrop country is NOT a country_mention; but if that same article reports sales volumes for a country, that country IS a country_mention and its region IS a `regions_mentioned` entry.

Design note: The example is country-agnostic and two-sided (when to exclude, when to include) to avoid biasing the LLM against any specific country. An article genuinely about the US market — reporting US registration volumes or plant data — will still correctly include United States and US.

Fix 2 — Postprocess safety-net scope [`src/postprocess.py`]

The `_has_explicit_us_signal()` check decides whether to keep or remove "US" from the region merge list. It was scanning `combined_text` — which includes the full raw PDF source text — so "US tariff conflicts" in the body of any document permanently confirmed US as a valid region signal.

Changed the input from the full source text to LLM-extracted record fields only (title, evidence_bullets, key_insights, keywords):

```
# Before — too broad: raw PDF text confirms "US" from any mention
_has_explicit_us_signal(combined_text, rec["country_mentions"])

# After — discriminating: only LLM-structured fields count as confirmation
record_fields_text = _record_text(rec)   # no source_text argument
_has_explicit_us_signal(record_fields_text, rec["country_mentions"])
```

Logic: If the LLM correctly identifies the US as an operational market, it will naturally mention it in the title, evidence bullets, or keywords — and the check returns True, keeping US in regions. If the US appears only as tariff backdrop, the LLM will not mention it in those structured fields — the check returns False, and US is removed. The safety-net still works for legitimate US articles; it just no longer triggers from raw source text noise.

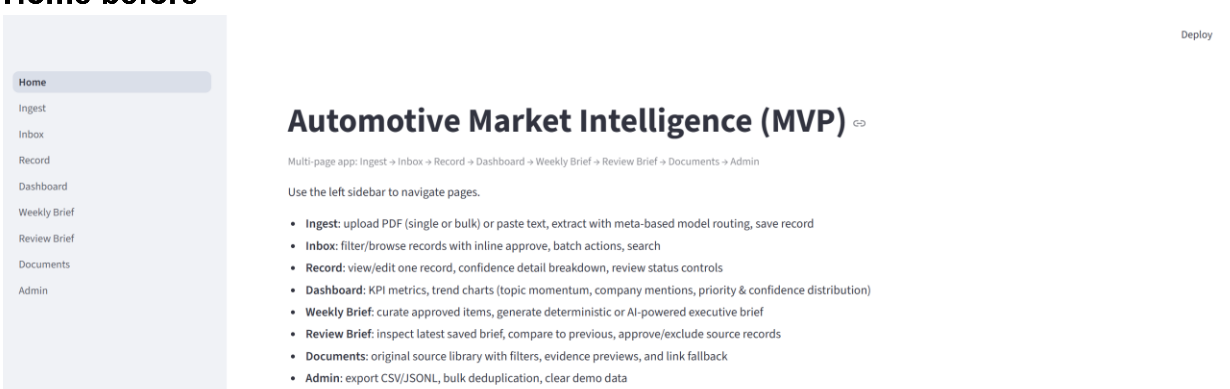
Why two fixes were necessary

Scenario	Fix 1 alone	Fix 2 alone	Both
LLM follows rule, US is tariff-only	✓ US not added	may still confirm US from source text if LLM slipped	✓
LLM slips and adds US despite rule	US added → postprocess can't clean it (text scan confirms)	✓ postprocess removes it	✓
Legitimate US market article	✓ US correctly added	✓ US confirmed via structured fields	✓

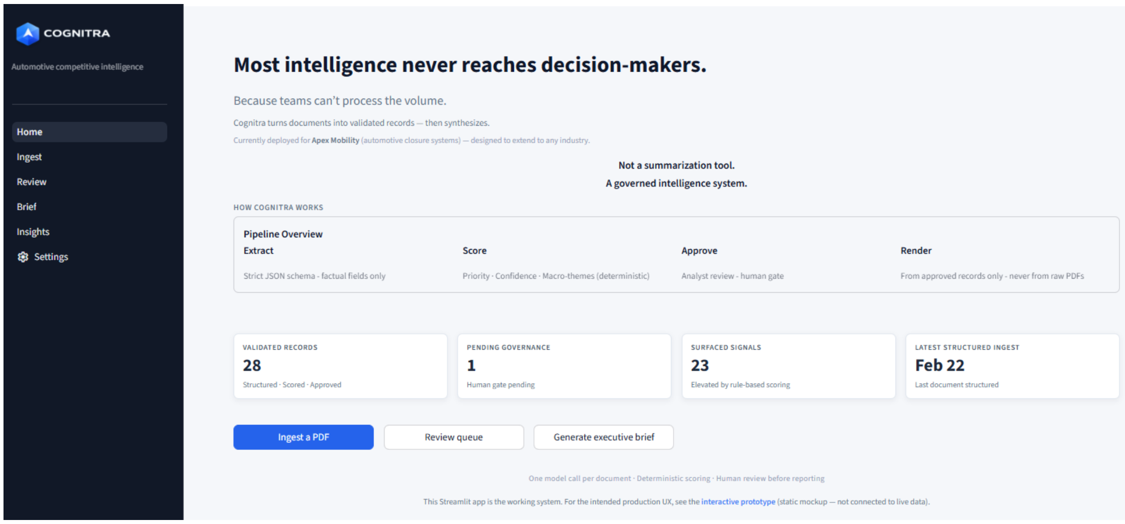
I. UI Before/After Screenshots

Some example of before after UI improvements despite Streamlit limitations.

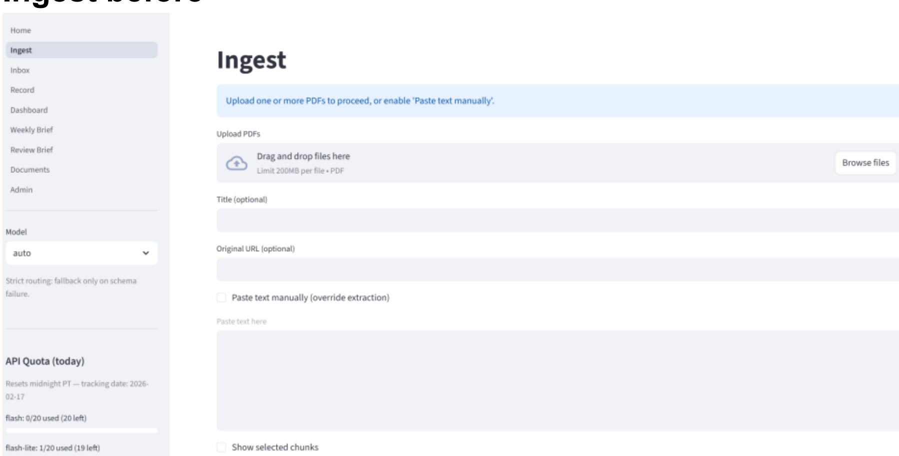
Home before

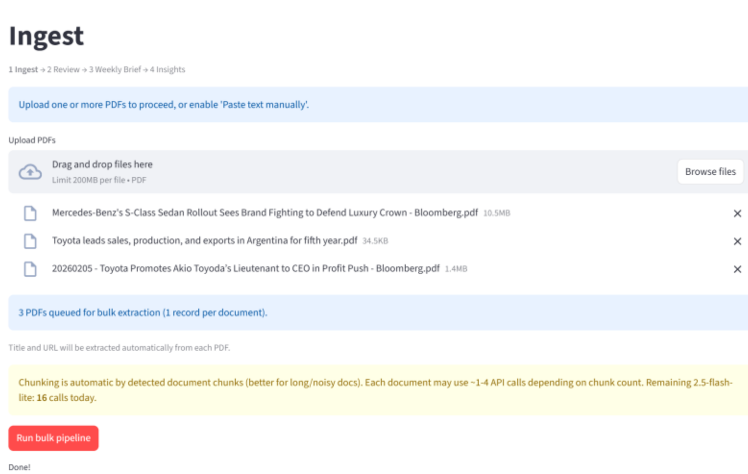


Home after (design for demo)

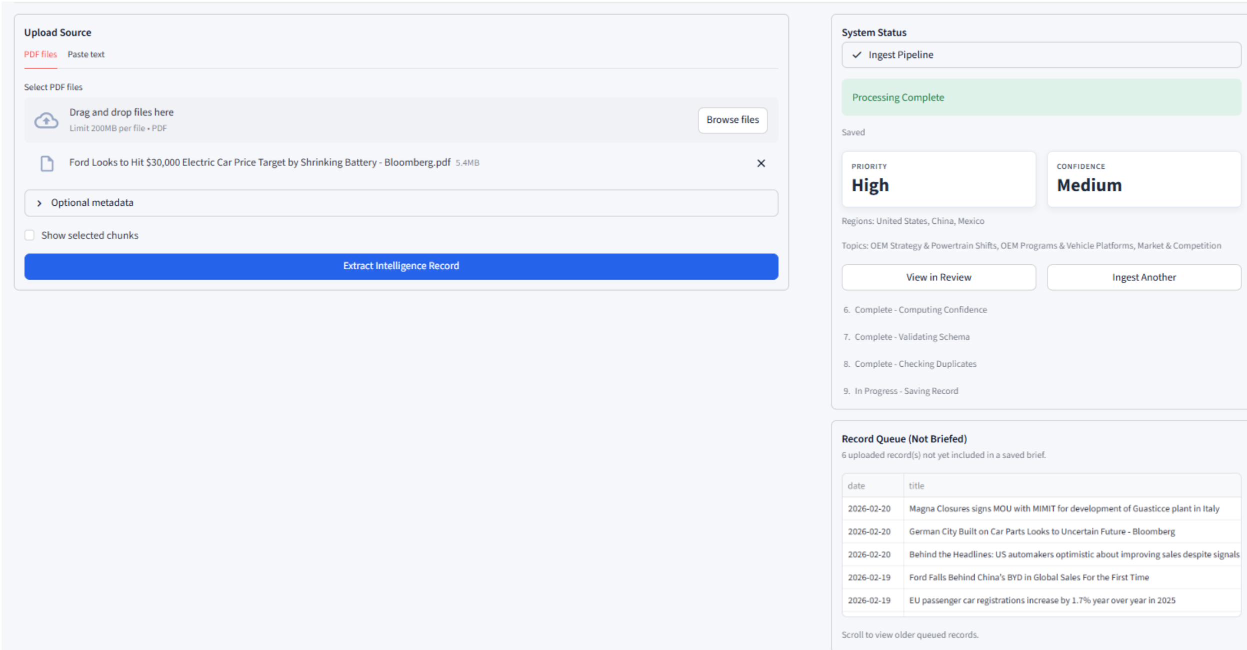


Ingest before



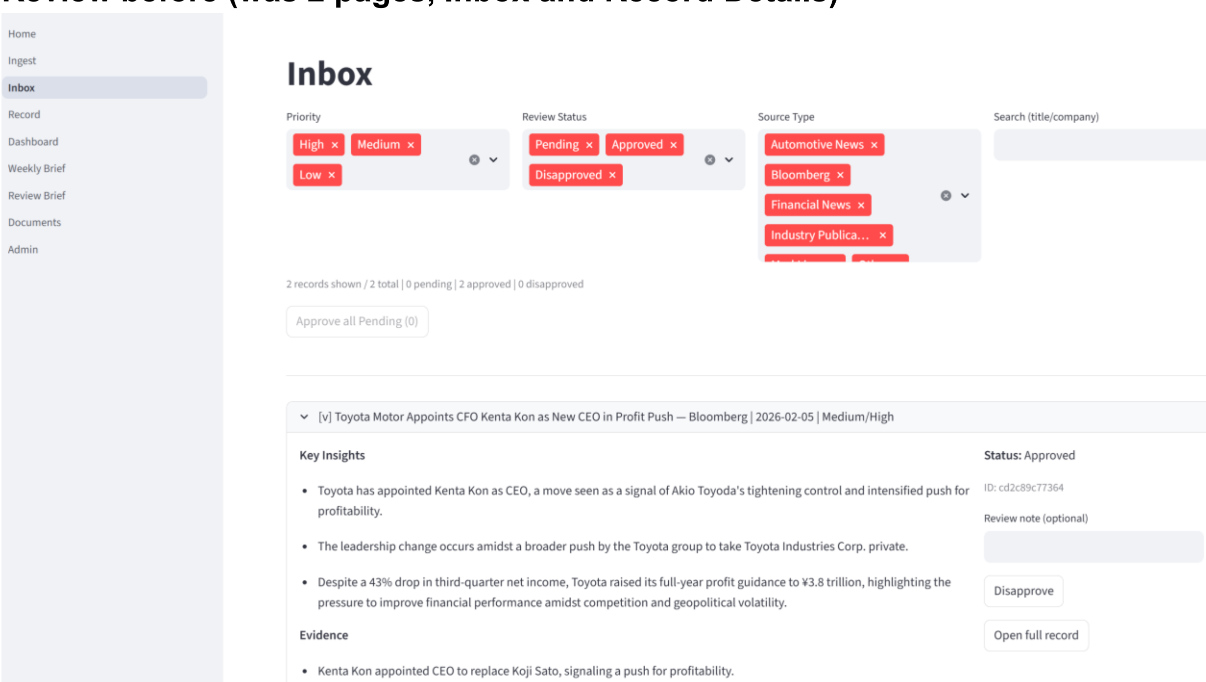


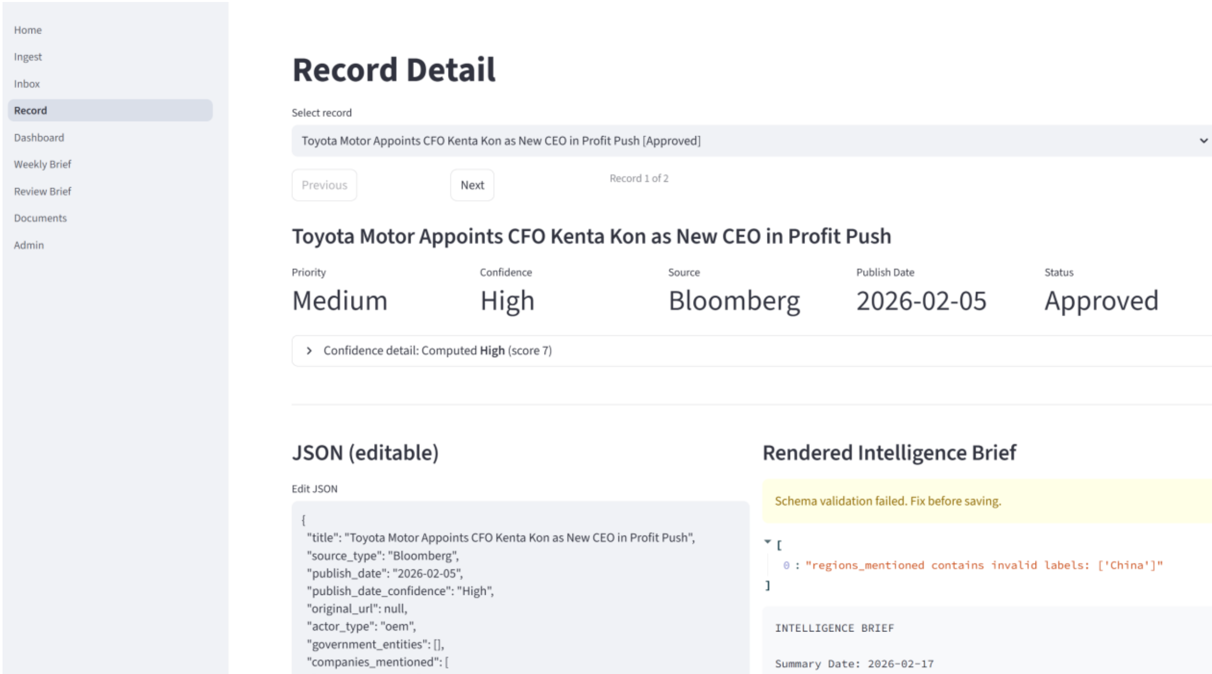
Ingest after



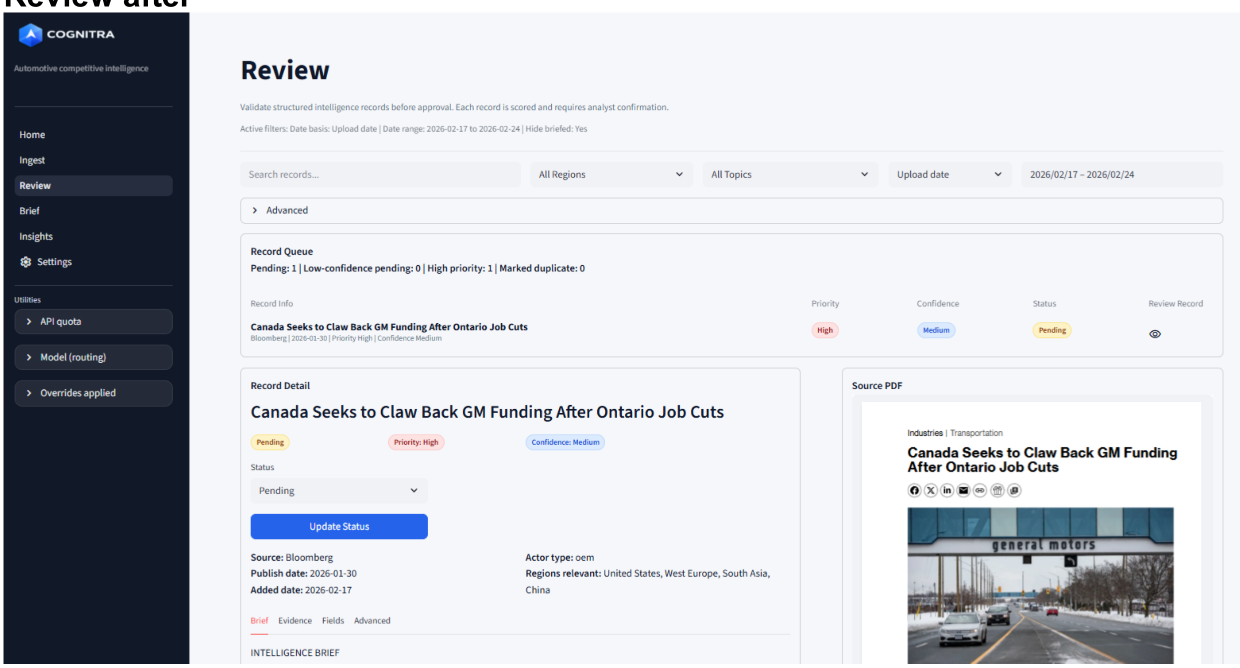
Before: Inbox / Record Detail

Review before (was 2 pages, Inbox and Record Details)

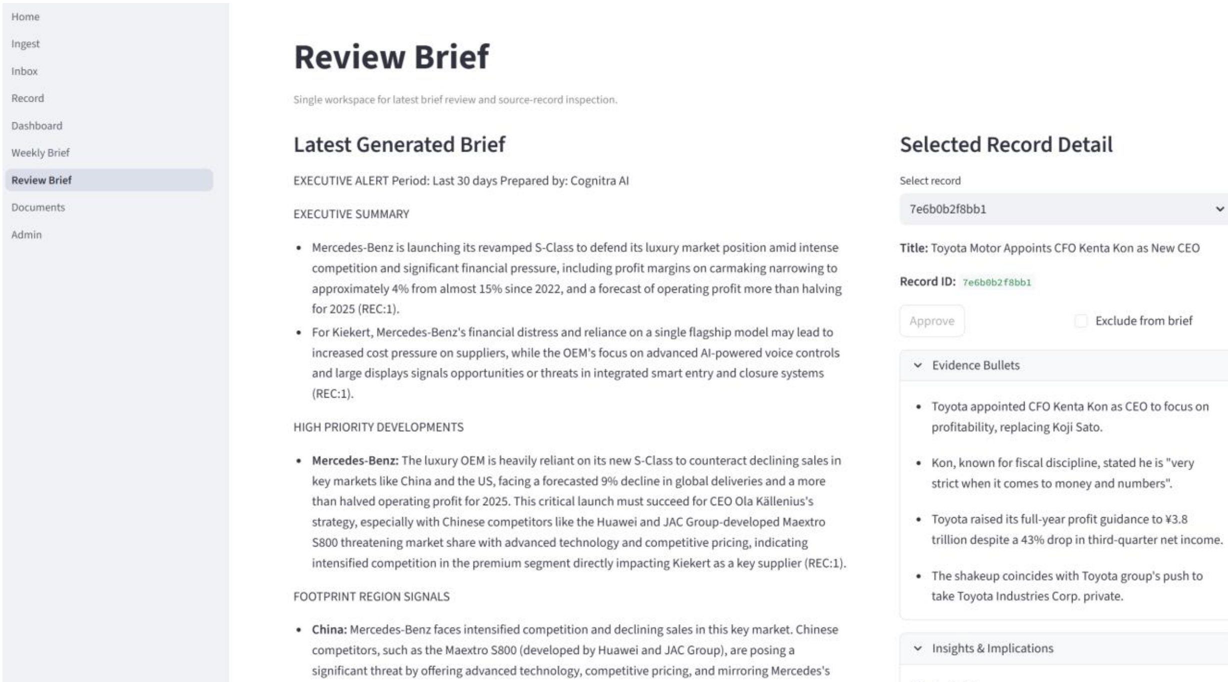




Review after



Brief before



Brief after