Architecture · Technical Paper

Compliance by Construction: Building a Terms-of-Service Risk Engine That Runs Entirely On-Device

Inside an open-source terms-of-service risk engine built to run entirely on local, on-device models, and what it means to make a governance idea operational.

Jennifer McKinney · July 29, 2026 · 7 min read

Terms of service and privacy policies are an awkward input for conventional software. They are long, mutable, ambiguous, and legally consequential. They rarely arrive in a clean schema. They arrive as a URL, a PDF, a DOCX file, an HTML page, or text pasted from a browser. The important clauses are often distributed across documents and written in language that is deliberately broad.

That makes them a tempting use case for a large language model. It also makes them a poor use case for a system that treats the model as an oracle.

I built the open-source **AI Terms & Policies Reviewer** around a different premise: contract and policy analysis should be a structured, reviewable risk workflow, and the documents under review should be able to remain on the operator's device. The project accepts URLs, PDFs, DOCX, RTF, HTML, and pasted text; analyzes the material against a defined risk taxonomy; maps results to jurisdictions and industry profiles; and produces reports in PDF, CSV, or JSON. AI Terms & Policies Reviewer README

The local-first choice is not aesthetic. A vendor contract can contain pricing, customer commitments, business strategy, data-processing terms, incident language, and negotiated exceptions. A privacy policy may be public, but the analysis around it often is not. Sending the full document and an organization's internal risk prompts to a third-party API changes the data boundary before the review has even begun.

Running inference locally creates a simpler boundary: the document, the prompts, the intermediate analysis, and the resulting record can remain in the environment the operator controls. That does not eliminate security work. Local deployment still requires sound endpoint security, access control, retention decisions, and operational discipline. It does mean that a contract-analysis tool does not need to make an external-model provider part of every review.

The implementation uses LocalAI as the local inference endpoint and is configured for two models: Apertus 8B Instruct for multilingual or global material and EuroLLM 22B Instruct for European-language material. The project exposes the model choice through configuration rather than burying it in application logic. That is intentional. Model selection is part of the review context, and it should be inspectable and changeable without rewriting the rest of the system. AI Terms & Policies Reviewer README Apertus 8B Instruct EuroLLM 22B Instruct

The architecture is deliberately ordinary where it should be ordinary. A FastAPI backend handles analysis, ingestion, model integration, and persistence. A Streamlit interface provides the operator-facing workflow. SQLite is the default local data store. There are endpoints for raw-text analysis, URL analysis, and file uploads, as well as endpoints to retrieve analyses, manage a watchlist, and export a report. This is not an attempt to turn a terms reviewer into an agentic system with broad authority. It is a bounded analysis service with explicit inputs and outputs. AI Terms & Policies Reviewer README

The core design question is not “Can an LLM summarize this policy?” It can. The more important question is “What risk categories must remain visible even when the prose changes?”

The system uses a fixed taxonomy for that reason:

Risk category	Review question
Data sharing	Does the document permit third-party sales, broad sharing, or data-broker relationships?
Automated decisions	Does it permit profiling or automated decisions without meaningful human review?
Dark patterns	Does it use deceptive consent language or hide opt-out choices?
Retention	Are data-storage periods indefinite or deletion timelines vague?
User rights	Are access, correction, and deletion mechanisms missing or obscured?
Minors	Are protections for children inadequate or unclear?

Sensitive data	How are biometric, health, or financial data addressed?
Unilateral changes	Can terms change without meaningful notice?
Liability	Do limitations of liability or forced-arbitration terms create disproportionate exposure?

This taxonomy is the first constraint. It prevents the model from deciding, ad hoc, what matters. The model can identify clauses and explain implications in plain language, but it operates within a known set of review dimensions. That distinction is easy to overlook. A generic chat interface produces a fluid conversation. A risk engine produces an accountable artifact.

The artifact needs more than labels. It needs severity, evidence, and a path to human judgment. The current implementation uses a severity-weighted average on a 0 to 10 scale and letter grades. It also documents an Impact, Likelihood, and Safeguards, or IRP, methodology as a planned enhancement rather than presenting a future feature as present functionality. The project's `REVIEW_THRESHOLD` configuration defaults to 0.80 for human review. AI Terms & Policies Reviewer README

That threshold is a small but important design decision. The system should not present all output with the same epistemic weight. A high-confidence, strongly evidenced clause can be surfaced differently from an ambiguous provision, a degraded extraction, or a document that sits at the edge of the available jurisdictional logic. A confidence gate is not a substitute for counsel. It is a way to route uncertainty to the point where judgment belongs.

Jurisdiction mapping is the second major constraint. The reviewer is configured to map analysis across U.S. federal law and selected state contexts including California, Colorado, Connecticut, and New York, plus the EU/GDPR, United Kingdom, Canada, Australia, and Brazil. It also provides industry profiles for retail, finance, health, gaming, social, and education. AI Terms & Policies Reviewer README

This should not be confused with automated legal advice. A term may be low-risk for one use case and unacceptable for another. Jurisdictional mapping does not turn a product requirement into a legal conclusion. It does something more pragmatic: it makes the context explicit. It gives a reviewer a structured place to ask whether a clause about retention, profiling, minors, or data sharing matters differently for a retailer, a health workflow, or a product serving multiple regions.

The watchlist feature extends that logic over time. A single policy review is useful, but terms change. Monitoring a document for changes turns a static due-diligence event into a record of what changed, when it changed, and which risk category needs to be reconsidered. That is the

beginning of provenance. The system is not merely generating a summary. It is maintaining a reviewable history of the policy surface that informed a decision. AI Terms & Policies Reviewer README

This is constraint-native thinking in a concrete system.

The privacy constraint is expressed through local inference rather than a preference statement in a policy. The review constraint is expressed through a defined taxonomy rather than an open-ended prompt. The uncertainty constraint is expressed through a configurable human-review threshold rather than an assumption that the model is correct. The jurisdictional constraint is expressed through explicit mappings rather than an after-the-fact disclaimer. The auditability constraint is expressed through saved analyses, watchlists, and exportable artifacts rather than a transient chat response.

None of those choices makes the tool infallible. The project explicitly states that it does not provide legal advice, replace qualified counsel, guarantee completeness, or make binding legal determinations. Those are not decorative limitations. They define the authority boundary of the system. AI Terms & Policies Reviewer README

That boundary is the point. In compliance work, a useful system should be confident about what it is designed to do and constrained about what it is not authorized to do. It should accelerate triage, preserve the evidence for a decision, and make escalation intentional. It should not manufacture certainty where a qualified human must exercise judgment.

The most interesting part of this project is not that an LLM can read a contract. That is quickly becoming table stakes. The interesting part is whether we can build analysis systems whose privacy, risk model, decision thresholds, jurisdictional context, and authority limits are visible in their construction. That is what makes a governance idea operational.

Sources

- AI Terms & Policies Reviewer README
- Apertus 8B Instruct
- EuroLLM 22B Instruct