From Repository to Regulation: Automated Detection, Risk Classification, and Documentation of AI Systems under the EU AI Act

Ahmed Gharbi
Guardia AI — https://guardia-ai.com

July 2026 — Draft v0.9

Abstract

The EU AI Act (Regulation (EU) 2024/1689) makes organisations legally responsible for AI systems they may not know they operate: obligations attach to what is deployed in production, while the deployed reality of a modern codebase — LLM SDKs, ML frameworks, vector databases, cloud AI services, credentials in configuration files — is rarely captured in any inventory. With high-risk obligations becoming enforceable on 2 August 2026, the first and least-served compliance step is establishing *what AI exists in the stack and what risk tier it occupies*. We present Guardia AI, a pipeline that (1) statically detects AI usage in software repositories across six language ecosystems, including AI usage visible only in configuration files; (2) classifies systems against the Act's risk tiers using a deterministic, citation-bearing rule engine with an evidence-based confidence score; (3) measures group-fairness metrics on customer-supplied model outputs; and (4) generates the Act's required documentation artefacts (Annex IV technical documentation, Article 27 FRIA, Article 17 QMS, Article 72 post-market monitoring plan, Article 50 transparency notices, Article 73 incident records) from detection data rather than blank templates, with continuous re-verification in CI. We validate the approach by applying the full pipeline to our own product and publishing the unedited results, including a defect the scanner surfaced in our own configuration. We discuss the deliberate choice of deterministic methods over LLM-based analysis for the evidentiary path, and the approach's limitations. This paper documents methodology; it is a self-assessment, not a certification, and nothing in it is legal advice.

1 Introduction

Compliance with the EU AI Act presupposes an inventory. Every substantive obligation — risk management (Art. 9), data governance (Art. 10), technical documentation (Art. 11/Annex IV), logging (Art. 12), transparency (Arts. 13, 50), human oversight (Art. 14), quality management (Art. 17), fundamental-rights impact assessment (Art. 27), post-market monitoring (Art. 72), incident reporting (Art. 73) — attaches to *an AI system*, identified and classified. Yet in small and mid-sized software organisations, no such inventory typically exists. AI capability enters codebases incrementally: a developer adds an LLM SDK for a summarisation feature; a data scientist ships a scikit-learn model inside an internal service; an infrastructure engineer configures a hosted-model endpoint in a `docker-compose` file that appears in no dependency manifest at all. Each addition is individually unremarkable; collectively they constitute regulated systems nobody has catalogued — a phenomenon we call *shadow AI* by analogy with shadow IT.

The prevailing remedies are manual: legal consultancies conduct interview-based inventories, and governance/risk/compliance (GRC) platforms administer questionnaires. Both inherit the same weakness — they record what people *believe* is deployed, not what the repository *shows*

is deployed — and both are point-in-time exercises that decay with the next merge.

Our contribution is an engineering treatment of the inventory-and-classification problem:

1. **Detection** (§4): static scanning of repositories for AI usage across the Python, JavaScript/TypeScript, Go, Java/Kotlin, Ruby, and Rust ecosystems, extended with a configuration-file analysis layer that catches AI usage invisible to manifest parsing.
2. **Classification** (§5): a deterministic rule engine mapping declared system properties to the Act's four tiers (prohibited / high-risk / limited / minimal) with per-verdict article citations and a multi-factor confidence score that degrades under ambiguous or inconsistent inputs.
3. **Measurement** (§6): group-fairness metrics (demographic parity, equalized odds) computed on customer-uploaded prediction data via standard estimators — measurement of model behaviour, deliberately not automated inference about it.
4. **Documentation generation and continuity** (§§7–8): the Act's documentary artefacts generated from detection and classification data, and re-verified on every CI run via the same detection engine shipped as a GitHub Action and GitLab CI/CD component.

A design position runs through the system: **the evidentiary path is deterministic**. Detection, classification, and document assembly use signature matching and explicit rules — identical outputs on identical inputs, every finding traceable to a named signature or rule, and reproducible by an auditor. Large language models are used only for clearly-labelled *advisory* text (an assistant, drafting aids) whose output a human reviews and edits; they make no compliance determinations. We argue (§10) that for artefacts whose purpose is to survive regulatory scrutiny, reproducibility beats sophistication.

2 Regulatory background

The EU AI Act entered into force on 1 August 2024 with staggered application: prohibitions and AI-literacy duties (Arts. 4–5) from 2 February 2025; general-purpose model obligations from 2 August 2025; and the main body of high-risk obligations from **2 August 2026**. Penalties reach €35M or 7% of global annual turnover for prohibited practices, and €15M or 3% for most other infringements.

The Act's structure is risk-tiered. **Article 5** prohibits practices outright (e.g., certain social scoring by public authorities; real-time remote biometric identification in public spaces for law enforcement, with narrow exceptions; emotion recognition in workplaces and education). **Annex III** enumerates high-risk areas — biometrics (§1), critical infrastructure (§2), education (§3), employment (§4), essential services including credit and insurance (§5), law enforcement (§6), migration (§7), administration of justice (§8) — that trigger the full obligation set of Chapter III. **Article 50** imposes transparency duties on systems that interact with natural persons or generate synthetic content. Everything else is minimal-risk.

Two properties of this structure matter for automation. First, tier assignment is *predominantly rule-shaped*: it turns on enumerable predicates (sector, whether natural persons are affected, biometric processing, safety-criticality, public-authority status) rather than open-ended judgment, which makes a transparent rule engine feasible and appropriate. Second, the obligations are *documentary*: what an authority or enterprise customer ultimately requests is a set of written artefacts with prescribed content (Annex IV most explicitly). A system that owns the inventory data is positioned to assemble those artefacts mechanically.

3 Related approaches

Legal consultancies produce high-quality bespoke assessments through interviews and document review, at four-to-five-figure engagement costs and multi-week timelines, with no code-level visibility and no continuity between engagements. **GRC platforms** (AI-governance modules of broader compliance suites) organise questionnaires, policies and attestation workflows; their inputs are human declarations, so undeclared AI — precisely the shadow-AI problem — never enters the system. **Static application-security tooling** (SCA/dependency scanners) proves the viability of manifest-level detection but stops at CVE and licence findings; it has no notion of regulatory risk tiers or of AI-specific configuration indicators. Guardia composes the third approach's mechanism with the first two's target: code-derived evidence feeding regulation-shaped outputs. We regard consultancies as complements — several of the generated artefacts are explicitly designed as inputs to legal review — and the honest boundary is stated in §10: no tool determines legal compliance.

4 Detection methodology

4.1 Acquisition

Given a repository URL, the scanner downloads the repository as a single tarball (one API request against the hosting provider) and scans locally. Compared with per-file API traversal this is faster, avoids file-count caps, and stays within rate limits; it also means the scanner sees the repository exactly as a checkout would. In CI deployments (GitHub Action, GitLab CI/CD component) the checkout already exists and the same detection module runs against the working tree.

4.2 Manifest-level signatures

The core detector matches dependency declarations against a curated signature database of AI-relevant packages, each entry carrying *(package → category, AI-Act relevance, note)*. Categories include LLM APIs (e.g., `openai`, `anthropic`, `groq`, `mistralai`), ML frameworks (`torch`, `tensorflow`, `scikit-learn`, `xgboost`), generative-AI libraries (`diffusers`), orchestration frameworks (`langchain`, `llama-index`, the Vercel `ai` SDK), vector databases (`pinecone`, `chromadb`, `qdrant`, `weaviate`), cloud AI services (Azure Cognitive Services, Google Cloud AI, AWS Rekognition/Bedrock clients), and biometric-specific libraries (`deepface`, `face-recognition`) which are flagged directly toward Annex III §1. Manifests parsed span six ecosystems: `requirements.txt`/`pyproject.toml` (Python), `package.json` (JS/TS), `go.mod` (Go), `pom.xml`/`build.gradle` (Java/Kotlin), `Gemfile` (Ruby), and `Cargo.toml` (Rust). Import-level scanning of source files corroborates manifest hits and catches undeclared vendored usage.

Signatures deliberately encode *regulatory* semantics, not just identity: `deepface` is not merely "a computer-vision package" but "facial recognition — likely high-risk under Annex III §1"; an LLM SDK implies *deployer* obligations, while `transformers` or `huggingface_hub` may indicate self-hosted models and therefore potential *provider* obligations. The detection layer thus produces findings already oriented toward the classification step.

4.3 Configuration-file analysis

A second detection layer scans configuration artefacts — `.env` files and examples, `docker-compose.yml`, Terraform, CI YAML, deployment manifests — for three indicator classes: **credential environment keys** (e.g., `OPENAI_API_KEY`, `GROQ_API_KEY`, `ANTHROPIC_API_KEY`), **model identifiers** (e.g., `gpt-4`, `llama-3`, `claude-`), and **provider endpoints** (e.g., `api.openai.com`, `api.groq.com`). This layer exists because AI usage frequently has *no manifest footprint at all*:

a service calling a hosted model through plain HTTP, an endpoint configured in infrastructure code, a credential provisioned for a feature whose library was later removed. Configuration findings are cross-checked against library findings; an AI credential with no corresponding declared dependency is exactly the shadow-AI signature the layer is designed to surface — and, notably, is the class of finding our own self-scan produced (§8).

4.4 False-positive control

Two false-positive classes shaped the detector. First, *lexical collisions*: short signature names (`ai`, `together`) appearing as substrings or in unrelated contexts; the scanner therefore matches structured manifest entries and genuine import statements rather than raw text occurrence. Second — an amusing occupational hazard — *self-reference*: a compliance scanner's own signature database is a list of AI package names, which a naive text scanner flags as massive AI usage. The detector distinguishes signature *definitions* (string constants) from *dependencies* (manifest entries, resolvable imports). Both controls are exercised by the test suite and were validated in the self-scan.

4.5 What static detection cannot see

Detection is static and inventory-shaped. It cannot observe runtime behaviour, distinguish a dead dependency from a load-bearing one, or determine *purpose* — whether the detected `xgboost` model scores loan applicants or sorts log lines. Purpose is precisely what the Act's tiers turn on, which is why detection feeds, but does not replace, the declarative classification step.

5 Risk classification

5.1 Rule engine

Classification takes a structured declaration of an AI system — sector, deployment type, whether natural persons are affected, biometric processing, safety-criticality, emotion recognition, social scoring, public-authority status, and the presence of governance controls (human oversight, technical documentation, audit logging, bias testing, data governance, transparency notices, risk management) — and evaluates rules in strict precedence:

1. **Prohibited (Art. 5).** Explicit predicates for the Act's banned practices, e.g. law-enforcement sector $\wedge$ biometric processing $\wedge$ public-space use $\rightarrow$ Art. 5(1)(d); emotion recognition $\wedge$ (employment $\vee$ education) $\rightarrow$ Art. 5(1)(f); social scoring $\wedge$ public authority $\rightarrow$ Art. 5(1)(c); biometric categorisation of sensitive attributes $\rightarrow$ Art. 5(1)(g). A prohibited verdict short-circuits with a critical finding and mandatory human review.
2. **High-risk (Annex III).** A declared sector within the Annex III mapping (biometrics, critical infrastructure, education, employment, essential services including credit/insurance and healthcare, law enforcement, migration, justice) combined with effect on natural persons; or biometric processing or safety-criticality independently.
3. **Limited (Art. 50).** Systems affecting people that fall outside Annex III, principally interaction-transparency cases.
4. **Minimal**, by exclusion.

Every verdict carries the *text* of the articles it rests on, drawn from an internal citation library, and for high-risk systems the engine emits gap findings keyed to specific articles (missing human oversight $\rightarrow$ Art. 14; missing Annex IV documentation $\rightarrow$ Art. 11; no logging $\rightarrow$ Art. 12; no bias testing or data governance $\rightarrow$ Art. 10; no transparency notice $\rightarrow$ Art. 13; no risk management $\rightarrow$ Art. 9), each with a severity, a remediation, and a "quick win" where one exists. Effort is

estimated from the count of critical/high findings. Prohibited and high-risk verdicts are always flagged for human review.

5.2 Evidence-based confidence

A classification's usefulness depends on the user knowing *how much to trust it*, so the engine computes a confidence score that is a function of evidence rather than a constant. The score composes: **(a) rule strength** — how directly the decisive rule fired (an explicit Art. 5 match scores higher than a tier reached by exclusion); **(b) corroboration** — independent signals agreeing with a high-risk verdict (Annex III sector, biometric data, safety-criticality, sensitive personal data) raise it; **(c) input specificity** — short or missing free-text descriptions lower it, because the verdict then rests on checkbox flags alone; and **(d) cross-signal consistency** — the free-text description is keyword-matched against Annex III areas, and agreement with the declared sector raises confidence while a match against a *different* area lowers it (the sector may be mis-declared). Ambiguity penalties apply near tier boundaries (e.g., autonomous deployment classified only limited/minimal; biometric flags alongside a low-risk verdict). The output is clamped to 50–98: the engine never reports certainty, and never reports less than coin-flip confidence for a verdict it is willing to emit.

This design makes the classifier's epistemic state legible: a 72% limited-risk verdict on a vague description invites exactly the human scrutiny it should.

6 Fairness measurement

For Article 10's bias-evaluation expectations, Guardia computes group-fairness metrics on customer-uploaded prediction data (CSV of predictions, optional ground-truth labels, and a sensitive attribute): demographic parity difference and ratio, and — where labels exist — equalized odds difference and ratio, via the standard `fairlearn` estimators [2], alongside per-group base rates. The engine validates column availability and reports explicit *limitations* with every result (sample-size caveats, absent labels restricting the metric set).

The design choice mirrors §1: this is **measurement, not inference**. Guardia does not train models, does not predict bias, and does not certify absence of bias — our published claim-wording rules forbid "unbiased" outright. It computes standard statistics that a data scientist or auditor can recompute, and places them in the documentation trail.

7 Documentation generation and continuous compliance

7.1 Generated artefacts

Each documentary obligation is a generator that consumes inventory, classification and fairness data: **Annex IV technical documentation** (Art. 11); **FRIA** (Art. 27) as a six-section guided assessment, also offered voluntarily to non-obligated deployers; **QMS documentation** (Art. 17, thirteen sections); **post-market monitoring plan** (Art. 72); **transparency disclosures** (Art. 50); **conformity workflow** with Annex V declaration of conformity (Arts. 43/47/48); **EU database registration package** (Art. 49/Annex VIII); **incident records** with Art. 73 deadline tracking; **AI-literacy role checklists** (Art. 4); and a mapping to **ISO/IEC 42001** controls [4]. Where optional LLM assistance drafts prose, it is labelled as such and editable; the structural and factual content comes from scan data. The value over blank templates is precisely that the documents are *pre-instantiated with evidence*: the Annex IV system description begins from the actual detected inventory, not from an empty heading.

7.2 Continuity

An inventory is an asset only while it is true. The same detection module (kept byte-identical across the GitHub Action and GitLab CI/CD component, ported from the backend scanner) runs on every CI execution, so a newly-introduced AI dependency or configuration indicator surfaces at merge time rather than at audit time. Monitoring alerts, incident deadline tracking, and the post-market monitoring plan close the loop the Act itself prescribes: compliance as an ongoing property of the development process, not a document produced once.

8 Validation by self-application

A compliance tool's most falsifiable claim is what it says about its maker, so we ran the full pipeline on Guardia's own monorepo and published the unedited artefacts — self-scan report, voluntary FRIA, and Annex III self-assessment — at `https://guardia-ai.com/trust` [5].

Findings. The scanner correctly identified the product's true AI surface: the OpenAI-compatible SDK pointed at Groq-hosted Llama models (an advisory chat assistant and a documentation-drafting aid — both labelled, both human-reviewed), scikit-learn/fairlearn as measurement tooling [3], and the corresponding `GROQ_API_KEY` configuration. It correctly did *not* flag the deterministic core (scanner, classifier, generators) as AI, and did not false-positive on its own signature database (§4.4). The classifier returned **limited risk** (Art. 50 transparency tier, 72% confidence, zero gaps) — the transparency duty attaching to the labelled chat assistant — and the published self-assessment explains why no Annex III area applies: Guardia makes no decisions about natural persons.

The scanner found a real defect in its own repository. The configuration layer flagged stale `OPENAI_API_KEY` references (in deployment configs and an environment example) left over from a provider migration, including a health-check endpoint still testing the *wrong* credential — which had been silently reporting a healthy subsystem as degraded. The finding was fixed the same day and retained in the published remediation log. We highlight it because it is the strongest kind of validation available to us pre-pilots: the configuration-analysis layer surfacing genuine shadow-AI residue that manifest-level scanning, and the authors, had missed.

Reproducibility. Anyone with repository access can regenerate the report by running the scanner against the repo root; the detection module itself is public in the CI integrations.

9 Limitations

Static visibility. Detection sees declared dependencies, imports, and configuration — not runtime behaviour, dead code, or dynamically-loaded models. A system whose AI usage leaves no repository trace (e.g., a purely manual process calling a web UI) is invisible to it.

Declaration dependence. Classification inputs describing purpose and context are declared by the user; the engine cross-checks descriptions against declared sectors (§5.2) but cannot detect deliberate misdeclaration. Garbage in produces confidently-cited garbage out, bounded only by the confidence penalties.

Rule coverage. A rule engine encodes the Act's enumerable structure well, but boundary cases (Annex III's exception clauses, the Art. 6(3) derogations, evolving Commission guidance) require legal judgment. This is why prohibited and high-risk verdicts are hard-flagged for human review, and why we position generated artefacts as inputs to counsel, not substitutes for it.

Fairness scope. §6 computes group metrics on supplied data; it does not address individual

fairness, causal claims, or data-collection bias, and its validity is bounded by the representativeness of the uploaded sample.

Self-assessment bias. §8 is validation by the party with the most to gain from a favourable result. We mitigate by publishing complete artefacts including adverse findings, and by making the run reproducible; future pilot deployments are the external test.

Not legal advice. Guardia produces technical evidence and documentation drafts. Compliance is a legal conclusion that only qualified counsel — and ultimately authorities and courts — can draw.

10 Conclusion

The EU AI Act's first practical demand — know what AI you run, and what tier it occupies — is an engineering problem wearing legal clothing, and it responds to engineering methods: signature-based detection with configuration-aware coverage, transparent rules with cited authority and honest confidence, standard fairness estimators, and document generation grounded in detected evidence, all re-verified continuously in CI. The deliberate absence of machine learning from the evidentiary path is, we argue, a feature: an artefact intended to survive scrutiny should be reproducible by the person scrutinising it. Self-application (§8) shows the pipeline finding true positives, avoiding the tempting false ones, and catching a real defect in its own house; future pilot deployments will test it in others'.

References

[1] Regulation (EU) 2024/1689 of the European Parliament and of the Council (Artificial Intelligence Act), OJ L, 12 July 2024.

[2] S. Bird et al., *Fairlearn: A toolkit for assessing and improving fairness in AI*, Microsoft Tech Report MSR-TR-2020-32, 2020.

[3] F. Pedregosa et al., *Scikit-learn: Machine Learning in Python*, JMLR 12, 2011.

[4] ISO/IEC 42001:2023, *Artificial intelligence management system.*

[5] Guardia AI, *Trust & methodology: self-scan report, voluntary FRIA, Annex III self-assessment*, `https://guardia-ai.com/trust`, 2026.