AUDIT-12 — Review-Log

Konvention: Append-only.

Initial-Lauf 2026-05-18

Auditor: R-COMP **Audit-Form:** Code-Inspektion + Surrogat-Rekonstruktions-Übung
Methodik: Acht Test-Vektoren W1–W8 gegen fünf forensische Frage-Klassen **Dauer:** 2 h

Audit-Auftrag

Belegen, dass eine in 2026 erstellte CE-Akte in 2031 von einer Marktüberwachungsbehörde rekonstruierbar ist — mit Beleg der damals verwendeten Modell-Versionen, Lookup-Stand und Doc-Set.

Befunde

- 1 HIGH (Duplikat AUD-8)
- 2 MED (Lookup-Hash + Modell-Provenance-CI)
- 1 LOW (Git-Tag-Konvention)

Bewertung

Stärke: CM-1-Stempel-Architektur ist substantiell, hash-deterministisch, append-only. Wir liegen damit besser als typische Mittelstands-Tools, die gar keinen Konfigurations-Stempel führen.

Schwäche: Lookup-yaml-Snapshot wird nicht im akt_state gespeichert. Bei Lookup-Pflege (PROC-K2) ändern sich die Hashes — eine 2031-Rekonstruktion würde dann auf einem anderen Lookup-Stand laufen. Das ist eine konkrete Reproduktions-Lücke, die mit ~3 h Code-Arbeit (Pipeline-Hook in `aggregiere_alle`) schließbar ist.

Audit-Schlussfolgerung

Die Architektur ist **forensik-fähig**, aber **noch nicht produktiv-vollständig** für eine 10-Jahres-Reproduktion. Die zwei MED-Befunde (F-01, F-03) sollten vor erstem produktiven Pilot-Mandanten geschlossen sein.

Empfehlungen

1. AUDIT-12-F-01 (Lookup-Hash-Snapshot) — ~3 h Code, vor Pilot ziehen
2. AUDIT-12-F-03 (Provenance-CI-Gate) — ~2 h Code, vor Pilot ziehen
3. AUDIT-12-F-02 (Git-Tag-Konvention) — kann mit FG-1-Sprint mitlaufen
4. **Live-Rekonstruktions-Übung jährlich** als Bestandteil von PROC-K4 verankern (statt einmaliger Pilot-Übung)

Nächster Lauf

- **Post-Fix:** nach Schließung F-01 + F-03 (vor erstem Pilot Q3 2026)
- **Jährlicher Folge-Lauf:** 2027-05-18
- **2031 (oder bei Erstanfrage einer Behörde):** Echte Rekonstruktions-Übung mit dem dann aktuellen Stand