

Data Sentinel

Ein selbst gehostetes KI-Gateway, das personenbezogene Daten maskiert, bevor sie ein externes Modell erreichen, und jedes Dokument automatisch nach DSGVO und EU-KI-Verordnung klassifiziert — entwickelt als DACH Enterprise AI Gateway, mit einer kleinen öffentlichen Demo beider Funktionen

Erstellt im September 2026 · Kern-Gateway entwickelt und intern verifiziert; öffentliche Demo entwickelt und in Envoys Live-Oberfläche eingebettet

1. Zusammenfassung

Wofür es gedacht ist. Ein Unternehmen, das KI auf seine eigenen Dokumente anwenden möchte — Verträge, Personalschreiben, Kundendaten — steht vor einem echten Konflikt: Diese Dokumente enthalten oft personenbezogene Daten, und die Übermittlung personenbezogener Daten an einen externen KI-Anbieter kann gegen Datenschutzrecht (DSGVO) oder die eigene Unternehmensrichtlinie verstoßen — insbesondere für Organisationen, die in Deutschland, Österreich oder der Schweiz (der „DACH-Region“) tätig sind, wo die Datenschutzdurchsetzung streng ist. Dahinter verbirgt sich ein zweites, leiseres Problem: Noch bevor KI überhaupt ins Spiel kommt, haben die meisten Unternehmen keine automatische Möglichkeit zu erkennen, in welche regulatorische Kategorie ein Dokument fällt oder wer innerhalb des Unternehmens überhaupt darauf zugreifen darf — und zwar in dem Moment, in dem das Dokument gespeichert wird. Data Sentinel wurde entwickelt, um beide Probleme zu lösen: KI sicher auf sensible Dokumente anzuwenden und automatisch zu wissen, wie ein Dokument klassifiziert und weitergeleitet werden soll, sobald es eintrifft.

Was es ist. Hinter dem Namen „Data Sentinel“ steckt eine umfangreiche Infrastruktur, die Luis entwickelt hat und die er das DACH Enterprise AI Gateway nennt: eine Software, die ein Unternehmen auf seinen eigenen Servern betreibt und die zwischen den eigenen Mitarbeitenden oder Systemen und dem jeweils genutzten KI-Anbieter sitzt. Sie erfüllt zwei unterschiedliche, sich gegenseitig verstärkende Aufgaben. Erstens werden bei jedem durchlaufenden Dokument personenbezogene Informationen automatisch erkannt und maskiert, bevor irgendetwas das Haus verlässt, und nach Eintreffen der KI-Antwort wieder durch die echten Werte ersetzt — sodass Nutzer ein normales, lesbares Ergebnis sehen, während der externe KI-Anbieter die echten Daten zu keinem Zeitpunkt zu Gesicht bekommt. Zweitens kann dieselbe zugrunde liegende Analyse automatisch, sobald ein Dokument

gespeichert wird, eine DSGVO-/EU-KI-Verordnungs-Compliance-Prüfung durchführen — nicht nur als einmaligen Bericht, sondern als Klassifizierung, mit der ein Unternehmen automatisch den richtigen Verarbeitungspfad und die richtigen Zugriffsrechte für dieses Dokument bestimmen kann, nach Abteilung oder Hierarchieebene, bevor eine Person oder ein nachgelagertes KI-System darauf zugreift. Eine kleine, sichere öffentliche Demo beider Konzepte wurde entwickelt, um auf Luis' Portfolio-Website eingebettet zu werden.

Was es tut. In der Praxis: Ein Dokument wird eingegeben, personenbezogene Daten werden erkannt und maskiert, bevor irgendein KI-Modell sie zu sehen bekommt, die Anfrage wird an ein passendes KI-Modell weitergeleitet — ein schnelles Cloud-Modell, ein leistungsfähigeres Modell oder, falls das Unternehmen möchte, dass überhaupt keine Daten das Netzwerk verlassen, ein vollständig lokales Modell —, eine Antwort kommt zurück, die personenbezogenen Daten werden wiederhergestellt, und das Unternehmen erhält eine saubere, nutzbare Antwort mit vollständigem Audit-Trail. Unabhängig davon, und ebenso nützlich, kann dasselbe Dokument in dem Moment, in dem es gespeichert wird, einer automatischen Compliance-Klassifizierung unterzogen werden — das Ergebnis ist eine klare Einschätzung seiner regulatorischen Sensibilität, mit der ein Unternehmen festlegen kann, wer es einsehen darf und wie damit umzugehen ist, auch durch jedes automatisierte oder agentenbasierte System, das es später verarbeitet.

2. Technische Zusammenfassung

Was es technisch tut. Das Gateway sitzt vor jedem ausgehenden LLM-Aufruf. Eingehende Dokumente werden in Chunks zerlegt, durchlaufen eine Presidio-basierte Erkennung und Anonymisierung personenbezogener Daten — einschließlich eigens entwickelter Erkennungsmodule, die auf deutschsprachige Dokumente abgestimmt sind — und werden mittels reversibler, verschlüsselter Tokens maskiert statt durch einfaches Schwärzen, sodass dieselben Werte später wieder entschlüsselt werden können. Jeder Chunk wird anschließend durch eine modellagnostische Proxy-Schicht geleitet, die je nach Konfiguration des Deployments ein Cloud-Modell oder ein vollständig lokales Modell aufrufen kann, optional gestützt durch abgerufenen Kontext aus einer RAG-Pipeline, und die Antwort wird de-anonymisiert, bevor sie zurückgegeben wird. Unabhängig von diesem Ablauf pro Anfrage kann das EU-KI-Verordnungs-Modul des Gateways ein Dokument bereits bei der Aufnahme einer Compliance-Klassifizierung unterziehen — einer strukturierten Einschätzung seiner regulatorischen Kategorie und Sensibilität —, die genau die Eingabe ist, die ein nachgelagertes System benötigt, um automatisch die richtigen Zugriffsrechte und den richtigen Verarbeitungspfad zuzuweisen, statt dass eine Person jedes Dokument manuell sichtet. Jeder Schritt wird auf beiden Pfaden zu Audit-Zwecken nachverfolgt und protokolliert.

Wie es das tut. Das Backend ist ein FastAPI-Dienst, der für die werkzeugnutzende Logik um LangGraph (ein Framework zur Agenten-Orchestrierung) herum aufgebaut ist, mit einem Next.js-Frontend. Ausgehende Modellaufrufe laufen über einen LiteLLM-Proxy, der mit benannten Modell-Aliassen konfiguriert ist — eine schnelle, EU-gehostete Stufe, eine leistungsfähigere Reasoning-Stufe, ein mehrsprachiges Embedding-Modell und ein vollständig lokales Modell für einen Modus ohne jeglichen externen Datenabfluss. Die Maskierung personenbezogener Daten läuft über zwei Presidio-Container (Analyse und Anonymisierung); der Compliance-Klassifizierungspfad wird vom gateway-eigenen EU-KI-Verordnungs-Modul gesteuert, eines von neun Modulen in seiner Registry; n8n übernimmt die Workflow-Automatisierung; Redis stellt das Caching bereit; und im umfangreicheren Deployment-Profil ergänzen Postgres mit pgvector und Langfuse persistente Speicherung und Observability. Welche Funktionen zur Laufzeit tatsächlich verfügbar sind, wird von einer echten Modul-Registry gesteuert — das System prüft, welche optionalen Komponenten tatsächlich konfiguriert sind, und gibt nur die Funktionen an, die wirklich funktionieren.

Wie es entwickelt wurde. Das Gateway wurde ausgehend von einem Architektur-Blueprint in einem strukturierten Prozess entwickelt: sechs Build-Phasen, die verschiedene Teile des Systems parallel umsetzen, gefolgt von einer Härtungsrunde, die Fehler behob, die bei End-to-End-Tests im Live-Betrieb zutage traten. Die öffentliche Demo entstand danach als bewusst separater, deutlich kleinerer Build: Sie übernimmt nur die Konzepte der Maskierung und der Compliance-Prüfung — ein paar Pattern-Dateien wurden von Hand kopiert —, nicht jedoch den Code oder die Laufzeitumgebung des Gateways, und arbeitet mit vier festen, vorab verarbeiteten Beispieldokumenten statt mit Live-Infrastruktur, sodass sie sicher auf einer öffentlichen Portfolio-Website bereitgestellt werden kann.

3. Fachliche Dokumentation

Das adressierte Problem

KI-Anbieter wie Google oder OpenAI sind aus datenschutzrechtlicher Sicht ein Dritter. Übermittelt man ihnen ein Dokument mit dem Namen, der Adresse oder den Vertragsbedingungen eines Kunden, verfügt dieser Dritte damit über personenbezogene Daten — was nach der DSGVO eine Rechtsgrundlage und häufig ausdrückliche Schutzmaßnahmen erfordert und nach der kommenden EU-KI-Verordnung zusätzliche Pflichten für alles mit sich bringt, was einer automatisierten Entscheidungsfindung ähnelt. Viele Unternehmen in der DACH-Region wollen die Produktivitätsvorteile von KI nutzen, können diese Offenlegung aber nicht akzeptieren. Unabhängig davon fehlt den meisten Unternehmen auch eine systematische, automatische Möglichkeit, die Sensibilität eines Dokuments zu klassifizieren und den Zugriff darauf schon bei dessen Eingang korrekt zu steuern — diese Triage erfolgt meist manuell, uneinheitlich oder wird

schlicht übersprungen. Data Sentinel wurde entwickelt, um beide Probleme gemeinsam zu lösen, da ihnen dieselbe zugrunde liegende Analyse des Dokumentinhalts zugrunde liegt.

Funktion 1: Maskierung für KI-Anfragen

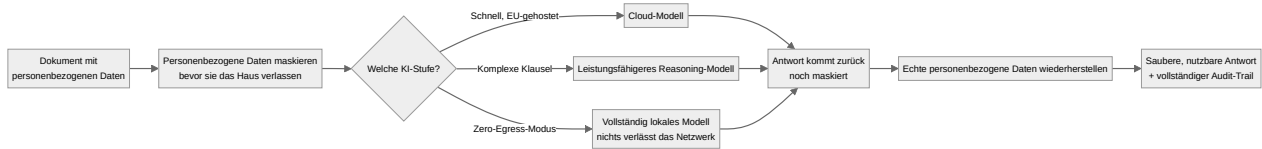


Abb. 1 — Funktion 1: Maskierung, bevor eine KI-Anfrage das Haus verlässt, Wiederherstellung, bevor der Nutzer die Antwort sieht.

Jede Anfrage auf diesem Pfad erzeugt eine Nachverfolgung: welche personenbezogenen Daten erkannt wurden, was tatsächlich an welches Modell gesendet wurde und was zurückkam. Das soll direkt die Frage beantworten, die ein Compliance-Beauftragter oder Auditor irgendwann stellen wird: Was hat dieses Haus verlassen, und an wen. Der Zero-Egress-Modus — die Weiterleitung an ein vollständig lokales Modell statt an einen Cloud-Anbieter — existiert für die strengsten Fälle, in denen selbst maskierte Daten, die das Netzwerk verlassen, inakzeptabel sind.

Funktion 2: Automatische Compliance-Klassifizierung und Zugriffskontrolle

Dies ist die weniger offensichtliche Hälfte des Systems und wohl die insgesamt nützlichere: Dieselbe Analyse, die dem Gateway die Maskierung eines Dokuments ermöglicht, erlaubt es ihm auch, dieses Dokument gegen den aktuellen Text von DSGVO und EU-KI-Verordnung zu lesen und eine strukturierte Compliance-Klassifizierung zu erstellen — automatisch, in dem Moment, in dem das Dokument gespeichert wird, nicht nur auf Anfrage. Diese Klassifizierung ist genau die Eingabe, die ein Unternehmen benötigt, um zwei operative Fragen zu beantworten, die es andernfalls für jedes Dokument von Hand beantworten müsste: Welche Verarbeitung darf dieses Dokument durchlaufen, und wer darf darauf zugreifen. Sobald ein Dokument diese Klassifizierung trägt, können Zugriffs- und Weiterleitungsregeln — nach Abteilung, nach Hierarchieebene, nach Dokumentkategorie — automatisch abgeleitet und angewendet werden, auch durch jedes nachgelagerte agentenbasierte oder automatisierte System, das das Dokument verarbeitet, ohne dass eine Person es zuvor manuell taggen oder freigeben muss.

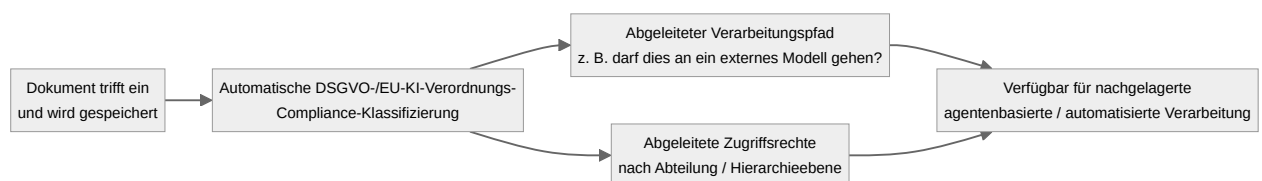


Abb. 2 — Funktion 2: Die Klassifizierung bei Eingang wird zur Grundlage für automatische Weiterleitung und Zugriffskontrolle.

Dies ist aus einem anderen Grund relevant als die Maskierungsfunktion: Es beseitigt die manuelle Compliance-Triage als Engpass, und zwar bereits an dem Punkt, an dem Dokumente in die Systeme eines Unternehmens gelangen, nicht erst dann, wenn jemand zufällig eine KI dazu befragt. Es ist außerdem die Funktion, die die übrigen Ausgaben des Systems innerhalb einer größeren automatisierten Pipeline nutzbar macht — ein nachgelagertes agentenbasiertes System kann sich auf die einem Dokument bereits zugewiesene Klassifizierung verlassen, statt sie erneut herzuleiten.

Ein modulares Modell — nur bezahlen, was genutzt wird

Das Gateway ist um eine echte Modul-Registry herum organisiert, nicht um ein einziges Alles-oder-nichts-Deployment. Das System prüft zur Laufzeit, was tatsächlich konfiguriert und lizenziert ist, und degradiert kontrolliert, statt bei fehlenden Komponenten auszufallen — ein Kunde kann mit der Kernfunktion Maskierung und Routing beginnen und bei wachsendem Bedarf weitere Module hinzufügen, ohne das gesamte System neu auszurollen.

MODUL	WAS ES HINZUFÜGT
Kernanalyse	Dokument-Chunking, Maskierung personenbezogener Daten, Modell-Routing — die Basis
Agentenbasierter Router	Klauselweises Routing zwischen schnellen Modellen und Reasoning-Modellen
RAG-Grounding	Antworten, die auf abgerufenen Präzedenzfällen basieren, nicht nur auf dem Eingabedokument
HITL-Prüfung	Human-in-the-Loop-Prüfpunkte, bevor ein Ergebnis finalisiert wird
EU-KI-Verordnungs-Modul	Automatische Compliance-Klassifizierung pro Dokument nach DSGVO/EU-KI-Verordnung — die Grundlage für automatisch abgeleitete Verarbeitungspfade und Zugriffsrechte
Sovereign-Modus	Vollständig lokales Modell, kein externer Netzwerk-Datenabfluss
Observability	Vollständiges Request-Tracing und Monitoring
Konnektoren	Abruf von Dokumenten aus externen Systemen (z. B. einem Freigabeordner, Ticketsystem)

Die Rolle der öffentlichen Demo

Das interne Gateway selbst ist nicht dafür gedacht, auf einer persönlichen Website offengelegt zu werden — es besitzt bewusst keine Authentifizierungsschicht, da es hinter dem privaten Netzwerk eines Unternehmens laufen soll. Die öffentliche Demo existiert ausschließlich, damit Besucher des Portfolios beide Funktionen sicher erleben können, anhand von vier festen Beispieldokumenten (zwei auf Englisch, zwei auf Deutsch): die maskierte und wiederhergestellte Version jedes Dokuments (Funktion 1) sowie eine unabhängige Compliance-Prüfung jedes Dokuments mit Verweis auf konkrete Artikel der DSGVO/EU-KI-Verordnung (Funktion 2) — bewusst als zwei gleichwertig reale, gleichwertig demonstrierte Konzepte dargestellt, nicht als eine Hauptfunktion mit einer Compliance-Fußnote.

Integrationsstatus: Envoys eigener Frontend-Quellcode bestätigt, dass diese Demo eingebunden ist — nicht nur fertiggestellt und bereit —: Sie wird von Envoys Server unter `/data-sentinel/*` per Reverse-Proxy bereitgestellt und über ein `iframe` auf diesem Same-Origin-Pfad eingebettet, ohne dass ein separater öffentlicher Hostname oder eine Cloudflare-Tunnel-Route erforderlich wäre. Sie läuft als eine wirklich eigenständige Anwendung (eigener Container, eigener Next.js-Build), die keine Laufzeitumgebung mit dem internen Gateway teilt, und wird ausschließlich über Envoys eigene Oberfläche erreicht.

4. Technische Dokumentation

Gateway-Architektur

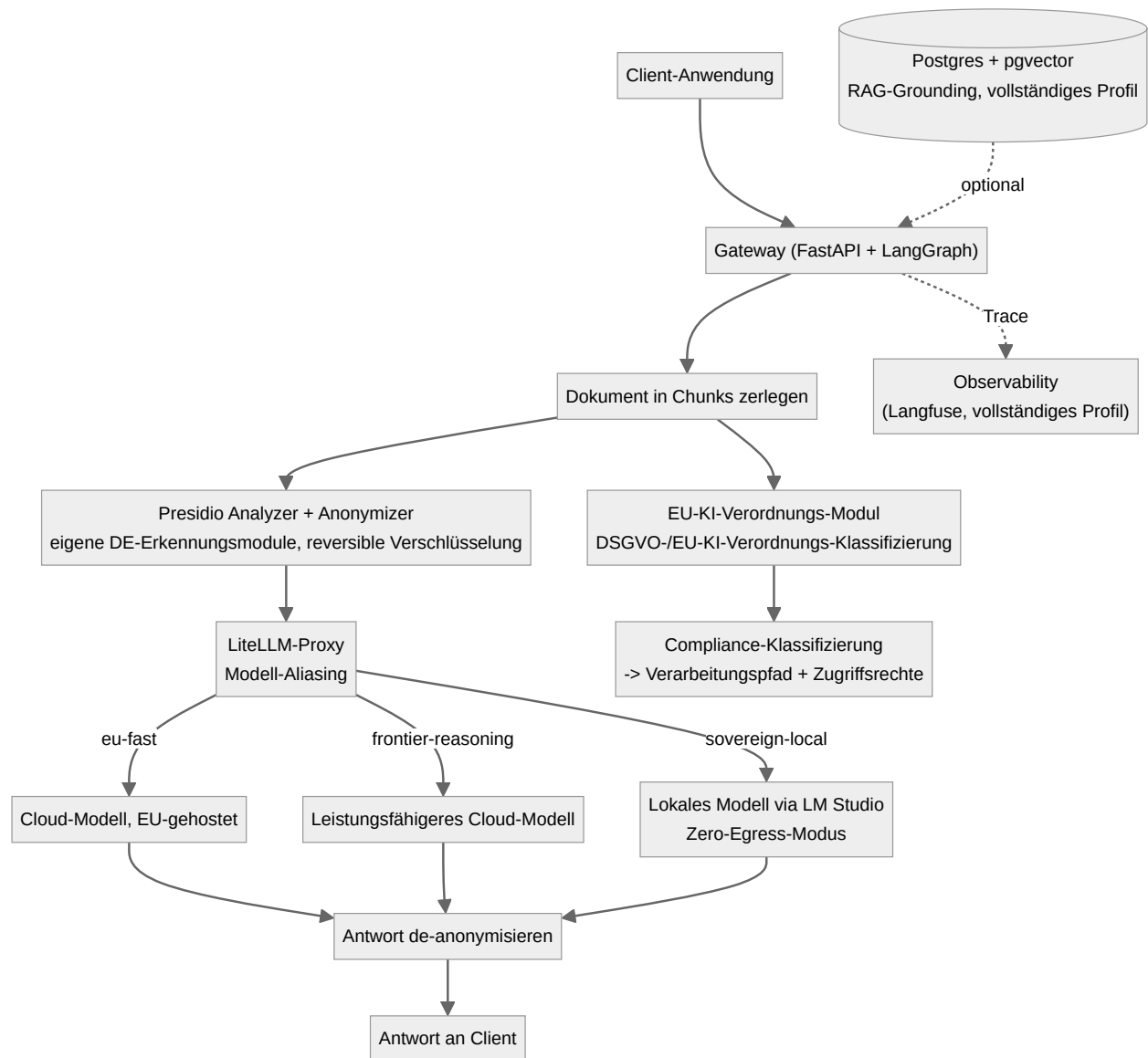


Abb. 3 — Gateway-Request-Flow: Maskierung und Routing auf dem einen Pfad, Compliance-Klassifizierung auf dem anderen, beide gespeist von derselben Dokumentanalyse.

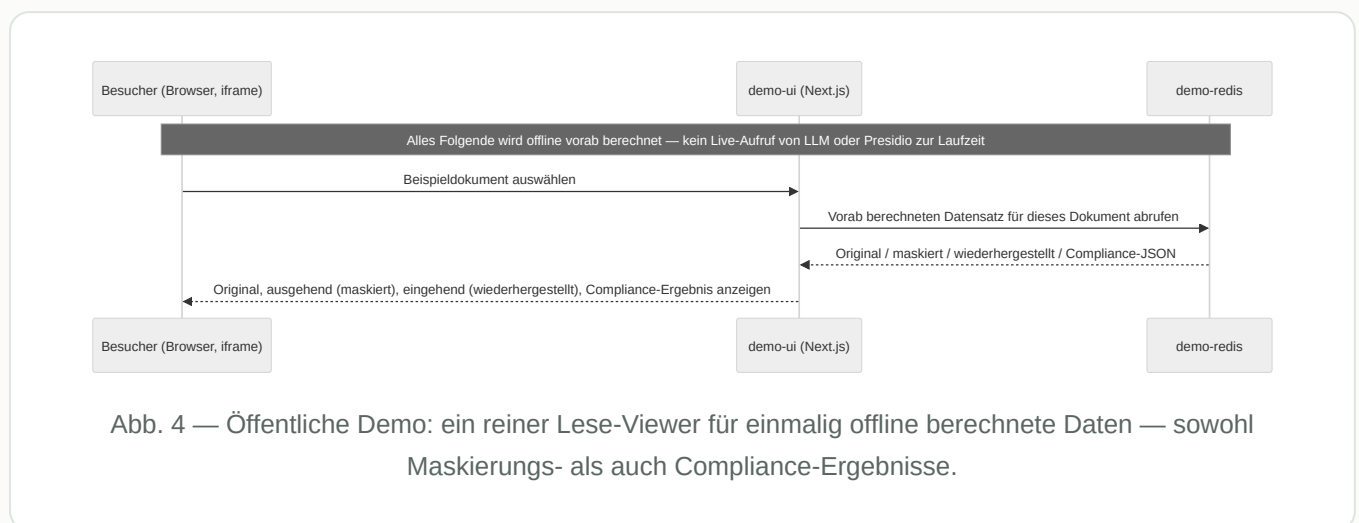
Bestätigte Service-Topologie

DIENST	ROLLE
Gateway (FastAPI + LangGraph)	Kernorchestrierung — Chunking, werkzeugnutzender Agent, Routing-Logik, Modul-Registry (einschließlich des EU-KI-Verordnungs-Klassifizierungsmoduls)
Presidio Analyzer + Anonymizer	Erkennung personenbezogener Daten und reversible Maskierung, eigene deutsche Erkennungsmodule
LiteLLM-Proxy	Modellagnostisches Routing, benannte Aliasse pro Stufe
n8n	Workflow-Automatisierung

Redis	Caching
Next.js-Frontend	Nutzerseitige Oberfläche
Postgres + pgvector, Langfuse (vollständiges Profil)	Persistente Speicherung und Observability, optionaler Deployment- Overlay

Das Basis-Deployment umfasst sechs Backend-Dienste plus Frontend; ein umfangreicheres Profil legt zusätzlich einen Datenbank- und Observability-Stack darüber. Die Modul-Registry gleicht ein deklariertes Modul-Manifest damit ab, welche optionalen Abhängigkeiten zur Laufzeit tatsächlich konfiguriert sind, und gibt das Ergebnis nach außen — sodass das System nur Funktionen für sich beansprucht, die tatsächlich aktiv sind, einschließlich des EU-KI-Verordnungs-Moduls.

Architektur der öffentlichen Demo



Die Demo ist bewusst eine separate, minimale Codebasis: zwei Container (eine passwortgeschützte Redis-Instanz ohne veröffentlichten Host-Port sowie die Next.js-App), die vier vorab berechnete JSON-Payloads einlesen, die einmalig offline von einem eigenständigen Precompute-Skript erzeugt wurden, das nie in das ausgelieferte Image eingebaut wird. Es gibt keine Texteingabe, die zur Laufzeit ein LLM oder Presidio erreicht — jedes sichtbare Ergebnis, sowohl Maskierung als auch Compliance, wurde im Voraus aus einem festen, handverlesenen Dokumentensatz erzeugt. Dadurch werden die Missbrauchsszenarien, gegen die sich eine öffentlich zugängliche Demo sonst absichern müsste, strukturell ausgeschlossen, statt sich auf Filterung zur Laufzeit zu verlassen, um sie abzufangen.

Die Einbettung ist auf ein iframe ausgelegt: Die Demo meldet ihre eigene Inhaltshöhe über `postMessage` an die übergeordnete Seite, sodass die einbettende Website den Frame responsiv anpassen kann, und eine Content-Security-Policy-Einstellung schränkt ein, welche Ursprünge (Origins) sie überhaupt einbetten dürfen — ein standardmäßig geschlossenes Tor statt eines offenen.

Verifizierungsstand

Die Dokumentation des Gateways selbst legt offen dar, wo der aktuelle Stand der „live verifizierten“ Nachweise steht: Eine frühere Runde von End-to-End-Tests wurde gegen eine vorherige Container-Image-Konfiguration durchgeführt. Seitdem wurden die Presidio- und LiteLLM-Dienste auf eigene, schlankere Container-Builds umgestellt, und diese Umstellung wurde noch nicht erneut end-to-end gegen dieselben Tests verifiziert. Dies ist eine offengelegte, selbst benannte Lücke in den eigenen Build-Notizen des Projekts, keine Diskrepanz, die diese Überprüfung eigenständig aufgedeckt hätte — es lohnt sich, dies in jeder technischen Ausarbeitung als „zuletzt vollständig gegen einen vorherigen Image-Stand verifiziert“ festzuhalten.

Ein Quellcode-Bundle des Gateways wurde am 07.09.2026 erstellt, gedacht zum Entpacken und Neubauen auf einem Deployment-Host (wobei echte Secrets, Abhängigkeiten und ein Corpus-Abruf diesem Host überlassen bleiben) — ein Beleg dafür, dass das Projekt um dieses Datum herum für eine Übergabe oder ein Deployment vorbereitet wurde, kein Beleg für eine bereits laufende öffentliche Instanz.

5. Demo

Zusammenfassung

Da das eigentliche Data-Sentinel-Gateway privat innerhalb des Netzwerks eines Unternehmens laufen soll — es besitzt bewusst keinen Login-Bildschirm, da es nicht für das öffentliche Internet gedacht ist —, hat Luis eine separate, deutlich kleinere öffentliche Demo gebaut, mit der Besucher des Portfolios beide Kernideen in Aktion sehen können, ohne dass echte Infrastruktur offengelegt wird. Besucher wählen eines von vier Beispieldokumenten (zwei auf Englisch, zwei auf Deutsch) aus und können Schritt für Schritt nachvollziehen, was damit geschieht: das Original, die Version mit maskierten personenbezogenen Daten, wie sie das Haus verlassen würde, eine simulierte KI-Antwort mit wiederhergestellten personenbezogenen Daten und — gleichwertig — eine unabhängige Compliance-Prüfung des Dokuments nach DSGVO- und EU-KI-Verordnungs-Regeln, alles reale Inhalte, nichts davon zu Effekthascherei erfunden.

Technische Zusammenfassung

Die Demo ist bewusst ein minimaler, in sich geschlossener Build: eine Next.js-Anwendung und ein passwortgeschützter Redis-Cache, sonst nichts. Alles, was Besucher sehen, wurde einmalig offline berechnet, von einem eigenständigen Skript, das die echte Presidio-Maskierungs-/Wiederherstellungs-Pipeline sowie die Compliance-Klassifizierungslogik gegen die vier festen Dokumente ausführt und die Ergebnisse in JSON schreibt; dieses Skript und seine Abhängigkeiten (Presidio, spaCy, ein etwaiger LLM-Schlüssel) sind nie im ausgelieferten Image

enthalten, sodass die dauerhaft laufenden öffentlichen Container zur Laufzeit nie ein KI-Modell oder eine PII-Engine aufrufen — sie liefern lediglich vorab berechnete Ergebnisse aus Redis aus.

Fachlicher und technischer Umfang

Die Demo existiert, um ein konkretes Vertrauensproblem für zwei Behauptungen zugleich zu lösen: „Personenbezogene Daten werden maskiert, bevor sie Ihre Umgebung verlassen“ und „Dieses Dokument wurde gegen die tatsächlich geltende, aktuelle Regulierung geprüft, nicht geraten.“ Beide Behauptungen sind leicht aufzustellen und allein aus einem Textabsatz schwer glaubwürdig zu machen. Dass Besucher das tatsächliche Vorher/Nachher an einem echten Dokument durchklicken und einen Compliance-Bericht lesen können, der konkrete Artikel der DSGVO/EU-KI-Verordnung zitiert, macht beide Mechanismen konkret erfahrbar statt bloß behauptet — für ein Publikum aus Recruitern, technischen Prüfern und potenziellen Kunden, die das zugrunde liegende Gateway bewerten und gute Gründe haben, unverifizierten Aussagen zum Umgang mit Daten skeptisch gegenüberzustehen.

Technisch gesehen teilt sich das Projekt keine Laufzeitumgebung und keinen Code-Pfad mit dem internen Gateway — lediglich ein paar Pattern-Dateien für die Erkennung personenbezogener Daten wurden von Hand kopiert und in der Datei, die sie verwendet, entsprechend gekennzeichnet; das ist die gesamte Verbindung. Die Einbettung erfolgt über ein iframe in Envoy, direkt bestätigt durch Envoys eigenen Frontend-Quellcode: Envoys Server leitet `/data-sentinel/*` per Reverse-Proxy an den Container dieser App weiter und bindet sie über ein iframe ein, das auf diesen Same-Origin-Pfad zeigt, sodass kein separater öffentlicher Hostname oder eine Cloudflare-Tunnel-Route nötig ist. Sie meldet ihre gerenderte Höhe automatisch an die übergeordnete Seite, sodass sich das iframe selbst in der Größe anpasst, und eine Content-Security-Policy-Einstellung schränkt ein, welche Ursprünge sie überhaupt einbetten dürfen — standardmäßig geschlossen, bis der Ursprung der echten Website ausdrücklich eingetragen ist. Die Container der Demo, die Redis-gestützten API-Routen und die vorab berechneten Ausgaben aller vier Dokumente — sowohl Maskierung als auch Compliance — sind vollständig, in sich konsistent und in Envoys Live-Oberfläche eingebunden, keine ausstehende Integration.