

Envoy

Die KI-gestützte persönliche Website hinter luism.pro — ein von Grund auf entwickeltes System, das agentische Orchestrierung, Retrieval-Augmented Generation (RAG) und produktionsreife KI-Engineering-Praxis nicht nur beschreibt, sondern demonstriert

Erstellt im September 2026 · Live-System, aktiv im Einsatz

1. Zusammenfassung

Wozu es dient. Eine Lebenslauf- oder Portfolio-Website ist normalerweise eine statische Seite: derselbe Text, in derselben Reihenfolge, für jeden Besucher — egal ob er nach Stichworten sucht oder nach architektonischen Details. Envoy soll genau diese Lücke schließen: Statt jeden Besucher zu zwingen, sich durch dasselbe feste Layout zu graben, kann er einfach in eigenen Worten fragen, was er wissen möchte, und erhält eine Antwort, die auf der tatsächlichen Frage basiert statt auf einer Einheitsseite für alle. Darüber hinaus ist Envoy bewusst so gebaut, dass es nicht nur genutzt, sondern auch betrachtet werden kann: Dieselbe Technik — Retrieval, Orchestrierung, Schutzmechanismen (Guardrails), Monitoring, geordnete Fehlerbehandlung —, die die Frage eines Besuchers beantwortet, ist auch zur Einsicht offengelegt, sodass ein Besucher, der Luis' technische Fähigkeiten beurteilt, sie bei der Arbeit beobachten kann, statt einer Beschreibung davon vertrauen zu müssen.

Was es ist. Envoy ist Luis' persönliche Website unter luism.pro, aufgebaut in zwei Schichten statt einer einzigen Seite. Darunter liegt eine klassische statische Website — Profil, CV-Download, Portfolio —, die unabhängig auf ihrer eigenen Cloud-Infrastruktur gehostet wird, sodass ihre Verfügbarkeit von nichts anderem abhängt. Darüber liegt, wenn alles funktionsfähig ist, eine konversationelle KI-Schicht, die zur Standardmethode wird, mit der Besucher die Seite erkunden: Statt eines Menüs lädt eine Chatbox zu Fragen ein, und der Inhalt der Seite ordnet sich um das Gespräch herum neu an. Jeder Besucher kann einen „Technische Details anzeigen“-Schalter selbst aktivieren, um die Mechanik hinter jeder Antwort zu sehen — das ist die einzige Unterscheidung, die die Plattform trifft; sie erkennt oder routet Besucher nicht nach Typ. Anders als Luis' übrige Portfolio-Projekte, die jeweils Tools und Integrationen sind, gebaut, um ein einzelnes Problem zu lösen, ist Envoy selbst die Demonstration: ein System, das gezielt gebaut wurde, um technische Praxis zu zeigen, mit Data Express und Data Sentinel direkt darin eingebettet als funktionierendem Beweis.

Was es tut. Ein Besucher landet auf der Website, wird begrüßt und kann nach Luis' Werdegang, Arbeitsstil oder einem bestimmten Projekt in seinem Portfolio fragen; der Assistent antwortet ausschließlich auf Basis geprüfter Fakten über ihn. Ein Besucher kann außerdem eine Stellenbeschreibung einfügen und eine strukturierte Abgleichsanalyse gegen Luis' tatsächliches Profil erhalten, die Data-Express- und Data-Sentinel-Demos direkt innerhalb von Envoys eigener Oberfläche

öffnen und ausprobieren, den Lebenslauf als Download anfordern oder eine Nachricht senden, die Luis per E-Mail erreicht — während die Website nebenbei anonymisierte Nutzungsdaten erfasst, damit Luis sehen kann, was Besucher tatsächlich interessiert hat.

2. Technische Zusammenfassung

Was es technisch tut. Envoy ist ein Retrieval-Augmented-Generation-System (RAG), das über ein Standard-Web-Frontend gelegt ist. Jede Nachricht eines Besuchers löst eine Ähnlichkeitssuche über eine kuratierte, handgeschriebene Wissensdatenbank über Luis aus (Karriere, Fähigkeiten, Projekte, das System selbst), und nur die Fakten, die diese Suche liefert, werden zusammen mit der Frage des Besuchers an ein Sprachmodell übergeben — das Modell wird nie gebeten, aus allgemeinem Wissen oder aus dem Gedächtnis zu antworten, was seine Antworten faktisch verankert und das Risiko verringert, dass es sich etwas über Luis ausdenkt, das nicht stimmt.

Wie es das tut. Die dynamische Schicht läuft als React/Next.js-Anwendung, die jede dynamische Anfrage (Chat, Job-Abgleich, Kontakt) an ein n8n-Automatisierungs-Backend weiterleitet, wobei der gesamte dynamische Stack derzeit während der aktiven Entwicklung in Docker auf Luis' eigener Infrastruktur läuft und nur über einen Cloudflare Tunnel erreichbar ist — es sind keine Ports zum öffentlichen Internet geöffnet. Die statische Website wird bewusst nicht von derselben Infrastruktur ausgeliefert: Sie wird unabhängig auf ihrem eigenen Cloud-Host gehostet, sodass ein Ausfall der Heiminfrastruktur nur die KI-Schicht beeinträchtigt und niemals die gesamte Website lahmlegt. Auf der dynamischen Seite sortiert n8n jede Nachricht zunächst dem passenden Spezial-Workflow zu (über Luis, über das System selbst, über das Portfolio, Job-Abgleich oder Kontakt), und dieser Spezialist bettet die Anfrage mit einem Gemini-Embedding-Modell ein, durchsucht einen Postgres/pgvector-Index nach passenden Wissensdatenbank-Abschnitten und erzeugt die Antwort mit einem Gemini-Sprachmodell. Redis übernimmt das Rate-Limiting des Chats und merkt sich die Sprache eines Besuchers; MinIO speichert die RAG-Wissensdatenbank-Dokumente selbst; Uptime Kuma überwacht jeden Dienst.

Wie es gebaut wurde. Die Retrieval-Logik durchlief mehrere Optimierungsrunden, nachdem frühe Tests eine zu geringe oder zu starke Trefferquote zeigten: Der Ähnlichkeits-Schwellenwert wurde neu kalibriert, ein Re-Ranking-Schritt mit Prioritäts-/Schlagwort-Gewichtung wurde zusätzlich zur reinen Vektor-Ähnlichkeit eingeführt, und ein Schritt zur Erweiterung um verwandte Abschnitte wurde eingeführt, damit eine Antwort auch angrenzenden Kontext einbeziehen kann, den die reine Suche übersehen hat. Die gesamte Orchestrierung — Routing, Retrieval, Generierung, Guardrails — liegt in visuell gebauten n8n-Workflows statt in handgeschriebenen Backend-Diensten. Jede dieser Design-Entscheidungen wurde so getroffen, dass sie sowohl funktional als auch nachvollziehbar ist: Der eigene „Über dieses System“-Spezialist der Website und der Schalter für technische Details existieren gezielt dafür, dass ein Besucher einer Behauptung über die Funktionsweise nicht vertrauen muss — er kann direkt danach fragen oder sie beobachten.

3. Fachliche Dokumentation

Das adressierte Problem

Eine statische Lebenslaufseite verlangt von jedem Besucher, selbst herauszufinden, was für ihn relevant ist. Recruiter überfliegen; technische Prüfer graben tief; ein Personalverantwortlicher, der die Eignung für eine bestimmte Rolle beurteilt, will eine direkte Antwort, kein Dokument zum Interpretieren. Eine klassische Website kann sich dem nicht anpassen — sie zeigt denselben Inhalt, unabhängig davon, wer liest oder weshalb.

Mehr als adaptiver Inhalt — eine technische Leistungsschau

Envoy unterscheidet sich bewusst vom Rest dieses Portfolios. Data Express und Data Sentinel sind Tools und Integrationen, jeweils gebaut, um ein Problem gut zu lösen. Envoy ist ein System, von Grund auf gezielt so gebaut, dass ein Besucher — insbesondere einer, der Luis für eine Rolle als AI Automation Engineer oder Architect beurteilt — nicht nur über seine technische Praxis liest, sondern sie direkt in Aktion beobachten kann: agentische Orchestrierung, Retrieval-Augmented Generation, Sicherheits-Guardrails, Sicherheits- und Latenz-Monitoring, Nachvollziehbarkeit von Informationen, Systemintegration, Selbstreferenz und geordnete Leistungsminderung. Jedes dieser Elemente ist eine bewusste Demonstration, und jedes davon entspricht einem Wert, auf den Luis bei jedem Projekt hinarbeitet:

TECHNISCHES ELEMENT	DEMONSTRIERTE WERTE
Agentische Orchestrierung — Guardrail, Triage und Spezialisten-Routing in n8n	Effizienz, Relevanz, Ergebnisoptimierung
Retrieval-Augmented Generation, verankert in einer kuratierten Wissensdatenbank	Relevanz, Prüfbarkeit, Nachvollziehbarkeit
Sicherheits-Guardrails — keine offenen Ports, ausschließlich Cloudflare Tunnel, Rate-Limiting	Sicherheit
Sicherheits- und Latenz-Monitoring — jeder Dienst überwacht, jede Interaktion zeitlich erfasst	Zuverlässigkeit, Transparenz
Nachvollziehbarkeit von Informationen — ein Live-Trace pro Interaktion mit Route, Guardrail-Ergebnis, Latenz und verwendeten Modellen	Nachvollziehbarkeit, Transparenz, Prüfbarkeit
Systemintegration — Data Express und Data Sentinel direkt in Envoy eingebettet, nicht nur verlinkt	Integration, Nutzererlebnis
Selbstreferenz — Envoy kann seine eigenen echten Workflows und RAG-Dokumente zeigen, keine generierte Zusammenfassung	Transparenz, Prüfbarkeit, Nutzererlebnis
Geordnete Leistungsminderung — die statische Website funktioniert weiter, wenn etwas Dynamisches ausfällt	Zuverlässigkeit, Nutzererlebnis
Kostenbewusstsein — keine API-Kosten durch Bots/Crawler, ratenbegrenzter Chat, gestufter Modelleinsatz	Kostenbewusstsein, Effizienz

Eine einzige Unterscheidung, vom Besucher gewählt

Jeder Besucher spricht mit demselben Assistenten, in seinen eigenen Worten — die Plattform erkennt oder routet Besucher nicht nach Typ (etwa Recruiter vs. technisch versiert). Die einzige einstellbare Option ist ein „Technische Details anzeigen“-Schalter, den jeder Besucher selbst aktivieren kann und der jeder Antwort einen Live-Trace hinzufügt — die genommene Route, das Guardrail-Ergebnis, die Latenz und welche Modelle verwendet wurden —, ohne sonst etwas am Gespräch zu ändern. Es ist ein optionales Fenster in die Mechanik, kein separates Erlebnis.

flowchart TD
A[Besucher kommt auf luism.pro an] --> B{Dynamische Schicht funktionsfähig?}
B -- Nein --> C[Nur statische Website
CV-Download, Portfolio, Kontakt-E-Mail]
B -- Ja --> D[KI-Begrüßung + Schnellaktionen]
D --> E{Was möchte der Besucher?}
E -- Frage zu Luis --> F[Über-Luis-Spezialist
Karriere, Fähigkeiten, Persönlichkeit]
E -- Frage zu einem Projekt --> G[Portfolio-Spezialist
Data Express / Data Sentinel / Envoy]
E -- Frage zur Funktionsweise der Seite --> H[Über-das-System-Spezialist]
H -- Stellenbeschreibung einfügen --> I[Job-Abgleich-Spezialist
strukturierter Eignungsbericht]
I -- Möchte Kontakt aufnehmen --> J[Kontaktformular
Name, E-Mail, Nachricht]
J --> K[Inhaltsbereich aktualisiert sich
+ Menü wächst kontextabhängig]
K --> L["Technische Details anzeigen"-Schalter
Route, Guardrail-Ergebnis, Latenz, Modelle]

Abb. 1 — Besucher-Journey: statischer Fallback im linken Zweig, KI-gesteuertes Routing rechts, durchgehend ein optionaler Detail-Schalter.

Was die Website abdeckt

BEREICH	INHALT
Über mich	Profil, Werdegang, Ausbildung, Zertifizierungen, Arbeitsstil
Portfolio	Data Express, Data Sentinel, Envoy selbst — jeweils mit direkt in die Oberfläche eingebetteter Demo
Über das System	Wie Envoys eigene KI-Schicht aufgebaut ist, einschließlich der echten Workflow-Diagramme und RAG-Quelldokumente
Job-Abgleich	Stellenbeschreibung einfügen, Eignungsanalyse gegen Luis' echtes Profil erhalten
Analyse	Die eigene Sitzung des Besuchers, auf Anfrage in einfacher Sprache zusammengefasst
Kontakt	Direktes Kontaktformular (Name, E-Mail, Nachricht), per E-Mail an Luis gesendet, mit Bestätigungskopie an den Besucher

Zuverlässigkeit als geschäftliche Entscheidung

Die statische Schicht ist kein nachträglich angeflanschter Fallback — sie ist die bewusste Grundlage, und die Zuverlässigkeitsentscheidung reicht tiefer als reine Frontend-Logik. Das System wird nicht von einem einzigen Punkt aus gehostet: Die statische Website läuft auf ihrer eigenen Cloud-Infrastruktur, unabhängig

vom Docker-Stack, der die dynamische KI-Schicht betreibt, sodass ein Ausfall auf der KI-Seite niemals die gesamte Website lahmlegt. Ist die KI-Schicht, die Datenbank oder ein Backend-Dienst nicht verfügbar, funktioniert die Website weiter: Der Lebenslauf lässt sich herunterladen, das Portfolio ist lesbar, Kontaktinformationen sind sichtbar. Besucher können die statische Version auch freiwillig über einen Link im Header wählen, selbst wenn die KI-Schicht funktionsfähig ist. Dass eine persönliche Website genau in dem Moment komplett ausfällt, in dem ein Recruiter sie prüft, ist genau der Fehlerfall, den dieses Design verhindern soll.

4. Technische Dokumentation

Hosting und Deployment

Die dynamische KI-Schicht — n8n und alles, wovon es abhängt (Postgres/pgvector, Redis, MinIO, der Monitoring-Stack) — läuft als ein einziger Docker-Compose-Stack, gestartet mit einem einzigen Befehl und öffentlich ausschließlich über einen Cloudflare Tunnel erreichbar; n8n selbst hat keine direkte öffentliche Route, und es sind keine Ports auf dem Host geöffnet. Dieser Stack läuft derzeit auf Luis' eigener Infrastruktur, während das System aktiv weiterentwickelt wird.

Die statische Website wird bewusst nicht von demselben einzelnen Punkt aus gehostet — ein System mit einem einzigen Host für alles würde bedeuten, dass ein Ausfall die gesamte Website lahmlegt, und genau diesen Fehlerfall soll diese Architektur vermeiden. Die statische Website wird unabhängig über ihr eigenes Cloud-Hosting ausgeliefert, sodass ihre Verfügbarkeit nicht von der Heiminfrastruktur abhängt, auf der die dynamische Schicht läuft: Ist dieser Docker-Stack einmal nicht erreichbar, bleibt die statische Website auf eigene Faust verfügbar, und nur die KI-Ebene fällt auf das unten beschriebene Fallback-Verhalten zurück.

Dienste

DIENST	ROLLE
React/Next.js-Frontend (dynamische Schicht)	Liefert die KI-Ebene aus, leitet dynamische Aufrufe an n8n weiter und bettet beide anderen Portfolio-Projekte ein — derzeit per Docker auf Luis' eigener Infrastruktur gehostet, getrennt vom eigenen Cloud-Hosting der statischen Website
n8n	Orchestriert jedes dynamische Verhalten — Triage, Retrieval, Generierung, Kontakt, Job-Abgleich, Analytics
Postgres + pgvector	Vektor-Speicher für die Wissensdatenbank; Analytics-Speicher (Bronze/Silver/Gold-Schema)
Redis	Rate-Limiting für den Chat und Caching von Sprachhinweisen pro Besucher
MinIO	Objektspeicher für die RAG-Wissensdatenbank-Dokumente selbst
Uptime Kuma	Visuelles Monitoring-Dashboard über alle Dienste hinweg
cloudflared	Der einzige aus dem öffentlichen Internet erreichbare Dienst

Retrieval- und Generierungs-Pipeline

Die Wissensdatenbank besteht aus einer Reihe von Markdown-Dokumenten über Luis, die an Überschriftsgrenzen in Abschnitte zerlegt und pro Abschnitt mit Kategorie, Priorität, Schlagwörtern und Verweisen auf verwandte Abschnitte versehen sind. Bei einer Anfrage läuft der Ablauf so: Anfrage einbetten (Gemini) → pgvector-Ähnlichkeitssuche über die gesamte Basis → Kandidaten neu ranken, wobei die Priorität-/Schlagwort-/Kategorie-Metadaten jedes Abschnitts als weicher Boost zusätzlich zur reinen Ähnlichkeit dienen → das Ergebnis nur akzeptieren, wenn der höchste (ungeboostete) Score 0,60 übersteigt und mindestens zwei Abschnitte qualifizieren → optional um eine begrenzte Menge verwandter Abschnitte per Querverweis erweitern → die Antwort erzeugen (Gemini), verankert genau in diesem abgerufenen Kontext.

```
sequenceDiagram
    participant V as Besucher
    participant FE as Next.js-Frontend
    participant N8N as n8n (Triage + Spezialist)
    participant EMB as Gemini-Embedding
    participant DB as Postgres / pgvector
    participant GEN as Gemini-Generierung

    V->>FE: Chat-Nachricht
    FE->>N8N: Weiterleitung an Webhook
    N8N->>N8N: Triage / Guardrail-Routing
    N8N->>EMB: Anfrage einbetten
    EMB-->>N8N: Anfrage-Vektor
    N8N->>DB: Ähnlichkeitssuche (+ Metadaten-Re-Ranking)
    DB-->>N8N: Top-Abschnitte (Score >= 0,60, >= 2 Abschnitte)
    N8N->>DB: Verwandte Abschnitte erweitern (begrenzt)
    DB-->>N8N: Verwandter Inhalt
    N8N->>GEN: Frage + abgerufener Kontext
    GEN-->>N8N: Faktisch verankerte Antwort
    N8N-->>FE: Antwort
    FE-->>V: Gerenderte Antwort + Aktualisierung des Inhaltsbereichs
```

Abb. 2 — Datenfluss einer Chat-Anfrage, von der Nachricht bis zur faktisch verankerten Antwort.

Projektübergreifende Integration

Beide anderen Portfolio-Projekte sind direkt aus Envoys eigener Oberfläche heraus erreichbar, nicht nur extern verlinkt — das ist das in Abschnitt 3 gezeigte Element „Systemintegration“, nicht nur eine Behauptung. Data Express ist als nativer Port seines eigenen Demo-Widgets eingebettet, direkt in Envoys React-Frontend integriert, mit zwei Modi: einem manuellen Modus, der die zugrunde liegenden Tool-Funktionen direkt ohne Modellbeteiligung aufruft, und einem Agentenmodus, der denselben Gemini-Function-Calling-Ablauf wie seine eigenständige Demo verwendet — beide arbeiten auf demselben In-Memory-Sitzungsdatensatz, erreicht über den eigenen Next.js-Server dieser App, der an den Data-Express-Backend-Container unter `/api/demo/chat`, `/api/demo/tool` und `/api/demo/health` weiterleitet. Data Sentinel läuft als vollständig eigenständige Anwendung — eigener Container, eigener Next.js-Build —, wird aber von diesem selben Frontend unter `/data-sentinel/*` per Reverse-Proxy weitergeleitet und über ein iframe eingebettet, das auf denselben Origin-Pfad zeigt. Keine der beiden Demos benötigt einen eigenen öffentlichen Hostnamen oder eine separate Cloudflare-Tunnel-Route.

Selbstreferenz

Envoys eigener Portfolio-Eintrag beschreibt das System nicht nur in Prosa — er zeigt die echten Artefakte. Ein Bilderkarussell zeigt die tatsächlichen Screenshots der laufenden n8n-Workflows (Guardrail, Triage, jeder Spezialist, Ingestion, Admin-Abläufe), und eine Dokumentenliste zeigt die echten RAG-Quelldateien, gruppiert genau so, wie das eigene Retrieval-Scoping der Spezialisten sie gruppiert — keine generierte

Zusammenfassung als Ersatz für das Original. Zusammen mit dem über den Detail-Schalter verfügbaren technischen Trace pro Interaktion (Route, Guardrail-Ergebnis, Latenz, Modelle) kann ein technisch versierter Besucher die Funktionsweise der Website anhand der Website selbst überprüfen.

Weitere Datenflüsse

- **Analytics:** Frontend-Ereignisse (angesehener Bereich, verbrachte Zeit, CV heruntergeladen) gehen an n8n, das das Besucherland aus dem von Cloudflare mitgelieferten Header ermittelt (kein separater Geolocation-Aufruf), in Postgres speichert und auf Anfrage über Gemini eine Sitzungszusammenfassung in einfacher Sprache erzeugen kann.
- **Kontakt:** Der Besucher füllt ein einfaches Formular aus (Name, E-Mail, Nachricht) → n8n validiert die Eingabe, speichert sie und versendet parallel zwei E-Mails: eine Benachrichtigung an Luis und eine Bestätigungskopie an den Besucher. Bei diesem Ablauf ist kein KI-Entwurf beteiligt.
- **Job-Abgleich:** Eine eingefügte Beschreibung oder URL (n8n ruft die URL ab, mit einem Fallback auf eingefügten Text, da viele Stellenportale automatisierte Abrufe blockieren) geht zusammen mit Luis' Profil aus dem RAG-Speicher an Gemini; das Ergebnis ist so formuliert, dass Lücken benannt werden, statt schöngeredet zu werden.

Verhalten bei Beeinträchtigung

Beim Laden der Seite führt das Frontend einen Same-Origin-Health-Check mit einem 2-Sekunden-Timeout durch. Schlägt dieser fehl, fällt die Website in einen geordneten Fallback zurück: Der statische Inhalt bleibt vollständig sichtbar, und die Chatbox zeigt die Meldung „KI-Assistent offline“ mit einem direkten E-Mail-Link — keine defekte Oberfläche, keine Sackgassen.