PROJECTS

Comprende

Local Comprehension Proxy for AI Coding Tools

Lead Architect · 2026

Comprende is a local comprehension proxy that sits between AI coding tools and LLM APIs: it scores each prompt for clarity before it reaches the model, holds vague requests for clarification instead of forwarding them blindly, and journals every session as an auditable, sealable .cpr record.

---

1. Overview

Comprende sits quietly between your AI coding tools and the language models they call. Before a prompt reaches the model, Comprende scores it for clarity; vague requests are held for clarification rather than forwarded blindly, and every request and response is journaled into an auditable, cryptographically sealable .cpr record. It speaks both the OpenAI Chat Completions and Anthropic Messages protocols, so it drops in front of the tools you already use while routing to any upstream provider — and it runs entirely on your own machine.

2. Key Features

- **Comprehension Gate** Each prompt is scored locally for clarity before it reaches the model; vague requests are held for clarification, with optional escalation to a cloud LLM.
- **Drop-In Compatibility** Multi-protocol inbound (OpenAI Chat Completions + Anthropic Messages) means it works in front of Claude Code, Codex, Aider, Continue.dev and Goose by changing a single base URL.
- **Provider-Agnostic Routing** Routes to any OpenAI-compatible or Anthropic backend, so the proxy is independent of which model you ultimately call.

- **Auditable .cpr Sessions** Every session is journaled and sealable as a signed .cpr document — a tamper-evident record of what was asked and answered.
- **Local & Private** Runs as a local daemon with a Tauri + React dashboard; Ed25519/X25519 signing and encryption keep everything on your machine.

3. Architecture

Comprende is built in Rust as a fifteen-crate workspace: a comprehension gate, an axum streaming HTTP proxy, a process-detection daemon, dedicated Claude / Cursor / Copilot integrations, plus coaching, attribution and observability crates and shared types. A Tauri shell wraps a React + Vite dashboard for monitoring and configuration, and a crypto crate provides Ed25519 signing and X25519 encryption for sealing sessions. The proxy listens locally on :17380 and gates, forwards and streams between client and upstream.

TECH STACK

Rust · axum · Tauri · React · Vite · Ed25519 / X25519 · 15-crate workspace

LINKS

https://www.comprende.dev/
https://www.jonathandumitru.com/projects/comprende