

HealthClaw Guardrails

SHARP-on-MCP compliance layer for AI agents accessing FHIR data

Submitted to the **PromptOpinion Agents Assemble Challenge**

Build Interoperable Healthcare Agents at the Intersection of MCP, A2A & FHIR

A project of **fhirig**

healthclaw.io · github.com/aks129/HealthClawGuardrails

May 2026

Links

Marketplace — Agent	app.promptopinion.ai/marketplace/agent/019e183d...
Marketplace — MCP Superpower	app.promptopinion.ai/marketplace/mcp/019e1831...
Demo video (under 3 min)	youtu.be/2fVL28CW9p8
Source code	github.com/aks129/HealthClawGuardrails
Live MCP endpoint	mcp-server-production-5112.up.railway.app/mcp
Marketing + skills site	healthclaw.io

Problem

Every health system in the country is running AI experiments. Almost none have agents touching production charts. The blocker isn't capability — it's compliance. The moment a model sees a name, an MRN, or a date of birth, that conversation is governed by HIPAA, every state's analog, and the organization's BAA stack. Projects stall at *we can't let the agent touch real data*.

The status quo gives three bad options:

- **Build per-EHR.** Re-implement guardrails in every Epic, Cerner, MEDITECH, athenahealth, eClinicalWorks deployment.
- **Anonymize upstream.** Ship a one-way de-identification pipeline that destroys the round-trip — fine for analytics, broken for agentic workflows that need to read *and* write back.
- **Trust the model.** Include "do not output PHI" in the system prompt and hope for the best. This is the current default. It is not auditable.

HealthClaw makes the right thing the default.

What it is

HealthClaw is an MCP server (Superpower) plus an A2A agent that sits between any AI agent and any SMART-on-FHIR endpoint. The agent host obtains a SMART access token; HealthClaw forwards it on every call via `X-FHIR-Server-URL`, `X-FHIR-Access-Token`, `X-Patient-ID` headers, routes the request to the correct upstream EHR, applies the full guardrail stack on the response, and only then returns data to the model.

The same deployment works against Epic, Cerner, MEDITECH, athenahealth, eClinicalWorks, HAPI, SMART Health IT — no per-EHR code, no per-customer rebuild. That portability comes from compliance with two open specs:

- **SHARP-on-MCP** (sharp-on-mcp.com) — vendor-neutral header-forwarding contract advertised under `capabilities.experimental`
- **PromptOpinion FHIR Extension** — advertised under `capabilities.extensions["ai.promptopinion/fhir-context"]` with a SMART-on-FHIR scope manifest (`patient/*.read` required, `patient/*.write` and `offline_access` optional)

Both specs declare the same headers and the same scope model, so a single MCP server satisfies both ecosystems.

What every response gets

On every read

- PHI redaction — names → initials, identifiers masked, addresses stripped, birth dates truncated to year, photos removed
- Immutable `AuditEvent` appended to a tenant-scoped, append-only trail
- Medical disclaimer injected on clinical resources
- Upstream URLs rewritten so the source EHR never leaks into the response

On every write

- HMAC-SHA256 step-up tokens with 128-bit nonce and 5-minute TTL
- Human-in-the-loop gate on clinical resources (HTTP 428 until `X-Human-Confirmed`)
- Local `$validate` runs before commit
- `ETag / If-Match` concurrency control

Always

- Tenant isolation enforced at the database layer (local mode) or propagated as a guardrail header (proxy mode)
- OAuth 2.1 + PKCE (S256), dynamic client registration, token revocation

Why this needs generative AI

Three places where generative AI does what conditional logic can't:

1. Tool selection under uncertainty. A clinical question doesn't map cleanly to one endpoint. *What's this patient's recent diabetes control look like?* turns into `fhir_search(Condition, code=diabetes) → fhir_lastn(Observation, code=HbA1c) → fhir_stats(Observation, code=glucose) → narrative synthesis`. The agent does the planning; HealthClaw does the policy.

2. Curatr semantic quality checks. A smoking-status field set to *current smoker* combined with a clinical note saying *patient denies tobacco use* is a contradiction no validator catches with conformance rules. The agent reads both, flags the inconsistency, and (with step-up + HITL) proposes a Provenance-linked fix.

3. Guardrail narration. The demo agent doesn't just retrieve data; it points out *what HealthClaw did* to each response — *"the patient's name has been truncated to initials per HealthClaw's HIPAA Safe Harbor de-identification"* — making the compliance layer visible to clinical reviewers in a way pure log lines can't.

Potential impact

PHI exposure is THE blocker for clinical AI deployment in 2026. Every CIO survey says it; every pilot that doesn't ship cites it. HealthClaw is a deployable architectural pattern that converts the blocker into a configuration choice. Once a health system trusts the redaction + audit + HITL guarantees, the conversation moves from *can we let the agent see this?* to *which tools should we enable?*

Because the server is SHARP-on-MCP + PromptOpinion compliant rather than EHR-specific, a single HealthClaw deployment can sit in front of an entire health system's SMART-launched agent ecosystem — a 50x reduction in per-vendor compliance work versus the per-EHR alternative.

The pattern is also reusable beyond healthcare. Any regulated domain with similar boundary requirements — GLBA financial records, FERPA education records, government CUI — fits the same shape: agent host forwards an access token, server applies policy on the response, audit trail emitted.

Feasibility

This is not a slide-deck submission. The full stack is deployed today:

Flask app	Railway · app.healthclaw.io · FHIR REST facade, OAuth 2.1, audit, redaction, Curatr
MCP server	Railway · mcp-server-production-5112.up.railway.app · Streamable HTTP + SSE + JSON-RPC bridge
OpenClaw stack	Telegram personas (Sally-PCP, Mary-pharmacy, Dom-fitness, Kristy-scheduler)
Marketing site	Vercel · healthclaw.io · skills catalogue + quickstart PDF

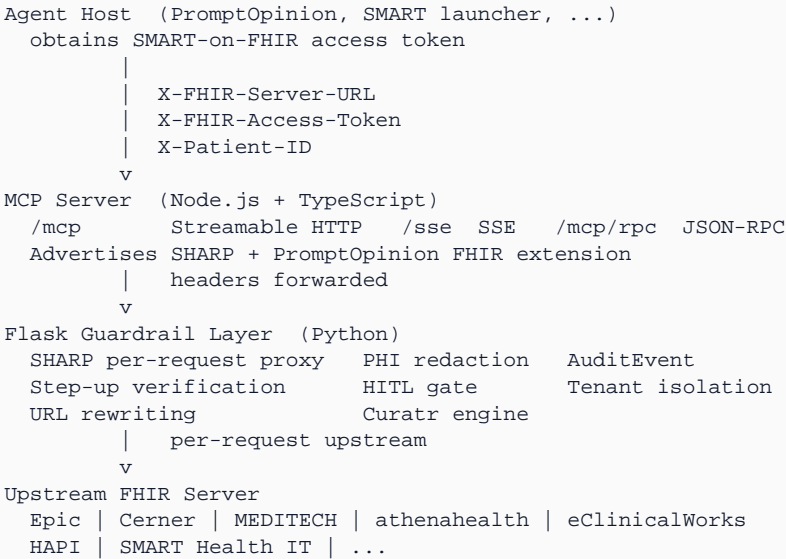
Test coverage: 516 Python tests + 49 Node tests, all passing. TypeScript strict-mode `tsc --noEmit` clean. End-to-end gate script (`scripts/demo_e2e.sh`) covers 10 compliance gates: liveness → seed → read-with-redaction → audit trail → cross-tenant isolation → Curatr evaluate → human-in-the-loop.

Compliance posture:

- HIPAA Safe Harbor de-identification on by default; patient-controlled mode preserves selected fields
- SOC2-aligned audit trail with database-level immutability
- HITRUST-aligned tenant isolation
- `.claude/compliance/{hipaa,soc2,hitrust}.md` gate checklists committed to the repo

Demo data is synthetic. The desktop-demo tenant is seeded with a Grover Keeling sample record on first boot. No real PHI was used in any test, screenshot, or video.

Architecture



Tool catalog

Read tier	context_get, fhir_read, fhir_search, fhir_validate, fhir_stats, fhir_lastn, fhir_permission_evaluate, fhir_subscription
Write tier (step-up)	fhir_propose_write, fhir_commit_write, curatr_apply_fix
Utility	fhir_get_token, fhir_seed

Coverage spans FHIR R4 US Core v9 stable resources (AllergyIntolerance, Immunization, MedicationRequest, Procedure, DiagnosticReport, Coverage, ServiceRequest, Goal, CarePlan, Patient, Encounter, Observation, Condition, ...) and FHIR R6 ballot3 experimental resources (Permission, SubscriptionTopic, DeviceAlert, NutritionIntake).

Tech stack

Backend	Python 3.11+ (Flask, SQLAlchemy, httpx, gunicorn), Node.js 20 + TypeScript (MCP SDK, Express)
Specs	MCP 2024-11-05, SHARP-on-MCP 1.0, PromptOpinion FHIR Context ext, SMART-on-FHIR, FHIR R4 US Core v9, FHIR R6 ballot3
Infrastructure	Railway (Flask + MCP + Postgres + Redis), Vercel (marketing), GitHub Actions
Storage	SQLite default · PostgreSQL on Railway · Redis (rate-limit, sessions, token cache)
Testing	pytest (516), Jest (49), Playwright (browser e2e), demo_e2e.sh (10 gates)

What's next

- Strict StructureDefinition + terminology binding validation (currently structural only)
- SubscriptionTopic notification dispatch (currently storage + discovery)
- Cryptographic human-in-the-loop confirmation (currently header-based)
- Cross-version translation for R5/R6 upstreams (currently pass-through)
- Provider Directory de-duplication on the upstream proxy path

Open source. github.com/aks129/HealthClawGuardrails · healthclaw.io

A project of fhiriq.