CASE STUDY

Before the Denial Lands:

An Embedded Prior Authorization AI Agent That Thinks Ahead Revenue Cycle

Executive Summary

Labs lose PA revenue not from coverage gaps but from inconsistent submissions. NonStop built an embedded AI agent that reads the clinical order, pulls the right payor policy, maps the correct codes, and assembles a complete submission package before it goes out — with a counselor reviewing every draft before submit.

Top Problems Faced by the Business

1. Every payor has different templates, codes, and submission rules — no consistency across reps or counselors
2. Same clinical indication coded differently by different staff — wrong code pairings, missing documents, wrong templates
3. Submissions go out incomplete or miscoded — denials trigger not from coverage gaps but from translation errors

Impacts Those Problems Created

1. Denial rates climbed on preventable submission errors — not coverage issues, execution issues
2. Staff burned on resubmission cycles — chasing status, correcting errors, back-and-forth with payors
3. Revenue silently lost at scale — labs don't realize how much is leaking until it compounds

Solution We Built Self-hosted

AI agent embedded between order intake and PA submission — pulls payor-specific templates and policies via RAG, translates clinical intent to correct ICD-10/CPT pairings, assembles complete submission package, flags gaps before it goes out, counselor reviews and submits

Impact of Our Solution

- Submission consistency: rep-dependent and variable → standardized per payor-test combination
- Code accuracy: probabilistic human judgment → deterministic rules engine validation before every submission
- Missing documentation: caught post-denial → flagged before submission
- Counselor role: rebuilding submissions from scratch → reviewing a complete, policy-matched draft
- Denial rate: materially reduced — submission errors eliminated upstream
- PHI exposure: zero — self-hosted, stateless, never leaves the VPC

The Problem

Prior authorization is where genetic testing revenue either gets secured or silently lost, and most labs are losing more than they realize, not because the test isn't covered, but because the submission itself is inconsistent.

Every payor has its own PA template, its own policy documents, its own submission channel, and its own rules for what constitutes a clean submission. The same clinical indication, say, a family history of hereditary breast cancer, gets translated into ICD-10 and CPT codes differently depending on which counselor or lab staff member is preparing the request.

> **“One person codes it as Z15.01 with CPT 81162. Another uses Z80.3 with 81432. Both are defensible, but only one matches what that specific payor's policy expects for auto-approval.”**

This inconsistency compounds at every step. The wrong template gets used. A required clinical document is missing. The code pairing doesn't match the payor's medical policy bulletin. The submission goes out, gets denied or pending, and now the lab is in a back-and-forth cycle, resubmitting with corrections, chasing status updates, burning staff time on work that should have been right the first time.

The downstream impact is real: denial rates climb not because of coverage gaps but because of submission errors. This isn't a coverage problem. It's a translation problem, clinical intent to payor-specific documentation, and it's different for every payor, every test, and every submission.

Our Approach

The goal isn't to replace the PA workflow; it's to embed intelligence at the point where inconsistency enters: the translation of clinical intent into payor-specific submission packages.

Intelligent PA agent

The AI agent sits between the lab's existing order/accessioning system and the PA submission step. When a test order comes in, the agent reads the clinical context, test type, patient diagnosis, ordering physician notes, family history, prior test results, and does three things:

First, it identifies the correct payor and pulls the relevant PA template, medical policy document, and submission requirements for that specific payor-test combination.

Second, it translates the clinical indication into the payor-preferred ICD-10 and CPT code pairing.

Third, it assembles the complete submission package: populated template, supporting clinical documentation, and correct codes, and flags anything missing before it goes out. The agent presents this as a draft, the counselor reviews, confirms or adjusts, and submits.

The Architecture

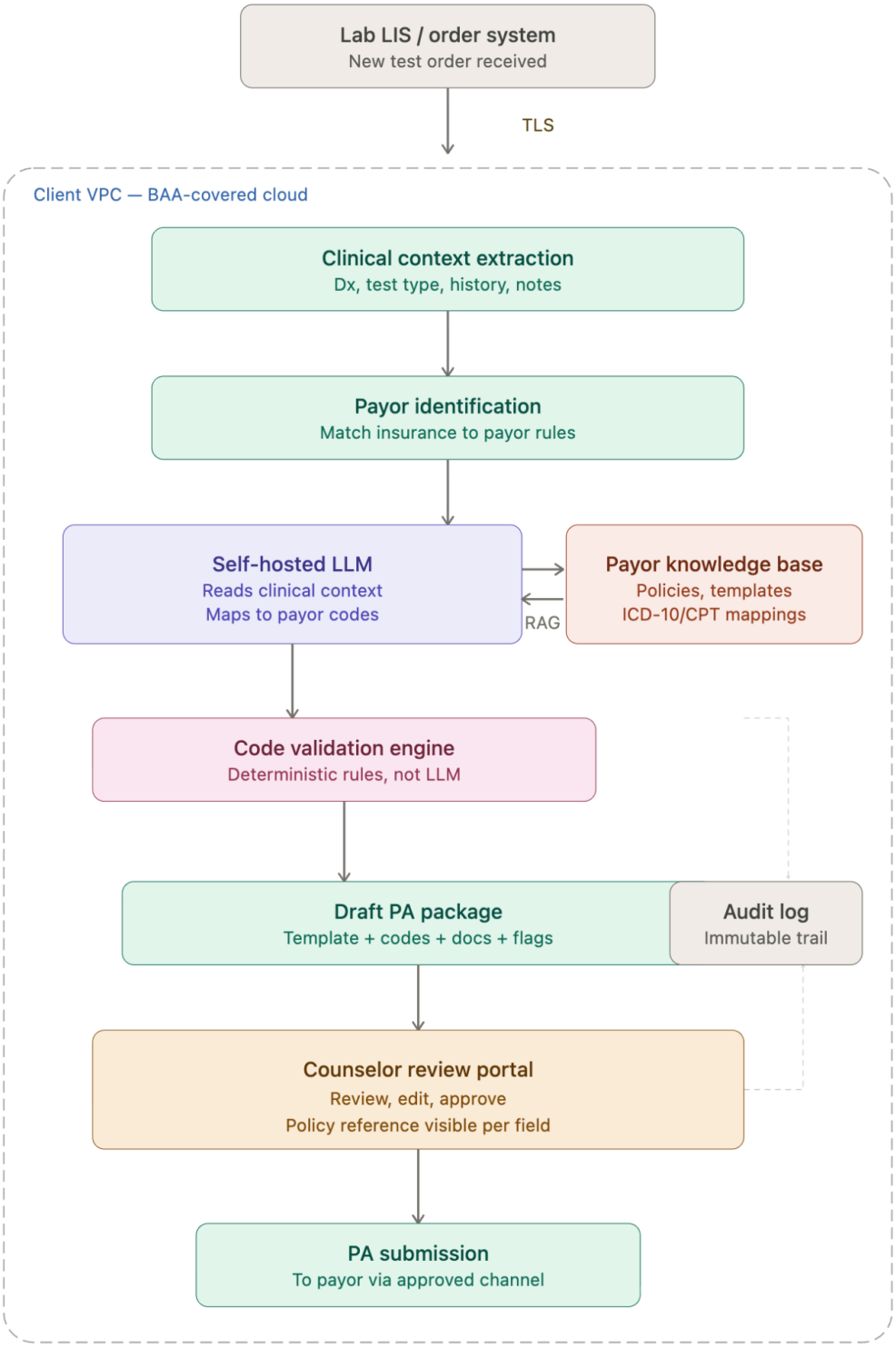
The architecture is driven by two constraints: patient data flows through this system, so HIPAA compliance is non-negotiable, and the agent needs access to payor-specific policy logic that changes frequently, so the knowledge layer has to be maintainable without retraining a model.

Self-hosted model

No external API calls with PHI. The LLM runs inside your cloud environment. Clinical data, patient records, diagnosis details, and physician notes never leave the VPC. The model processes in memory inside a stateless container, produces structured output, and retains nothing.

Retrieval-augmented generation for payor policy logic

We don't bake payor rules into the model's weights, policies change quarterly, and retraining is neither fast nor cheap. Instead, the agent queries a structured knowledge base of payor-specific medical policies, code mappings, template requirements, and submission rules at inference time. When a payor updates their policy bulletin or changes their accepted CPT codes for a test category, the knowledge base is updated, and the model doesn't need to be retouched.



Code validation layer

Before presenting the draft to the counselor, the agent runs the proposed ICD-10 and CPT code pairing through a deterministic rules engine, not the LLM, that cross-references against the payor's known acceptance criteria. This is a hard check, not a probabilistic suggestion. If the code pair doesn't match, the agent flags it and suggests the correct pairing with the policy reference. The LLM handles the messy part (reading clinical context, interpreting physician notes). The rules engine handles the precise part (code validity against payor policy).

Human-in-the-loop

The agent never submits a PA autonomously. Every draft goes to the counselor's review queue with full transparency. Here's the code pairing the agent chose, here's the policy reference it used, and here's what's in the submission package. The counselor approves, edits, or overrides.