Technical Partner

A practice grounded in inspectable work

Pablo Cardozo
Buenos Aires, Argentina
GitHub ↗ · LinkedIn ↗ · Notes ↗

Abstract

Models often fail after the demonstration, not during it. They remember the wrong user, pass the wrong test, or return valid data without a defensible source. My work begins at that boundary: I inspect the implementation, isolate the claim on which a build depends, and look for the observation that would force a different decision. The three studies gathered here draw on systems to which I contributed and then place one consequential claim under closer examination. One shows how an 87% aggregate score concealed complete failure in payments. Another separates 569 passing software tests from an extraction result that remains unmeasured. The third defines an executable memory-isolation protocol whose outcome is still open. Together they form the record behind my work as a technical partner.

Keywords: early-stage engineering; feasibility; conversational memory; model evaluation; document provenance

1. The point before the build

A young company can make an expensive engineering choice while the question beneath it is still vague.

A team may need to know whether memory should persist between sessions, whether extracted data can be traced to its source, or whether a bad outcome comes from the model, the state machine, or the test itself. The first available number is rarely enough. Recall accuracy does not prove tenant isolation. A broad test suite does not prove task accuracy. An average score can hide total failure at the step that completes a transaction. Once vendors, schemas, and interfaces harden around the wrong proxy, correcting the premise can cost more than correcting the code.

I work in that interval. The aim is to turn one consequential uncertainty into a written claim, then expose that claim to the smallest test capable of changing the decision. Sometimes the answer supports a build. Sometimes it narrows the design. Sometimes it shows that the proposed architecture has not earned the cost it would impose. All three outcomes are useful if they arrive before commitment.

The role is deliberately bounded. I assess implementation feasibility, architecture, data flow, memory, permissions, evaluation design, latency, operating cost, and failure recovery. Customer discovery, market size, pricing, distribution, sales, and product–market fit remain with the founder or with specialists in those disciplines.

1.1 What the work must produce

The useful object is not more software by default. It is a decision whose reasoning can be inspected after the meeting ends.

Question	Required artifact	What it prevents
What are we deciding?	One sentence with a named consequence	A discussion that changes subject without resolving anything
What is already known?	Sources, constraints, and observed behavior	Treating recollection or preference as fact
What remains assumed?	A claim that can be contradicted	Building around an idea that no result could disprove
What would change the choice?	A bounded test and an explicit threshold	A demonstration designed only to look successful
Where does the answer stop?	A recommendation with stated limits	Generalizing beyond the data, revision, or environment examined

Table 1: The minimum record for a defensible technical recommendation.

2. A method that can be contradicted

Every engagement begins with a sentence of the form: *We need to decide whether X is justified before we commit Y.* The language matters. If the sentence cannot name the choice, the cost of being wrong, and the observation that would change the answer, the work is not ready to begin.

I then separate the record from the hypothesis. Repositories, logs, schemas, saved runs, provider constraints, and existing tests belong to the record. Expectations about accuracy, latency, isolation, or cost remain hypotheses until an observation supports them. The test is chosen last, because a test inherits the confusion of the question that produced it.

The final recommendation uses one of three terms:

Viable	The inspected result supports the proposed choice within the stated conditions.
Conditional	The choice can proceed only if a named constraint, missing measurement, or mitigation is accepted.
Not justified	The available record does not support the cost or risk of the proposal.

These words are not grades for the founder. They describe the relation between a claim and what was actually observed.

3. The record behind the offer

A service can promise judgment. A record lets the reader inspect how that judgment was earned.

The documents below are not a catalogue of victories. Each begins with an implementation or a preserved run and asks whether the available proxy can support the decision being made. The useful passages are often the ones that narrow a conclusion.

3.1 The defect hidden by the average

A saved judge run for a restaurant assistant reported 40 passes in 46 cases and a mean score of 80 out of 100. Read alone, the aggregate suggests broad competence. Read by category, it says something more useful: every payment case failed, while only one case failed elsewhere.

Failure-Focused LLM-as-a-Judge Evaluation reconstructs that run and examines what its scores can support. The concentration justifies inspecting intent routing and transaction state before attempting broad model tuning. The audit also finds that cases labelled as security or resilience checks never exercise the HTTP, retry, rate-limit, or circuit-breaker behavior their names imply. Local verification passed 79 assertions, but the test process retained open handles, and the end-to-end judge run was not repeated.

The report changes the next engineering step: inspect and test the transaction boundary; move deterministic behavior out of the model judge; calibrate open-ended scoring against blinded human review.

TR-01 · Saved 46-case run · 5 payment failures · Independent rerun still open

Read the full technical report ↗

3.2 Provenance before plausibility

Schema-valid JSON can still be wrong in the place that matters most: its relation to the source. A catalog parser may omit a row, merge two products, or attach a correct-looking price to the wrong item while satisfying every field type.

Provenance-Constrained Multi-Stage LLM Extraction examines an implemented pipeline that separates document metadata, source-row reconstruction, and product normalization. The last stage works in ten-row batches and must echo the expected batch identity, row set, and page coverage before data can be persisted. Additional gates reject items without a source row and defer ambiguous or non-normalizable products for review.

At the pinned public revision, the complete local suite passed 569 tests across 20 files in 1.06 seconds. That result supports the encoded validators and application behavior. It does not establish extraction accuracy. The separate throughput test uses an in-memory array, not the database, provider, or end-to-end path. No labelled corpus or single-pass baseline exists in the inspected artifacts, so the comparison remains a protocol awaiting execution.

RP-01 · 569 tests reproduced · Extraction accuracy not yet measured

Read the full research protocol ↗

3.3 Memory that respects a boundary

For persistent conversational memory, forgetting is the obvious failure. The more dangerous error is plausible recall from the wrong tenant: the answer may read well while the system has crossed the boundary that makes recall acceptable.

Tenant-Scoped Long-Term Memory in Conversational Agents formalizes four invariants: scoped writes, scoped retrieval, negative assertions against leakage, and complete deletion. The checked-in runner creates two synthetic tenants, writes mutually exclusive facts, and applies six probes with lexical requirements and a latency target. On that dataset, the declared accuracy threshold requires six correct cases out of six.

There is no preserved live-run result at the cited private revision. The document therefore establishes a threat model and an executable protocol, not compliance with the target. Its practical consequence is clear: recall quality cannot be reported without testing whose memory supplied the answer.

WP-01 · Executable protocol · No public live-run artifact

Read the full working paper ↗

3.4 What the three documents establish

Record	Supported	Still open
WP-01	Isolation invariants, threat model, six-probe runner, decision rule	A versioned live run and broader adversarial coverage
TR-01	Saved case-level results, failure concentration, audit of test validity	Human calibration, repeated seeds, clean-tree end-to-end rerun
RP-01	Three-stage controls and a reproduced 569-test software suite	Labelled-corpus accuracy, paired baseline, latency and API cost

Table 2: Claims are limited to the artifact and revision named in each paper.

4. The publication standard

These are independent working papers, not peer-reviewed publications. Each names the repository revision, preserved output, local environment, or missing artifact on which its claims depend. Implementation, reproduced tests, saved model judgments, and proposed experiments are kept separate because they answer different questions.

A portfolio usually selects its victories. This one also records the points at which a number would say more than the work can support: a memory benchmark without a live result; an 87% pass rate whose failures all gather around payment; 569 software tests without a labelled extraction corpus. Those limits do not weaken the record. They define it.

The writing follows the same rule as the engineering. A conclusion should be no broader than the observation beneath it. When new runs exist, the documents should change. Until then, the absence remains visible.

5. Working together

The work fits a founder who has one consequential engineering question, some material worth inspecting, and enough freedom to change course if the result contradicts the current plan. Useful material may include a repository, an architecture sketch, logs, sample inputs, saved outputs, cost assumptions, or a failing workflow.

5.1 Engagements

Format	Work and output	Timing	Fee
Technical Decision Session	Brief pre-work, a 75-minute session, and a one-page Technical Decision Brief within 24 hours	75 minutes	USD 120 for the first five clients; then USD 180
Technical Hypothesis Sprint	Two sessions, one risky claim, a fixed-scope proof or spike, explicit failure conditions, and a 30-day technical plan	7 days	USD 550 for the first three cases; then USD 850

Table 3: Fixed-scope ways to examine one primary technical question.

The session fee is credited toward a sprint when I accept the follow-on work. Neither format includes a complete MVP, open-ended development, indefinite support, or a promise of commercial results. One sprint can be active at a time.

The application asks what must be decided, why it matters now, what already exists, and whether the project conflicts with excluded logistics or last-mile work. Submitting it does not book a session or charge anything.

Submit a technical question ↗

Terms ↗ · Privacy ↗ · Privacy request ↗

Pablo Cardozo · Buenos Aires, Argentina
github.com/pjcdz ↗ · linkedin.com/in/pjcdz ↗ · blog.cardozo.com.ar ↗