Confidence and Accuracy Framework

Attachable artifact. Generic method, no client data, no client identity.

This is the document the job post asks for as a deliverable (“documentation of extraction accuracy and confidence thresholds”), written up front so you can see how I think before hiring me. The thresholds here are starting points from my production experience; the real ones get set from your labelled baseline in week 1.

---

1. Where confidence comes from

Never ask a model how confident it is and trust the answer. Compute confidence from evidence you can inspect.

Field-level score. Each target field gets a weight by materiality and a score by evidence quality.

EVIDENCE QUALITY	SCORE	EXAMPLE
Anchored match	1.0	Value found adjacent to its own label in the layout
Structural match	0.8	Value inferred from a table column position, label absent
Template match	0.7	Format-specific parser produced it
Model output	0.5	LLM returned it with no anchor in the text
Inferred / defaulted	0.2	Filled from a per-customer default

Record-level confidence is the weighted sum, capped by the weakest mandatory field. A record with a beautiful header and no line items must not score 0.8. Missing anchors cap the score regardless of how clean everything else looks.

Provenance is stored per field, not per record. Which tier produced it, which parser or model, the source text span, and the page or cell coordinate where layout is known. This is what makes a review fast instead of a re-key.

---

2. Threshold bands

Four bands. One global threshold is always wrong, because the cost of a false auto-pass and the cost of a needless review differ per document type and per field.

BAND	TYPICAL RANGE	BEHAVIOUR
Auto-pass	>= 0.90	Write to the core system, log everything, no human touch
Auto-pass with sampling	0.90 - 0.95	Write, plus N percent random review so drift is caught early
Review queue	0.60 - 0.90	Held. Human sees extracted values beside the source evidence
Reject / escalate	< 0.60	Never reaches the core system. Alert, not just a queue entry

Field-level overrides on top of record-level bands:

- Any monetary amount, account identifier, or counterparty identity below 0.99 goes to review. Full stop. These are the fields where a plausible wrong value is worse than a missing one.
- Line-item quantities must reconcile against the document's own total quantity. Disagreement is a hard fail, not a confidence penalty.
- Dates that resolve ambiguously (DD/MM versus MM/DD) never auto-pass unless a locale anchor from the same document disambiguates them.

Asymmetric gates. The threshold for "let this through automatically" and the threshold for "route this to a human" are deliberately different numbers, and the gap between them is where the system is allowed to be cautious. For a fintech buyer, the asymmetry should lean the other way from my grocery pipeline: there, losing a real order was the expensive error, so the classifier fails open. Here, writing a wrong financial record is the expensive error, so it should fail closed. I would set that direction explicitly with you in week 1, in writing, per document type.

---

3. What to measure, and how to report it

Per document type, per week, generated from telemetry rather than written by hand:

METRIC	DEFINITION	WHY YOU SHOULD CARE
Field accuracy	Correct fields / checked fields, per field	Tells you which field is the problem
Record auto-pass rate	Records that never saw a human	The actual labour saved
Review rate	Records held for a human	Cost, and a proxy for parser health
Silent drop count	Accepted inputs with no resulting record	The number that catches real incidents
False auto-pass rate	Auto-passed records later found wrong	The safety number. Target near zero
Model fallback share	Tier 2 calls / total	Cost control and template coverage
Cost per document	Model spend / documents	What the pipeline is actually worth
p50 / p95 latency	Intake to decided	Whether this scales with your volume

Go / no-go gate for any new parser or model change. The shape I have used:

- Field-by-field accuracy at or above 95 percent versus the incumbent path, on a labelled corpus.
- No increase in manual review rate.
- Model fallback under 25 percent of volume for a format that has a template.
- Zero new data loss versus the incumbent path.
- Automatic rollback if accuracy drops below 90 percent, or if model-invented identifiers exceed 5 percent of model-assisted records.

Baseline before building. I measure the current manual process first (minutes per document, error rate, backlog). Without a baseline there is no honest accuracy claim at the end, and the buyer has no way to check my numbers. I

would rather report an unflattering week 1 baseline than a vanity final report.

---

4. The human review loop has to be a product, not a spreadsheet

Review queues rot when the reviewer cannot work fast. Requirements I hold myself to:

1. Source evidence and extracted values on one screen, positioned next to each other.
2. Approve, correct, or reject in one action. Corrections are captured as labelled data.
3. Every correction feeds back: repeated successful corrections on one sender draft a new template, which retires the model call for that shape permanently.
4. Reviewer identity and decision retained. That is an audit artifact, not an optional column.
5. Aging visible. A queue nobody watches is worse than no pipeline, because the business assumes the documents are handled.

---

5. Audit and compliance posture I can build to

I have not personally carried a SOC 2 Type II audit and I will not imply that I have. What I do build is the evidence trail an auditor asks for, because a pipeline without one does not survive contact with a compliance review:

- Append-only operational event log covering intake, classification, holds, validation, deployment attempts and failures, with retention policy.
- Immutable per-record extraction evidence, so any stored value can be traced back to the document.
- Approval gates where a human decision, identity and reason are persisted, never inferred.
- Deployment mode gating: `disabled`, `shadow`, `live`. Production writes stay off until accuracy is certified, and the mode is a recorded setting, not a conversation.
- Secrets and document bodies excluded from UIs and logs by default. Encrypted token storage.
- Least-privilege service accounts scoped to what the pipeline actually needs.
- Reconciliation and drift alerts, so “it is broken and nobody knows” is an event the system reports.
- A runbook that a non-author of the system can follow at 3am.

If you have an auditor, bring them into week 1. I would rather design to their checklist than retrofit to it.