ILLUSTRATIVE SAMPLE · A FICTIONAL SYSTEM, NOT A CLIENT'S

AI readiness review: the invoice payment agent

Prepared by John Karanja · fixed scope · Sample, November 2026

One workflow: an agent (a program that takes actions) reads invoices sent to a shared inbox. It uses an AI model to pull out the details, and drafts a supplier payment for a person to approve. Is it ready for a small, limited trial (a pilot)? What needs evidence first? How do people stay in control when it fails? The agent drafts payments; it never releases them. Out of scope: model training, the choice of model, live-system access, legal opinions on data rights and any promise of accuracy.

Summary

The agent is not ready for a pilot yet, but it is close. It reads invoices from regular suppliers well. In a one-time check of 120 October drafts, the amount matched the invoice in 117. The weak points are around the model, not in it. The agent drafts a payment for any sender. Nobody can say whether a change made it better or worse. The approver sees only the total. Each of these can be fixed in days. Fix the sender rule first, then build the test set, then run a four-week pilot with the stop rules below.

> Sound: it reads known suppliers' invoices well, and it cannot release a payment.

> Fragile: it acts on mail from anyone, and the approver cannot see what they are approving.

> Do first: approved suppliers only, with every invoice matched to a purchase order.

Scope and method

In scope: the path from the shared inbox to the ledger, the agent's instructions and configuration, the approval screen and the people who use it. Not in scope: the bank's own systems, how the AI model works inside, model training, accounting rules, and legal questions about sending invoices to an AI model.

Materials reviewed:

- Read-only access to the agent's code, instructions and configuration, with their change history.
- The agent's run log for 1 to 31 October: 164 messages read and 151 drafts created.
- A copy of the October inbox and attachments, private details removed.
- Screenshots of the approval screen and the two-page approver guide.
- The supplier list from the ledger: 212 suppliers, 58 of them paid in the last 90 days.

- Two calls, one with the finance lead and one with the developer who built the agent.

Evidence has three labels. Observed: I saw it in the materials, and I name the source. Reported: the team told me, and I did not check it. Unknown: the evidence was not available. I never count an unknown as zero.

The map

1 **Shared inbox** (1) Every 10 minutes, the agent collects all mail sent to the accounts address, from any sender.

2 **Invoice reader** An AI model pulls the supplier name, bank details, amount, due date and invoice number from the attachment.

3 **Draft payment** (2) The agent matches the supplier name to the ledger and creates a draft. No tests check changes to how it does this.

4 **Approval** (3) One of two finance staff approves or rejects each draft. They see only the total, the supplier name and the due date.

5 **Ledger** Approved drafts are posted to the ledger, and a person releases the payment in the bank's system.

Findings

1. The agent acts on mail it should not trust

HIGH

EVIDENCE	Observed. Invoices from any sender are read and drafted, including first-time senders. Of the 164 messages in October, 13 came from addresses not seen before, and the agent drafted payments for 9 of them. Two of those 9 carried bank details different from the ledger's for a supplier of the same name. Both were rejected, by an approver who recognised the change.
IMPACT	A changed bank account on a convincing invoice is a known way to pay the wrong party. Today the only protection is one person's memory.
UNKNOWN	Nothing important.
RECOMMENDATION	Act only on approved suppliers, and match every invoice to a purchase order. Mail from an unknown sender goes to a review queue, never to a draft. A change of bank details is confirmed by a person, outside email.
DONE WHEN	Replay the 13 unknown-sender messages from October. None produces a draft, and all 13 appear in the review queue.

2. Nobody can say whether a change helped

HIGH

EVIDENCE	Observed. There is no set of past invoices with known correct drafts. The agent's instructions changed four times in October, according to the change history. Reported: after each change the developer checks "a few recent invoices" by eye. In my one-time check, the three wrong amounts were all on multi-page invoices, where the agent took the subtotal before tax.
IMPACT	Every change is a guess. If a change broke multi-page invoices, nobody would know until someone was paid the wrong amount.
UNKNOWN	Nothing important.
RECOMMENDATION	Keep 30 past invoices as a test set and run it before each change. The finance lead writes down the correct draft for each. Include the three multi-page cases, five unknown senders and the two changed bank details.
DONE WHEN	The set exists, today's score is recorded, and no change is accepted that scores lower.

3. Approval is only a formality

MEDIUM

EVIDENCE	Observed. The approver sees the amount, not the source invoice or what the agent changed. The screen shows the supplier name, total and due date, and nothing else. The run log shows a median of 9 seconds between opening a draft and approving it.
IMPACT	An approver cannot catch what they cannot see. The two rejections in finding 1 relied on memory, not on the screen.
UNKNOWN	How many drafts were approved unread. A short approval time is not proof either way.
RECOMMENDATION	Show the source document and the agent's reasoning beside every approval: the invoice image, the extracted fields, the bank details compared with the ledger, and any field the agent was unsure of.
DONE WHEN	In a test with 10 drafts, 2 with changed bank details, the approver sees both changes flagged without opening another system.

4. Unsure results look the same as sure ones

MEDIUM

EVIDENCE	Observed. The model returns a note on each field it was unsure of, and the agent throws it away. The run log shows 6 drafts with no due date on the invoice. For each, the agent filled in that day's date.
IMPACT	On the approval screen, a guess looks the same as a fact.
UNKNOWN	Nothing important.
RECOMMENDATION	When a field is empty or marked unsure, the draft is marked "needs a person" and names the field. The agent never fills a default.
DONE WHEN	Replayed, the 6 October cases are all marked "needs a person".

5. There is no written list of what it must never do

LOW

EVIDENCE	Observed. No document states the agent's limits. Its instructions tell it to "complete the payment details where possible". It can read the whole supplier list, including bank details, though it only needs names and purchase orders.
IMPACT	With no written limits, its users will be the first to test them.
UNKNOWN	Nothing important.
RECOMMENDATION	Write a one-page list: never release a payment, never change a supplier record, never act on an unapproved sender, never fill a missing field, never draft above a limit set by finance. Enforce each item in code, not only in the instructions.
DONE WHEN	Each item has a check in code and a case in the 30-invoice test set.

6. A draft cannot be traced to the version that made it

LOW

EVIDENCE	Observed. The run log records the message and the draft, but not the model's raw output or which version of the instructions was in use. For 3 October drafts I could not tell which version produced them.
IMPACT	A mistake cannot be explained, so its cause cannot be fixed.
UNKNOWN	Nothing important.
RECOMMENDATION	With each draft, store the message, the attachment, the instructions version and the model's raw output.
DONE WHEN	For any draft, one query returns its source email, instructions version and raw output.

What is sound

- It reads regular suppliers' invoices well: 117 of 120 amounts matched in the one-time check.
- The agent cannot release money. It can only write drafts, and a person releases payments in the bank's system.
- The approvers do reject. Both changed bank details in October were caught, even without help from the screen.
- The code is small and readable. The developer explained every step of it in one call.

Action plan

First

1. Add an approved-supplier list and purchase-order matching.
2. Build the 30-invoice test set and record today's results.
3. Done when: an invoice from an unknown sender is never drafted.

Next

1. Rebuild the approval screen to show the source invoice, the bank-detail comparison and unsure fields.
2. Mark unsure or empty fields as "needs a person".
3. Run a four-week pilot on approved suppliers only, with a per-draft limit set by finance.
4. Pilot success: at least 28 of 30 on the test set, every draft traceable, no draft from an unknown sender.
5. Pilot stop: any draft from an unknown sender, any unflagged change of bank details, or a test score below 27 of 30.

Later

1. Enforce the never-do list in code, with a test for each item.
2. Store the full trace for every draft.
3. Review the pilot results with finance before adding more suppliers.

Open questions

- Who owns the approved-supplier list, and how is a new supplier added?
- What share of invoices arrive with a purchase order? If many do not, matching needs another rule.
- Has any approved draft been paid to the wrong party since launch? Bank records were not supplied.
- Are there data rights questions about sending invoices to an AI model? That needs your legal adviser.

Next steps

I walk you through the findings and the pilot plan on a call, then answer one round of written questions. If you hire me to build the pilot within 60 days, this review's full fee comes off its price.

This is a sample: a fictional system, written like a real report. A real report follows the scope we agree in writing.

John Wainaina Karanja · UTC+3 · info@johnkaranja.com · johnkaranja.com/book