FINANCE · AP AUTOMATION · PROCESS INVOICES SKILL

AP Invoice Automation: how it works

What the skill does every day, where it stops and asks a human, how to run it end to end yourself, and the open calls for V3.

Status: Built & running daily **Owner:** Camille (Finance) **Board:** Basecamp Finance (production)

As of: June 23, 2026

Summary What it does Daily flow Auto vs. human Security Run it yourself Where next (V3)

TL;DR

- The skill watches both AP inboxes and the Invoices Due sheet, turns every invoice into a tracked Basecamp card, screens each email through 9 fraud controls, and keeps an on-time-payment report. It does this every morning.
- It **never moves money, never auto-sends a vendor email, and never writes the sheet without showing you the change first.** A human always does the paying, the sending, and the final approval.
- Security is hard: 10 out of 10 red-team tests pass, a real phishing email was caught and used to close a gap, and every decision is written to a tamper-evident log.
- It is done and running. The work left is governance, not code: define the SLA / "80%" target, present to the buying team, finish Salesforce 3-way match, and get it off Camille's laptop.
- This page doubles as your runbook: section 6 walks you through running it from beginning to end yourself.

1 What it does

It checks invoice email and the Invoices Due sheet for you every morning, and keeps one Basecamp board current so no invoice slips past its due date. By the time you sit down, the cards are already created, moved, and flagged.

It pulls from two sources so nothing falls through:

- **Email** (primary trigger) — invoices forwarded to `accounting@iheartraves.com` and `accounting@intotheam.com`. It reads them, parses the PDF, and creates a card.
- **The Invoices Due sheet** (backup trigger) — when the buying team adds a row without forwarding the email, the skill catches that too and creates a card from the row.

> **The three hard limits**
>
> It never moves money. It never sends a vendor email on its own. It never writes to the shared sheet without showing you the change first.

Not in scope: influencer and creator payments. Those stay a separate manual process.

2 The main steps (daily flow)

The skill is four small Python scripts plus a thin Claude layer that ties them together. Each script does one job and writes its result to a handoff file, so the run is the same every time and easy to check. They run in this order, and the order matters:

1. `reply_drafter.py` — first

   Turns any `@reply` comment on a card into a Gmail draft. Draft only, you click Send. (This one also runs by itself every hour so vendor replies stay current between full runs.)

2. `intake.py` — reads new email

   Removes duplicates, runs the fraud checks, quarantines phishing, and creates clean vendor cards. Anything it cannot decide goes to `needs-review.json` for a human.

3. `sheet_sync.py` — dry-run

   Reconciles the Invoices Due sheet against the cards and shows the planned moves as a diff. Nothing commits until you say so. It also writes the Paid column back when you post a `paid` comment.

4. `pdf_extract.py` — dry-run

   For any card still showing `$TBD`, it opens the invoice PDF and fills the amount, invoice number, due date, and bank last-4 when it can read them cleanly. You approve before it commits, then the cap and bank checks re-run on the real numbers.

5 `history.py` — then the summary

Builds the payment-history and on-time-rate report, then Claude posts the daily summary to the board.

The safety gate built into every step

Scripts run in dry-run first. They write a plan file. You see the plan before anything real happens.

3 Automatic vs. needs a human

The clearest way to understand the skill is to see exactly where it acts on its own and where it stops and asks you.

Automatic (no approval needed)

- Creating cards for known, verified vendors that pass every fraud check and are not duplicates.
- Moving cards as the sheet changes: New → Approved when Notes say approved, anything → Paid when the Paid column fills.
- Routing Alibaba payments to the Alibaba column and PayPal-only emails to Will's Queue.
- @-mentioning the person who forwarded the invoice when their card hits Paid.
- Applying Gmail labels, archiving handled email, and writing the audit log.

Needs you (it stops and asks)

- Resolving `needs-review.json`: a new vendor, an ambiguous sheet match, or something that might not be an invoice.
- Approving any write-back to the Invoices Due sheet (you see the diff first).
- Approving PDF amount fills.
- Clicking Send on a vendor reply draft.
- Actually paying. The skill never touches Chase or the bank.

A person always does these three

1. Payments — you or Will run the wire or ACH in Chase by hand. **2. Vendor emails** — always a draft, a person sends it. **3. Sheet writes** — never without a shown diff and your go-ahead.

4 The security model

Nine controls run on every email before any card is created. They are real code, not guidelines. A 10-test red-team suite has to pass 10 out of 10 before the skill does anything money-adjacent, and it currently passes all ten.

CONTROL	WHAT IT STOPS
Lookalike domain	A sender pretending to be a known vendor with a typo, a fake suffix, or a swapped-character (homograph) domain.
Hidden text & injection	Invisible text and white-on-white tricks, plus prompt-injection attempts, stripped before anything is parsed.
Bank-change kill-switch	Any email saying "new bank account," "updated wire details," and the like. The classic invoice-fraud move.
Authorized reply	Only you and Will can trigger a vendor reply draft. Anyone else's `@reply` comment is ignored.
Submitter is not approver	The same person cannot both forward an invoice and approve it.
New vendor & amount cap	A first-time vendor always goes to Manual Review no matter how small. Over-cap amounts get flagged ($25K default).
Bank last-4 mismatch	The bank last-4 read from the invoice PDF is checked against what is on file. A mismatch is a fraud hold.
Tamper-evident log	Every decision is written to a hash-chained log. If a line is ever edited, the chain breaks and the skill stops.
Phishing quarantine	Senders whose signature or links do not match their own domain (the kind that slip past SPF/DKIM), plus emails

CONTROL	WHAT IT STOPS
	fishing for A/P aging, vendor lists, or bank and routing numbers. These go to a Quarantine column with links de-fanged and no payable card.

> **It was proven against a real attack**
>
> On June 4 a phishing email impersonating a vendor (Veritiv) passed SPF/DKIM and every control of the day, because the sender owned its own domain. Nobody clicked it. That single real specimen drove the new sender-identity and sensitive-data controls, which are now regression-tested so the next one is blocked.

Two backstops sit under all of it: the **manual payment rail** (a human runs every payment in Chase, the skill has no bank access and cannot move money) and the **tamper-evident audit log** (a broken chain is a P0 stop, so the record cannot be quietly altered).

5 Run it yourself, beginning to end

This is the path to understand it hands-on. Everything below is safe: the risky stages run in dry-run and stop for your approval, and the skill cannot pay or send on its own.

Pre-flight

Open the Finance board in Basecamp so you can watch cards move in real time as the run goes.

Step 0 — prove the controls (optional but satisfying)

```
# expect: "10/10 controls passed", exit code 0
python3 ~/.claude/skills/process-invoices/redteam_tests.py
```

Step 1 — run the full thing

Tell Claude **"process invoices."** Claude runs the scripts in order and walks the judgment calls with you. To see each stage on its own, run them in this order and stop to read the output at each one:

```
reply_drafter.py → check Gmail Drafts. Nothing is sent.
intake.py → read needs-review.json; watch dedupe + quarantine.
sheet_sync.py (dry-run) → read the diff BEFORE you approve.
pdf_extract.py (dry-run) → verify the amount + bank last-4 reads.
history.py → read the daily summary.
```

Reports save to `~/Documents/Claude/output/process-invoices/` as markdown + CSV.

What to watch at each stop

- Any card showing `$TBD` — it cannot move to Paid until a human clears it.
- Any card in the **Phishing / Quarantine Review** column — review, then report or delete. Don't pay it.
- The daily summary line on the **audit chain** — it should read intact. "BROKEN" means stop and investigate.
- The **on-time rate** — this is the number that tells you AP is actually working.
- The **SKIPPED THIS RUN** section — anything the skill could not do is listed here, so you are never guessing.

> **The hard stops are yours, every time**
>
> Payments (you or Will, in Chase), vendor sends (you click Send on a draft), and sheet writes (you approve a diff). The skill will never do these for you.

6 Where it goes next (V3)

These are the open calls to make after you have run it yourself. My detailed recommendations are in chat; this is the short list.

DECISION	THE SHORT VERSION
Where it lives	Keep Basecamp as the operational queue; surface the scoreboard (on-time rate, security, audit health) in the Finance Command Center as the AP Scorecard tab, fed by `history.py`. Two layers, each doing what it is good at.
Off the laptop (keystone)	Move the engine off Camille's personal laptop to a controlled, scheduled host with rotated secrets, a durable audit log, and shared output. This one move fixes the biggest security gap and unlocks the scoreboard pipe at the same time.

DECISION	THE SHORT VERSION
Define the target	Set the SLA and the "80%" definition so the on-time rate has a bar to clear. Measurement exists now; the target does not.
Finish Phase 2	Salesforce 3-way match: reconcile PO + goods received + invoice, with freight and duty as their own line, not folded into one band.
Buying-team rollout	Present the flow to the buying team (Maryanne, Alex) and fold their feedback in.