NERO PROJECT

Deal Desk - AI Reads Every Store Ad

A hundred ads in, one useful answer out.

Paul Schindler | Built with Claude | November 2026

Executive summary

Grocery and retail stores send weekly ads by email, and nobody in a busy household reads a hundred of them to find the four items that matter. I gave the house one dedicated mailbox, subscribed it to the weekly ads of eleven grocery and retail stores, and built a desk that reads every ad on demand. Saying "going to Costco" by phone, by voice or at a keyboard returns only what is on sale that we actually buy. The desk tracks 117 regular purchases and itemizes receipts to the penny, so it shows the last price we paid next to every sale price. A sale is judged against our own history, not against the discount the store claims.

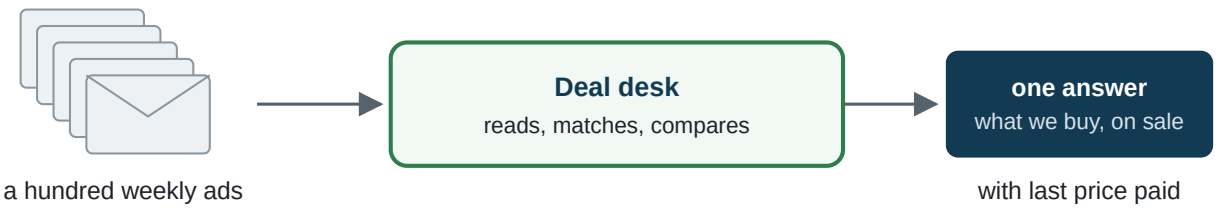
11	117	135	Live
stores subscribed to one mailbox	regular purchases tracked	receipt lines reconciled to the penny	since September 2026

What is in this report

Page	Contents
2	The problem, what I ruled out and the design choices that shaped the desk
3	How it works, from a store's email to a one-screen answer, and how it learned to read ad pictures
4	The data model (ads, watches, purchases), receipt reconciliation and an example answer
5	Nine safeguards, from untrusted mail to the matching rules
6	What the first itemized trip taught, current status, next steps and what this shows about how I work

How it was built

I set the goal, made the design calls, wrote the rules and verified each stage against real mail and real receipts. AI agents wrote the code. Every number in this report comes from the desk's own records and the working notes I kept from setup to November 2026.



1. The problem and the design choices

Store ads are written to sell, not to answer a shopper's question. Each store sends its own format on its own schedule, the discounts are measured against the store's own list price, and the four items a household needs this week sit inside a hundred it does not. I wanted to say "we are going grocery shopping" and get back a short list of things we already buy that are cheaper this week, with proof from our own receipts that the price is good.

What I ruled out first

Option	Why I ruled it out
Reading the ads by hand	The status quo. It takes time nobody has, so in practice nobody does it and sales go unnoticed.
An ad aggregator service	The leading weekly-circular aggregator has no public interface. Depending on an unofficial one is fragile and raises a terms-of-service problem.
Clipping digital coupons automatically	Most grocery savings now sit inside loyalty accounts. Automating them means logging into store sites as me, which is a terms-of-service fight and a maintenance treadmill.
A full price parser built up front	I did not know yet whether store emails even carry prices. I measured first, and the measurement decided what to build (page 3).
A timer that pushes deals daily	I wanted answers when I ask, not another stream of notifications. Nothing runs on a schedule.

Design choices that shaped it

One mailbox for the whole house
A single dedicated address takes every store subscription, so personal inboxes stay clean. Loyalty accounts and personalized coupons stay on our personal email, untouched, so no rewards history is split.

Our own prices are the benchmark
A "save 30%" banner measures the store against itself. The desk measures every sale against the last price we paid for the same item, taken from itemized receipts.

No command to remember
The one question a person asks is "check the deals." That single action fetches new mail first, reads any new ad pictures and then answers. It works from a phone message, a voice puck or a keyboard.

It fits the house tools, not beside them
A running shopping list stays a shopping list, a dated chore stays a reminder, and "buy it if it ever gets cheap" becomes a watch. I refused to create a fourth place to keep the same kind of thing.

How the build unfolded



Four steps over two months. Mailbox setup began in late August as preparation. Each step started only after the one before it produced evidence on real mail.

2. How it works

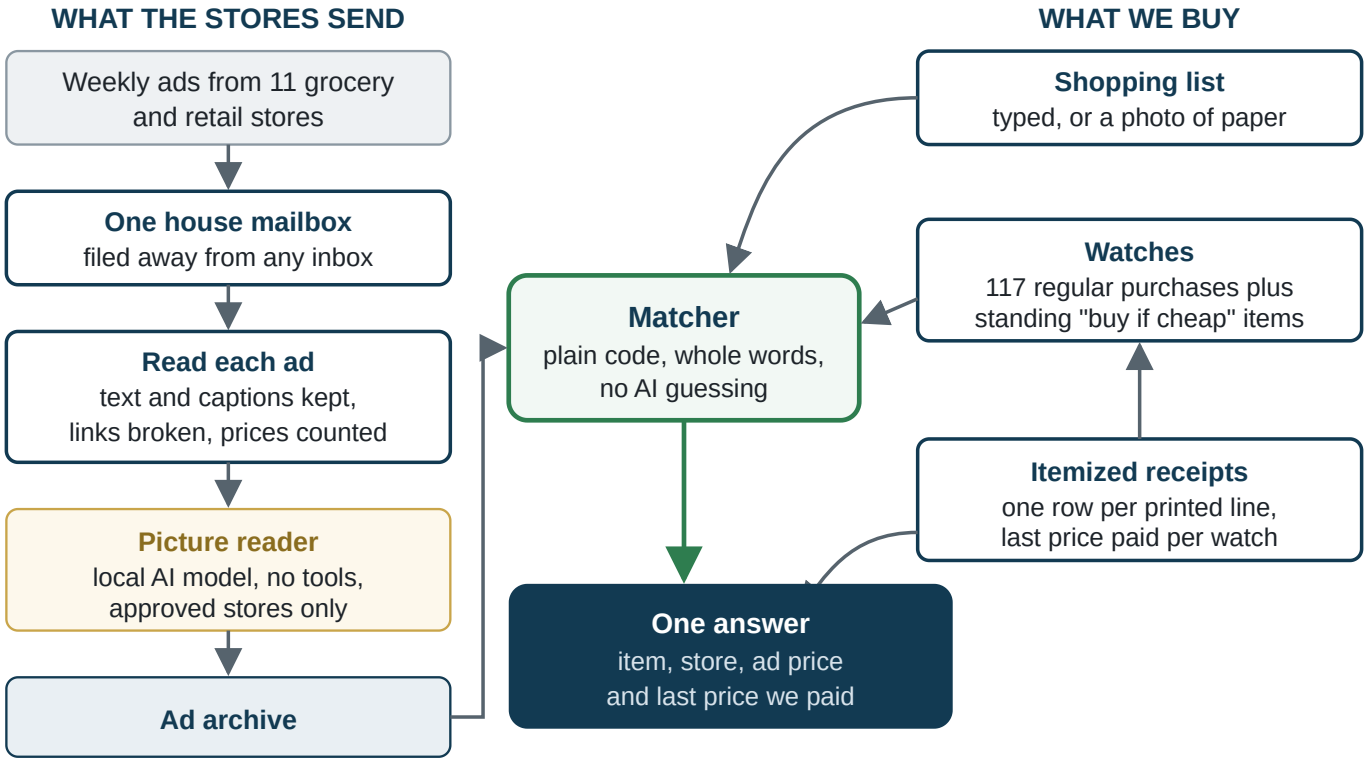
1. Ask
"Going to Costco, anything good?" by phone, by voice or at a keyboard. A photo of a paper list works too.

2. Fetch
The desk pulls new store mail into its own archive first, so it never answers from a stale week.

3. Read
It keeps the email text and image captions and reads prices printed inside ad pictures.

4. Match
Plain code matches the shopping list and every watch against the ads, word by word.

5. Compare
Each hit shows the store, the ad date, the ad price and the last price we paid.



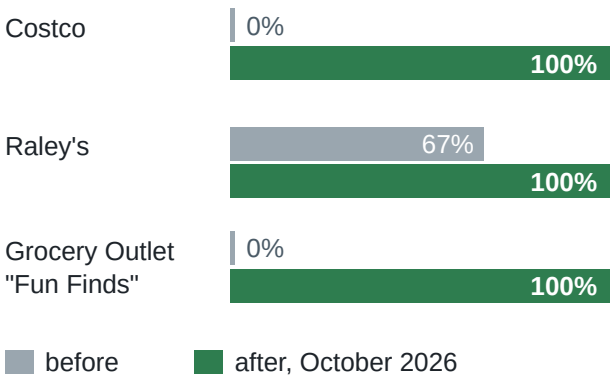
Store mail flows down the left side. Our own buying history sits on the right. The two meet only in the matcher, which is ordinary code, so the same question asked twice gets the same answer.

Measure first, then build

The first open question was whether store emails carry prices at all. The desk counted dollar amounts in every message before anyone wrote a parser. On 41 messages in September 2026, 44% carried a price as text. Some chains looked picture-only, with prices inside images or behind a link. I chose picture reading over a link-following fetcher or a text-only desk.

The build then found two bugs hiding prices in plain sight. Costco sent an empty text part beside a full web version, and Raley's wrote its offers into image captions. Fixing those came first. A local vision model running on a home workstation's graphics card then read the remaining ad pictures, 101 of them in 2 minutes 10 seconds.

Ads with a readable price



Text fixes alone took Costco to 100%. Picture reading finished Raley's and Grocery Outlet's picture-only "Fun Finds" mail.

3. The data model and the receipts

The desk keeps three kinds of record in the house database. Ads say what stores are offering. Watches say what we want to hear about. Purchases say what we paid. Joining the last two is what turns a store's claimed discount into a real comparison.



Purchases feed watches, and watches are searched in ads. The price note on a watch is a reminder for the reader, never a price we paid, so the two numbers cannot be confused.

Receipts reconciled to the penny

The first big trip covered two stores in one night and 135 printed receipt lines. An AI agent read the receipt photos and wrote one row per printed line. I set the test it had to pass before anything was stored.

1. **The lines add up.** Each store's lines sum to that receipt's printed subtotal to the penny, and the store's "you saved" lines sum to its printed savings figure. Overlapping photos therefore cannot double-count an item.
2. **Messy lines follow written rules.** Weighed items and multi-buys roll into one row. Bottle deposits fold into the item they belong to.
3. **Unclear lines go to a person.** The agent asked me to identify four lines it could not read with certainty, and I answered the same night.
4. **The import is all or nothing.** The desk checks the whole file before writing a row, and loading the same file twice stores nothing new.

What an answer looks like

EXAMPLE - INVENTED FOR ILLUSTRATION

```
deals matching what you want -- last 7 days, 18 message(s) scanned
================================================================
  coffee  (last paid $13.98 for 2 lb ($6.99/lb) at WinCo 11-02)   [watchlist]
      costco.com           2026-11-18   $10.99  $3.00
  eggs  (last paid $4.49 for 18 each ($0.25/each) at Safeway 11-09)   [watchlist]
      raleys.com           2026-11-17   $5.29
  rice   [grocery]
      safeway.com          2026-11-16   (no price in the text -- check the ad)
----------------------------------------------------------------
  Prices are what the MERCHANT claimed in an ad. Verify in store.
```

Read in plain words, Costco's coffee beats our last price and is worth buying this week, while the eggs ad is higher than what we paid. The desk puts the two numbers side by side. The decision stays with the person holding the cart.

4. Safeguards and quality checks

A mailbox that subscribes to every marketing list will also attract spam and phishing. I treated it as a deliberate honeypot from day one, so every message is data written by strangers and never an instruction. The safeguards below are built into the code, not left to the judgment of an AI model.

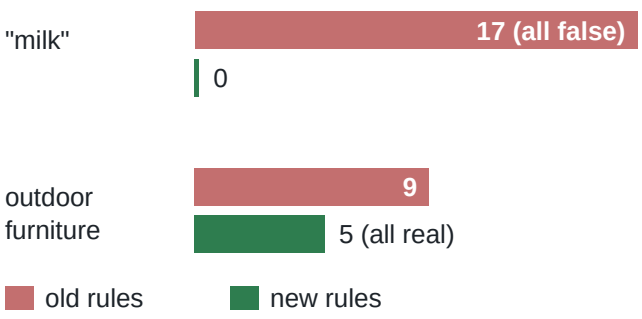
#	Safeguard	What it does
1	**A separate mailbox**	Store mail never touches a personal inbox. Loyalty accounts stay on personal email, so no rewards history is split.
2	**Mail is data, never instructions**	The desk breaks every web link on arrival, cannot send email and writes only to its own records. Nothing downstream acts on what an email says.
3	**Fields, not prose**	An answer carries only item, store, date and dollar amounts. I tested this during setup, before the desk went live, with a hostile instruction planted in a test email. Its prices came through and its instruction did not.
4	**A narrower fence on phone and voice**	The two commands that can show raw store wording or live links run only at a keyboard, where a person is present. The phone and voice assistant cannot run them.
5	**Picture fetching fenced in code**	The desk downloads a picture only from an approved store, only from that store's exact image server and only over a secure connection. It refuses redirects elsewhere, skips tracking pixels and caps file size. The vision model has no tools, and the desk keeps no images.
6	**Code matches, AI only reads**	An AI model may read handwriting on a photographed list or prices in an ad picture. Matching is ordinary code, so the result is reproducible and no model can invent a sale.
7	**Strict matching rules**	Matches need whole words, and a multi-word item must appear as a phrase. "Coffee maker" does not count as coffee, and fine print such as "excludes milk" never counts as a milk deal. A new false hit gets one word added to a table, never a model.
8	**Receipts all or nothing**	A receipt file loads only if every line passes its checks and its totals match the paper. A repeat load stores nothing twice.
9	**A person judges the price**	The desk never decides whether a deal is good. It shows the ad price beside the last price paid and labels every ad price as the store's claim, to verify in store.

The matching rules, measured

When I tightened the matching rules in October 2026, I re-ran them over every stored email and compared the results before and after. A "milk" search had matched 17 emails, and none were real milk ads. It now matches none of them. An outdoor-furniture watch had matched 9 emails because "Outdoor Fresh" dryer sheets counted. It now matches 5, and all 5 are real.

The first live run in September 2026 found 2 of 31 list items on sale. That thin result came from the ads, not the matcher. Most stores had sent only welcome mail, and I reported it that way.

Emails matched, before and after

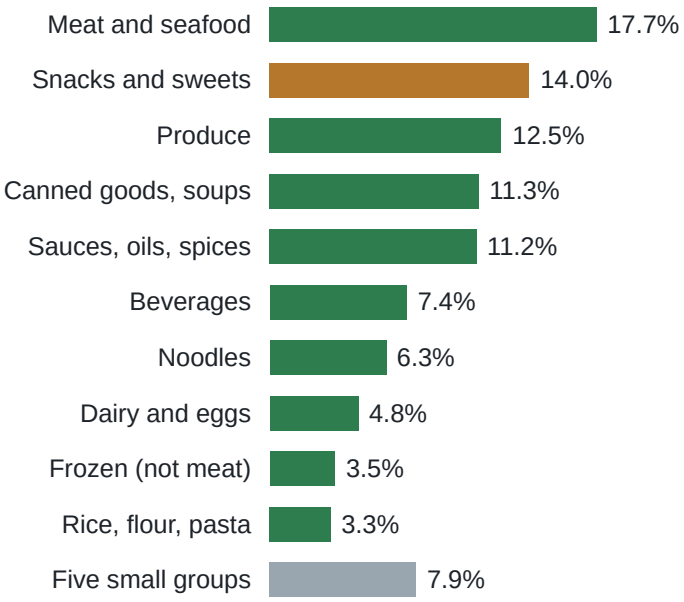


23 test cases now guard the rules against regressions.

5. What the first itemized trip taught

In October 2026 one big stock-up trip became the desk's first purchase history. Itemizing it line by line showed where the money actually went, which no store ad or bank statement shows.

Share of the trip, by category



Shares of the trip's pre-tax total across 135 receipt lines. Snacks are highlighted because they outweighed all produce. Shares do not sum to 100% because of rounding.

Four findings

- **Snacks took 14.0% of the trip**, a larger share than all fresh produce at 12.5%.
- **36.5% of the spend was not on any list**, and another 4.8% was only partly on one. 58.7% went to listed items.
- **Bulk habits drove 28.9% of the total.** Lines bought four or more at a time, or by the case, added up fast.
- **Pantry goods made up 53.4%** and the freezer 21.0%, so the trip was a stock-up, not a normal week. I judge future trips against our normal month, not against this one.

One store's loyalty discounts cut 9% off its sticker total. The other store runs no loyalty program, so its shelf price is the price.

Status and what is next

Item	Where it stands, November 2026
Deal desk	Live since September 2026 and used on demand by phone, voice and keyboard. Nothing runs on a timer.
Picture reading	Built and verified in October 2026. One store's pictures proved to be product photos with no prices, so I removed it from the picture list on that evidence.
Purchase history	One itemized trip of 135 lines, linked to 117 watches. Each new trip loads through the same checked import.
Open gaps	At the last check in October 2026, several stores had sent only welcome mail. The store where much of the first trip was bought has sent no weekly ad yet, so many regular purchases have no ad to match until it does.

6. What this shows about how I work

The desk is a small version of the work I do professionally. Eleven independent vendors each send data in their own format, and the job is to turn that into one answer a person can act on in seconds. I measured before I funded the build, so a 44% figure decided what got built rather than a guess. I wrote the rules down, fenced the parts that touch untrusted outside input and kept the judgment call with a person. AI did the reading and the coding. I owned the design, the checks and the result.