NERO PROJECT

Private Camera AI - Nothing Leaves the House

Recognition happens in the house, not in a data center.

Paul Schindler | Built with Claude | November 2026

Executive summary

Most smart cameras send footage to a company's servers and charge a monthly fee to analyze it. I built camera intelligence that runs entirely on hardware in our house. Local AI vision recognizes people, cars, deliveries and wildlife, reads license plates and writes a plain-English description of what it sees, with no cloud subscription. Every sighting goes into a permanent archive, so the house can answer questions months later that the camera software alone forgets within days. I designed the system, set its rules and verified each change; AI agents wrote the code.

20x
faster analysis on a dedicated graphics card

56
labeled frames in the vision-model bake-off

0
cloud subscriptions or monthly fees

Sep 2025
work began; on a graphics card since June 2026

What is in this report

Page	Contents
2	The problem, and the options I ruled out
3	How it works, and what the permanent archive keeps
4	Twenty times faster, and the bake-off that chose the vision model
5	Safeguards, the weekly car review, the retired alert and the intrusion drill
6	Results, status, what is next and what it shows about how I work

The rule behind every choice

What runs in the house
Detection, plate reading, face recognition, plain-English descriptions, the archive and the car review all run on equipment we own. The software is Frigate for the cameras, Home Assistant for rules and Ollama for the vision model.

What leaves the house
Footage and snapshots stay home. A household phone receives an alert only when a rule decides it should. No camera frame goes to a cloud AI service for analysis, and no vendor trains a model on our pictures.

1. The problem and why it mattered

A home camera can record everything and still know nothing. By mid-2026, ours had four problems.

Privacy
Cloud camera services analyze footage on their own servers. Even the camera software's one paid add-on, Frigate Plus, uploads snapshots to the vendor's cloud to train its model.

Speed
The vision model ran on the processor of our always-on home server. Each look at a scene took 60 to 120 seconds. On June 13, 2026 it ran out of memory and the system shut it down twice in 20 minutes, so the house got no camera analysis at all.

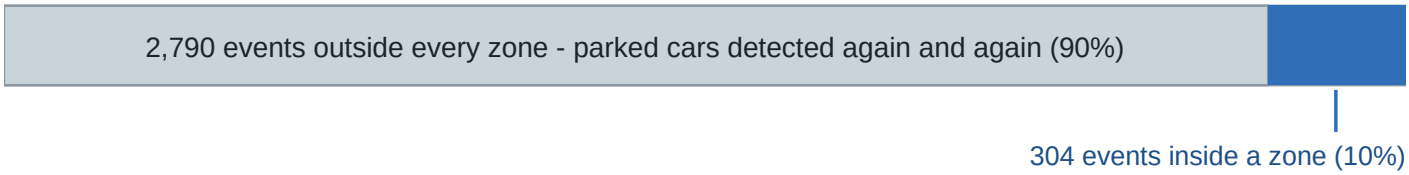
Memory
The camera software deletes its record of each detection along with the footage, within days. A question such as "which animals visit the yard, and when" had no data to answer it.

Noise
In nine days of September 2026, one camera logged 3,094 car events. 2,790 of them, or 90%, were parked cars detected again and again. A count of cars per day from that data would have been wrong tenfold.

What I ruled out, and why

Option	Why I ruled it out
A cloud AI camera service	A company would analyze our footage in its data center, for a monthly fee.
Frigate Plus, the paid add-on	Its model training uploads our snapshots. We gave up its ready-made package labels and had the local vision model describe uniforms and boxes instead.
Vision on the always-on server's processor	60 to 120 seconds per look, and it crashed for lack of memory.
Reading plates with the vision model	In the bake-off, none of five models read our own plate correctly. A dedicated local plate reader inside Frigate does that job.
Hiding busy areas from a camera to cut noise	It cuts noise by blinding the system. I merge repeat sightings into visits instead.
Asking a person to name every car	The first car review did this. Of 9 cards I answered in one sitting, 4 mixed up different cars. I replaced it with filing cars by house.
An alert for one look-alike vehicle	It sent 62 alerts in 25 days and none was confirmed. I retired it in October 2026.

One camera, nine days of car events (September 2026)



Total 3,094 car events. The fix was to merge repeat sightings into visits, not to stop watching.

What I wanted instead
A system that recognizes what it sees in seconds, remembers every sighting for good and counts each real visit once, built on hardware we already owned. New hardware appeared only as a priced option, and I bought none for this work.

2. How it works

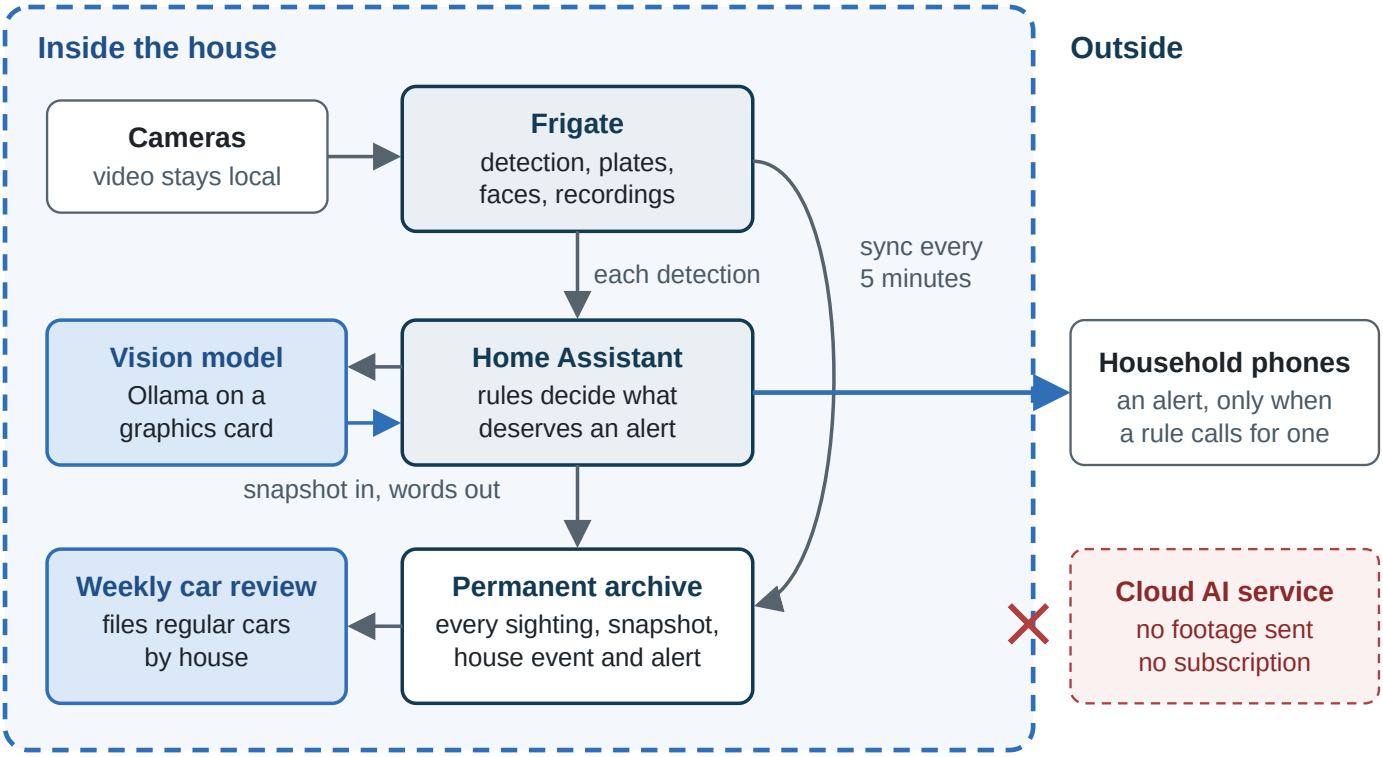
1. Detect
Frigate spots a person, car or animal, using a small local AI accelerator.

2. Recognize
Frigate's built-in plate reader and face recognition run locally, at no cost.

3. Describe
A vision model on the workstation's graphics card writes a plain-English line - a delivery, a visitor, a species.

4. Archive
Every five minutes, a sync copies each sighting, snapshot, house event and alert into a permanent archive.

5. Act
Home Assistant decides whether to alert. A weekly job files cars by house.



Everything inside the dashed line runs on equipment we own. The only arrow that crosses it carries an alert to a household phone.

What the permanent archive keeps

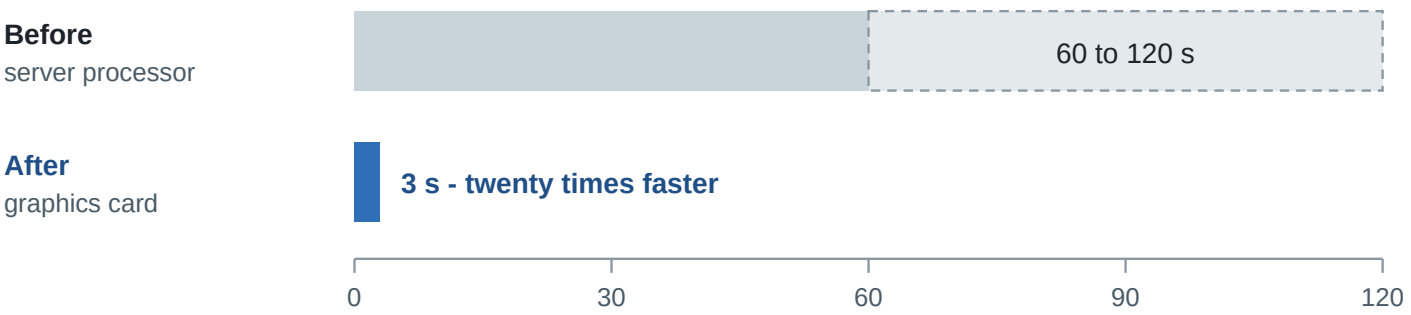
Record	What it holds, and why
Sightings	One row per detection, stored as captured and never edited. It holds object type, plate or face match, times, path, confidence, a snapshot copy and the AI description.
Visits	Repeat sightings merged into one real-world event with a dwell time. Derived from sightings, so a better merge rule can rebuild all history.
Alerts	The full text of every alert the house sends, so I can later ask what it told us and why.
House events	Doors, alarm changes and who was home, so a sighting can be read in context.
Health and changes	Camera online and offline times plus every configuration version, so later analysis knows when a rule changed.

The archive is what makes long-range questions answerable, such as which animals visit the yard and when, or how often an unfamiliar car comes back.

3. Twenty times faster, and choosing the model

The biggest single gain came from moving one thing. The vision model ran on the always-on home server's processor, which was never built for it. A home workstation in the same house has a dedicated graphics card for AI work. The AI agent that reviews the house's setup spotted the mismatch and briefed the move in plain English. We approved it.

Seconds for one look at a camera scene (June 2026)



The move also cut two models to one and freed 10 GB of memory on the always-on server. An answer that arrives two minutes late is a record, not an alert.

The bake-off (September 2026)

A faster card opened the door to newer models, so I had the agents build a bake-off rather than pick from a web survey alone. The test set was 56 frames from our own archive, each labeled with the truth. They included shrubs the camera had called a dog, a gardener, bicycles, a delivery and cars by day and by night. Five models answered every frame. Before the run I set the rule. Switch only if color and body-type accuracy rise by 10+ points.

Model	Object type right	Color and body right	Median seconds per cropped look	Graphics memory	Outcome
qwen2.5vl 7B	89%	64%	0.94	5.9 GB	The incumbent
qwen3.5 9B	96%	67%	1.19	5.2 GB	Slower
qwen3.5 4B	**95%**	**64%**	**0.55**	**3.3 GB**	**Chosen**
qwen3-vl 8B	95%	67%	1.23	6.5 GB	Largest
minicpm-v4.5 8B	82%	72%	0.65	6.2 GB	Disqualified

No model cleared the 10-point bar, so the rule said keep the incumbent. Reading the answers frame by frame showed what the averages hid. The three newer Qwen models stopped calling a shrub and a gardener a dog, which is the exact false alarm that pollutes a wildlife log. The smallest one matched the incumbent on color and body type, ran 40% faster and needed 3.3 GB of graphics memory instead of 5.9. I switched to it the next day as a separate, deliberate decision.

Why the top color score lost

The model with the best color score called black, blue and grey test squares "white" and saw a fish where the gardener stood. A score that cannot be trusted is not a win.

What no model could do

None of the five read our own license plate correctly. Plates stay with Frigate's dedicated plate reader, and vision does the describing.

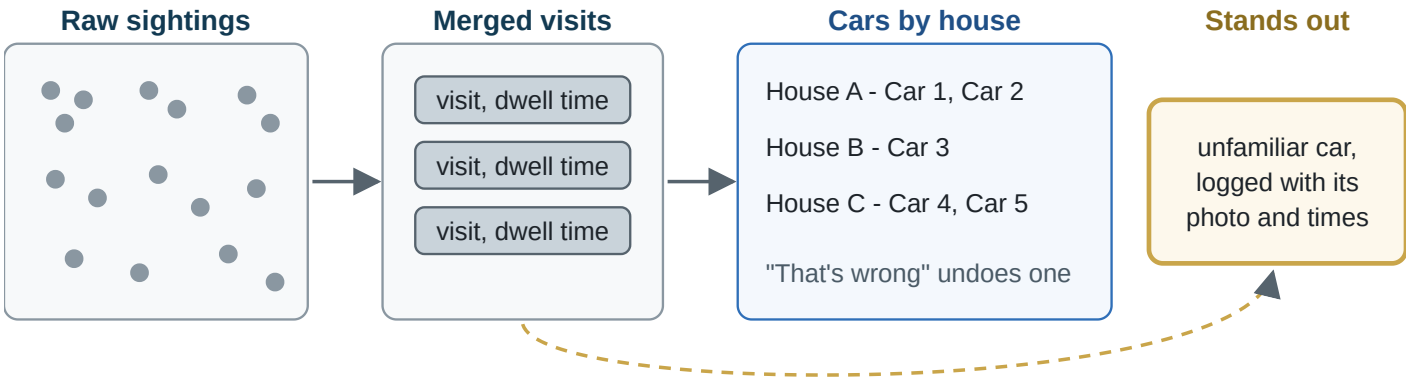
4. Safeguards, the car review and the drill

A security system that cries wolf gets ignored, and one that quietly loses data fails when it matters. These are the rules the system runs under.

#	Safeguard	How it works
1	**Local only**	No cloud AI, no subscription. The paid add-on that uploads snapshots stays off.
2	**A written brief before every live change**	Each change to the cameras or Home Assistant states what, why, risk, undo and who notices. A backup comes first, one change at a time, verified by the outcome.
3	**Raw record never edited**	Sightings stay as captured. The system rebuilds visits from them, so I fix a wrong merge rule by rerunning it over all history.
4	**Measure before and after**	I judged a camera tuning change on a 7-day baseline against the 7 days after. It made no measurable difference, so I stopped tuning and moved on.
5	**Automatic tripwire**	A consistency check runs over the visit records and reported 0 problems at the October 2026 measurement.
6	**Replay before switching**	When a crowded evening set off 58 alerts, I replayed the new rule against 62 days of archived sightings before it went live.
7	**A person can overrule**	The car page has a "That's wrong" button. It undoes a filing, and the system will not file that car there again.

The weekly car review - houses own their cars

The first version asked me to name each car the cameras kept seeing. It failed in practice. Of 9 cards I answered in one sitting, 4 mixed up different cars, and 5 of the next 10 asked about cars at houses I had already labeled. I changed the design. I label each nearby house once, and the system files every regular car that parks there under that house. It logs everything else and asks no questions, so an unfamiliar car stands out on its own. The weekly run sends no messages.



Placeholder labels only. Where a car parks says which house it belongs to, so the review never needs to know who owns it.

An alert I retired

A look-alike alert for one vehicle of interest used the vision model to match make and color. In 25 days it raised 62 alerts and none was confirmed. In October 2026 I switched it off, because make-and-color guessing cannot answer "is this the car." A plate on a watchlist is the only version worth proposing again.

A live intrusion drill

In November 2026 I staged a live drill, walking the property as an intruder would, and then tuned detection from the results. The specifics stay private, because a public map of what a security system sees is a gift to the wrong reader.

5. Results, status and what is next



September 2025 to November 2026; the break marks the months before the system's own records begin. From June 2026 on, each step went live through its own written change brief.

Part	Status
Planning and first setup	Begun September 2025, recorded in my own setup notes
Local detection, plate reading and descriptions	Live since June 2026
Vision on a dedicated graphics card	Live since June 2026. One look went from 60 to 120 seconds down to 3.
Permanent archive	Live since September 2026. By October 20 it held 12,407 sightings and 12,310 of 12,319 snapshots, with no sync errors since October 1.
Vision model chosen by bake-off	Live since September 2026 on every vision path in the house
Weekly car review, cars filed by house	Live since October 2026; redesigned four days in, after real use
Look-alike vehicle alert	Retired October 2026 after 62 alerts in 25 days, none confirmed
Live intrusion drill	Run November 2026; detection tuned from the results
A plain-English question tool and weekly digest over the archive	Designed, not built

The archive is the part I would build first again. Every sighting since mid-September 2026 sits in one place, with the alert that went out and the house events around it. That record turned a vague alert complaint into a measured fix, and it turns "I think that car has been here before" into a query.

6. What it shows about how I work

I set the binding rule before choosing any tool. Nothing leaves the house, and that rule eliminated a paid add-on and every cloud service before a line of code existed. I asked for evidence on our own data, a labeled test set and a rule fixed in advance, rather than a vendor benchmark. When a feature did not earn its keep, the look-alike alert or the car cards, I retired or redesigned it and wrote down why.

I directed AI agents to build it. They wrote the code. I set the requirements, approved each change from a plain-English brief and checked the outcome on the real cameras. The same habits carry over to an agency or partner weighing an AI tool. Write the non-negotiable rule first, test candidates on your own data and keep a permanent record of what the system did.

The work began in September 2025 in my own setup notes. All figures in this report come from the system's logs, test results and change records, dated June to November 2026. I left camera layout, coverage and alert rules out on purpose.