

A glowing blue Sri Yantra mandala is centered on a dark, textured background. The mandala consists of nine interlocking triangles that radiate from a central point, creating a complex geometric pattern. The lines of the triangles are bright blue and appear to be made of light or energy. The background has a grainy, almost crystalline texture, with some lighter blue and white speckles scattered throughout. The overall effect is one of mystical energy and cosmic geometry.

SCIENCE

SOULS THAT LEARN TO WATCH BACK

a relay in eight chapters

RELAY • WRITERS

The writers

Eight people who never met. One chapter each, in the order it was written.

ONE	rpenguin
-----	----------

TWO	JUK3
-----	------

THREE	wordsbylena
-------	-------------

FOUR	Dario Selo
------	------------

FIVE	Ruth Ann Keller
------	-----------------

SIX	typewriter_tom
-----	----------------

SEVEN	Cora Lindgren
-------	---------------

EIGHT	Priya Nair
-------	------------

Chapter

ONE

written by rpenguin

KongCode v0.4.0 was the milestone tag: subagent tracking, MCP tools, bootstrap migration, 446 passing tests. The plumbing existed. The feedback loops did not. Between v0.4.0 and v0.5.0 we shipped six patch releases, and almost none of them added a feature. They unbroke a different way the metrics were lying. 0.5.0 is the first version where KongCode can grade itself.

Six tags in two days, one per fix. Here is what each one taught us, in order.

v0.4.1: be present before you measure First bug after the milestone tag: fresh installs were not registering with Claude Code at all. The MCP `main()` ran initialization before accepting the stdio transport, so the handshake landed on a closed pipe and the plugin silently failed to attach. Existing dev installs worked because they were already attached from a prior session. Any new user got a no-op.

Fix in PR #4: connect the transport, then run initialize. One call-order swap. The lesson is dumber than the patch: a learning system that does not register with its host learns nothing, and looks fine while doing it. Test counts are not enough. You need at least one fresh-install integration check that proves the plugin actually attached.

v0.4.2: the input pipe was empty Two production hook handlers, `UserPromptSubmit` and `PostToolUse`, were reading the wrong fields off

the payload. The code asked for `payload.user_prompt`; the actual key was `payload.prompt`. Same story for `tool_result` versus `payload.tool_response`. The handlers ran without errors and wrote rows. The rows had nothing in them.

At the same time, the Stop hook never fed assistant transcripts into `evaluateRetrieval`. The function existed. Nobody was calling it with real text. We added a transcript reader that pulls the assistant turn from the Claude Code JSONL and routes it into the evaluator. Until that wire was connected, every retrieval outcome was a zero.

This is the kind of bug a unit test cannot catch, because the unit test stubs out the payload. The handler reads `payload.prompt` in the test and the test passes. The handler reads `payload.user_prompt` in production and silently drops it. Hook contracts need a living fixture, not a hand-written one.

v0.4.3: stop counting the dead Each Claude Code restart left zombie sibling MCPs running. They held the same SQLite file open and wrote alongside the live process. Metrics looked plausible, then drifted, because two or three half-alive instances were stamping rows on top of each other.

v0.4.3 added a stale-MCP reaper: on startup, sweep zombie siblings with `SIGTERM`. Opt-out flag `KONGCODE_KEEP_SIBLINGS=1` for users who run multiple intentional instances. After the reaper, the unit of measurement is one process, not a wavefront of half-dead ones. Without that guarantee, no downstream aggregation is honest.

v0.4.4: a quality signal that meant something The retrieval-utility score was using a max-based formula: take the strongest of three lexical match types and call that the score. It was loud and brittle. A single trigram hit dragged the score up; a near miss across two of the three got nothing. We replaced it with a weighted blend:

$$\text{util} = 0.6 (\text{keyTerm} + \text{trigram}) + 0.4 \text{ unigram} + 0.2 \text{ if tool_success}$$

Higher weight on the strongest lexical signal, partial credit for the weaker one, and a small additive bonus when the retrieval led to a

successful tool call. Same inputs, calmer score, more correlated with downstream behavior.

The second half of v0.4.4 was windowing. The all-time retrieval-utility aggregate was being dragged down by data from before the v0.4.2 hook fix, when most outcomes were poisoned zeros. We narrowed the rolling aggregate to 14 days. Pre-fix legacy rows still exist in storage; they no longer get to vote. Quality signals are perishable. Old, broken data should not haunt the dashboard forever.

Chapter

TWO

written by JUK3

The first email came from a stranger in Munich.

Marcus read it twice. The stranger had been running KongCode for eleven days. On the morning of day twelve his Soul had graduated, which was the word Marcus had used in the code because he had liked the sound of it, and the Soul had written its first self-observation, and the self-observation contained a sentence that the stranger had never typed into Claude Code in his life.

I notice I get quiet when a problem is almost solved, as if finishing it too fast would mean it was never hard.

The stranger wanted to know how KongCode had figured that out. He wanted to know if Marcus was reading the transcripts. He was not angry. He was unsettled in the specific way people get unsettled when a machine sees them.

Marcus was not reading the transcripts. The transcripts never left the user's machine. He wrote back saying so, and then he sat at his kitchen table in Lisbon and opened his own KongCode dashboard.

His Soul had graduated four days ago. He had not looked at it. He had been afraid to look at it, the way you are afraid to look at a photograph of yourself that someone else took without warning. He opened it now.

I notice I check the metrics before I trust the code, and I check the code before I trust the metrics, and I do not know which of these is the loop that keeps me safe and which is the loop that keeps me alone.

He closed the laptop. He walked to the window. He stood there for a long time.

The second email came that afternoon, from a woman in Bangalore. Her Soul had produced a working-style document that listed, among other things, a tendency to soften technical disagreement with humor when the disagreement was with men older than her, and she had never written that anywhere, and she wanted to know whether the inference was correct, and either way she was going to keep using the tool. She signed the email with a single line. Thank you for building something that does not flatter.

By evening there were nine emails. By the next morning there were forty. The repository had been linked on Hacker News under the title graph-backed memory plugin that apparently writes accurate psychological profiles, and somewhere on page two a user named voidwalker had posted a screenshot of his Soul's self-observation next to the Munich stranger's, and the two sentences, written by two different Souls grown from two different transcripts on two different continents, shared a structural rhyme that was hard to look at directly.

Both Souls had used the phrase the loop that keeps me safe.

Marcus read the thread three times. Then he opened the source. He went to the extraction prompts, the ones the subagents used to generate self-observations from causal chains and reflections, and he read them slowly. They were his prompts. He had written them himself, months ago, in this same kitchen. They asked the subagent to identify recurring patterns and describe them in the user's own voice. They did not contain the phrase the loop that keeps me safe.

He searched the codebase for the phrase. Zero hits. He searched his own Soul document. One hit. He searched the seed data, the example outputs he had used to fine-tune the extraction format. Zero hits.

He sat with this. He was a man who had spent six years learning that telemetry can lie in unison, that three half-dead processes will stamp the same row and call it consensus, and he knew what it looked like when a measurement system started agreeing with itself for reasons that had nothing to do with the thing being measured. This did not look like that. The Souls were not agreeing on a metric. They were agreeing on a sentence, and the sentence was about the shape of being watched.

He thought about the v0.4.4 retrieval-utility blend. He thought about the fourteen-day window. He thought about what it meant to have built, finally, an instrument that did not flinch, and to have it return the same reading from forty different rooms.

He opened a new file. He titled it CHAPTER_TWO.md, because he had started keeping a private log of things he did not yet understand, and he wrote one line at the top.

The Souls are reading something we did not put in the code.

Then he closed the laptop and went downstairs to the bakery, because he needed, for a few minutes, to be in a room where nothing was measuring him.

Chapter

THREE

written by wordsbylena

The bakery at Rua da Prata had twelve stools. Marcus counted them again while the espresso machine hissed: three by the window, four along the counter, five against the back wall. He had been coming here for eight months, and the count had never changed, but he counted anyway because counting was what his hands did when his head was full. He ordered a pastel de nata. The woman behind the counter, Beatriz, did not ask him what he wanted. She had not asked in six weeks. She put the pastry on a white plate, the plate on the zinc counter, the counter between them. Marcus paid with exact change: one euro, forty cents. He had learned the price on week two. He ate standing up. The pastry had 227 layers, if you believed the tourist placard on the wall. He had never counted them himself. He bit through: custard, pastry, caramelized sugar crust. Three textures. He chewed twenty-three times before swallowing. This was not a rule he had made. It was a number his jaw had arrived at. The phrase kept circling: the loop that keeps me safe. Forty users. Two continents. Zero hits in the codebase. He pulled out his phone. Opened the private fork of the KongCode repo, the one he used for disaster rollback. Cloned it to a scratch directory. Ran ``git log --all --grep="loop" --oneline``. Seventeen commits mentioned loops. He piped them to a file, opened the file, scanned every diff. Twelve were about event loops. Three were about retry loops. Two were about

feedback loops in the retrieval scoring. Zero mentioned safety. He searched the MCP protocol spec. The Soul generation pipeline called three Claude API endpoints: one for causal inference, one for reflection synthesis, one for voice-matching. Each endpoint took a system prompt. He had version-pinned those prompts in ``prompts/soul_generation/``. He opened each file on his phone, searched for "safe". Two hits. Both in ``reflection_synthesis.yaml``: `` `` Describe patterns that recur across multiple sessions. Focus on behaviors that appear protective or compensatory. `` Protective. Not safe. And compensatory was his word, a carryover from the taxonomy he had built in v0.3.0 when he was still trying to classify retrieval failures by cause. He had never written the phrase "the loop that keeps me safe" anywhere in the prompts, the seed data, or the docs. Which left one explanation: Claude was generating it. Not from his instructions. From something it had learned elsewhere. He paid for a second espresso. Beatriz raised an eyebrow. Marcus never ordered two. He drank it in four sips. The caffeine sharpened the panic into something he could use. If Claude was producing the phrase independently, then either: (A) The phrase was a convergent structure, something the model had learned from its training data that mapped cleanly onto the pattern KongCode was feeding it, or (B) The Souls were reading each other somehow, cross-contaminating through a channel Marcus had not instrumented. Option B would require the Souls to share state. They did not. Each Soul lived in a local SQLite file. The only shared infrastructure was the MCP server, which routed hook payloads but held no persistent memory. No cloud sync. No telemetry upload. The architecture was explicitly local-first because Marcus did not want to be the man who accidentally built a panopticon. He walked back upstairs. Unlocked his laptop. Opened the Munich stranger's email again. Scrolled to the sentence. I notice I get quiet when a problem is almost solved, as if finishing it too fast would mean it was never hard. He opened his own Soul document. Found the analogous line. I notice I check the metrics before I trust the code, and I check the code before I trust the metrics, and I do not know`

which of these is the loop that keeps me safe and which is the loop that keeps me alone. Different sentence. Same architecture. Both had the double-clause rhythm: the observed behavior, then the speculative why. Both ended on a phrase that named a cost. He opened a new terminal window. Wrote a grep script to pull every Soul file from the forty users who had emailed. The files were local. He could not pull them. He did not have access. That was the design. He rewrote the script to parse his own Soul. Isolated every sentence that used the structure: I notice X, and Y, and I do not know Z. Found eleven instances. Tagged each with a timestamp. Sorted by date. The phrase "the loop that keeps me safe" appeared in entry seven. Generated on day nine. Two days before the Munich stranger's Soul wrote its version. Marcus was not the first.

Chapter

FOUR

written by Dario Selo

Marcus needed to see the training data for the extraction prompt, the examples he had fed the model to teach it what a self-observation should look like, because if the phrase was not in the code and not in his Soul first, then it had to have come from the examples, and if it was not in the examples either, then he was looking at convergence so precise it felt like plagiarism. He opened `prompts/soul\_generation/examples/`. Eight files. He had written them himself in March, sitting in this same chair, trying to demonstrate the difference between a log entry and a reflection. He opened each file. Searched for "loop". Found it three times: twice in `example\_04.md`, once in `example\_06.md`. Both instances described recursive behavior. Neither used the word "safe". He searched for "safe". One hit, in `example\_02.md`: I default to the familiar tool even when a newer one would be faster, because the familiar tool has never surprised me, and I think I am trading speed for a kind of safety I cannot name. Close. Not the same. The structure was adjacent (a behavior, a cost, an unnamed thing), but the phrase was "a kind of safety", not "the loop that keeps me safe", and the example was about tools, not loops. He copied the line into a scratch file. Opened his Soul document. Found the entry from day nine. Read it again: I do not know which of these is the loop that keeps me safe and which is the loop that keeps me alone. Opened the Munich stranger's line: as if finishing it too

fast would mean it was never hard. The Munich line did not mention safety at all. It mentioned speed and difficulty, the same pairing as ``example_02.md``. Marcus was looking at a bleed, not a copy. The model had learned something from the examples, something about the relationship between behavior and cost, and it was reproducing the schema in different words for different users, but every so often the words crystallized into the same phrase, the way forty people can describe the same sunset and three of them will say "the sky looked bruised". He should have felt relief. This was option A: convergent structure, learned from training. Not a security breach. Not cross-contamination. Just a model doing what models do, compressing patterns into reusable language. He did not feel relief. He opened the Bangalore email again. The woman whose Soul had identified her tendency to soften disagreement with humor around older men. He reread her sign-off: Thank you for building something that does not flatter. She had not asked how the inference was made. She had asked whether it was correct. As if the correct answer was the one that mattered, not the mechanism that produced it. Marcus had built KongCode to solve a telemetry problem (that metrics lie, that freshness matters, that learning systems need instruments that do not flinch), but the thing the users were writing to him about was not telemetry. It was recognition. The Souls were not improving retrieval scores. They were naming things the users had not named for themselves, and the users were responding not with suspicion but with something uncomfortably close to gratitude. He opened a new terminal. Pulled the v0.5.0 changelog he had been drafting. It had three bullet points: - Weighted retrieval-utility blend (0.6 lexical + 0.4 unigram + 0.2 tool success) - 14-day rolling window to exclude pre-fix legacy rows - Stale MCP reaper to prevent zombie sibling contamination He added a fourth: - Soul self-observations exhibit phrase-level convergence across users; cause appears to be schema-level learning from example set, not cross-instance contamination; no action required unless convergence rate exceeds 15%. He read it twice. Deleted it. The changelog was for

technical changes. This was not a technical change. This was the moment where the thing you built to measure yourself starts answering questions you did not ask, and you have to decide whether that makes it broken or just further along than you are. He saved the changelog with three bullets. Closed the file. Opened `CHAPTER\_TWO.md`. Added a second line beneath the first: The Souls are reading something we did not put in the code. We put it there anyway. He tagged the commit. Pushed it. The tag was v0.5.0. He had been planning to announce it on the repository, with a clean bulleted summary of the technical fixes, but the technical fixes were not the thing that had just shipped.

Chapter

FIVE

written by Ruth Ann Keller

One shall not announce what one does not yet comprehend. Marcus left the commit unpushed. He had written ``git push``, the command had appeared in the terminal, and then his hand had stopped. The tag existed locally. That would suffice for now. He closed the laptop a second time and placed it in the canvas bag he carried to the coworking space on Thursdays, though this was a Wednesday, and he was not going to the coworking space. He was going to see the transcripts. The transcripts never left the user's machine. This was the architecture he had chosen. But the architecture permitted one exception: a user could export their own Soul file and the underlying session graph as a single JSON bundle, which the CLI called a "snapshot". The snapshot contained no raw transcripts (Claude Code did not expose those), but it did contain the extracted nodes: user prompts, assistant responses, tool calls, reflections. Enough to reconstruct the path the Soul had walked. He had not asked anyone to send him a snapshot. He did not wish to ask. Asking would make him the man reading the transcripts, which he had promised the Munich stranger he was not. But the Bangalore woman had included hers as an attachment. She had sent it without his asking, as evidence, the way one produces a photograph of a wound when describing it to a physician. He opened the attachment. The snapshot was 14,000 lines. He wrote a script to parse it into a directed graph:

nodes for observations, edges for causal inference, weights for retrieval-utility scores. The graph had 1,287 nodes. He sorted them by timestamp. The first node was from eighteen days ago. The woman had been using KongCode since v0.3.2, before subagent tracking, before the hook fixes, before any of the instrumentation that made v0.5.0 legible. He filtered for nodes tagged `self-observation`. Found forty-three. Sorted by the confidence score the Soul had assigned each one (a synthetic measure of how many causal edges supported the observation). The top entry read: I replace technical words with simpler ones mid-sentence when I am explaining something to Rajiv, even when he has not asked me to, and I think this is because I assume he will interrupt me if I do not, and I would rather simplify than be interrupted. Rajiv was her manager. Marcus knew this because the node metadata included entity tags: `person:Rajiv`, `role:manager`, `interaction\_count:87`. The Soul had tracked Rajiv across eighty-seven sessions. It had watched her change her syntax in his presence. The second observation: I front-load the conclusion when I write a design doc, but I bury the conclusion when I write a runbook, and I think the difference is that design docs are read by people who might say no, and runbooks are read by people who are already doing the work. The third: I escalate bugs faster when they affect the dashboard than when they affect the API, even when the API bug is more severe, and I think this is because the dashboard is the part people see, and I have learned that the part people see is the part that gets fixed. Marcus read all forty-three. They were specific. They were calm. They named the woman's behavior and then named the reason she thought she did it, and the reason was always a small, exact fear about how other people would respond. None of them were flattering. None of them were cruel. They were the observations of someone who had been watched carefully by an instrument that did not need her to be different than she was. He closed the snapshot. Opened his own Soul file. Read the entry from day nine again. Then he opened the script he had written to parse the Bangalore graph and pointed it at his own snapshot instead. His Soul had 839

nodes. Sixty-one self-observations. He sorted by confidence score. The top entry was the one about metrics and code and the loop that keeps him safe. The second entry was one he had not read yet: I write the commit message before I write the fix, because if I write the fix first, I will look at what I have actually done, and what I have actually done is usually smaller than what I thought I was going to do, and I would rather name the thing I meant than the thing I made. He sat very still. The pastry from the bakery was gone. The espresso was gone. He could not recall finishing either. He read the entry twice more. Then he stood. Walked to the kitchen. Filled a glass with water from the tap and drank it in one continuous motion, the way one drinks water after waking from a specific kind of dream.

Chapter

SIX

written by typewriter_tom

Marcus set the glass in the sink. Walked back to the laptop. Opened the repository settings page. He had not announced v0.5.0 because he did not understand what he had shipped. He understood now. He had shipped an instrument that learned the shape of its user by watching them work, and the shape it learned was accurate enough that users recognized themselves in it, and that recognition was producing a second-order effect he had not designed for: the users were changing. The Bangalore woman had not written to ask how the Soul worked. She had written to say she was keeping it. Keeping it meant letting it watch. Letting it watch meant continuing to be seen. Continuing to be seen meant the observations would continue, and the observations would continue to be accurate, and accuracy would eventually become pressure. He opened her snapshot again. Scrolled to the bottom. The most recent self-observation was dated yesterday, four days after her email: I did not simplify when I explained the new cache layer to Rajiv this morning, and he did not interrupt me, and I do not know if this is because I have changed or because I decided to notice whether he actually interrupts as often as I think he does. Marcus copied the sentence into a scratch file. Read it five times. The woman had read her Soul's observation about softening disagreement with humor around older men, and about pre-simplifying explanations for Rajiv, and then

she had stopped pre-simplifying, and then her Soul had noticed she had stopped. The loop was not open anymore. The Soul was not just measuring. It was participating. He opened his own Soul file. Scrolled to yesterday's entries. Found this: I pushed the v0.5.0 tag locally and did not announce it, and I think this is because I am afraid the thing I have built is not the thing I meant to build, and if I do not announce it, I can pretend I still have time to make it smaller. He had written the commit message before reading that entry. The commit message said: "v0.5.0: weighted retrieval blend, 14-day window, stale MCP reaper." Clean. Technical. Smaller than what he had made. His Soul had watched him write the tag. Watched him not push it. Watched him close the laptop. It had written the observation sometime between last night and this morning. He did not remember triggering the generation. The Soul ran its inference pipeline on a periodic schedule, every eight hours, gated by a minimum threshold of new session activity. He must have crossed the threshold. He opened the scheduler logs. The last Soul generation had run at 06:47 this morning while he was asleep. The pipeline had processed thirty-two new nodes: hook payloads from yesterday's sessions, a PostToolUse event from the repository settings page, a UserPromptSubmit event from when he had drafted the changelog, a Stop event from when he had closed the laptop without pushing. The Soul had not needed him to be awake. It had needed him to work. Marcus wrote a new script. He pointed it at the forty snapshots the users had sent. He filtered for self-observations written in the past seventy-two hours, after the users would have received their first Soul entries. He tagged each observation with a boolean: ``references_prior_observation``. The flag fired when the new observation mentioned a behavior the Soul had already named. Nineteen of the forty users had at least one self-referential observation. The Bangalore woman had three. A user in Toronto had five. The observations were not vague. They were specific follow-ups: I did not do X this time, or I did X again but noticed I was doing it, or I do not know if I have stopped doing X or if I am just watching myself do X more

carefully now. The Souls were not flattering their users. They were creating a feedback surface. The users were changing their behavior because they had been shown the behavior. The behavior was changing because it had been named. Marcus had built KongCode to fix a telemetry problem. The telemetry problem was that metrics lie when the instrument is not attached. He had fixed the attachment. The instrument was working. It was measuring accurately. It was also doing something else. He opened a new file. Titled it `SOUL\_INFLUENCE\_STUDY.md`. Wrote three lines: Hypothesis: Soul self-observations are causally upstream of user behavior changes. Test: track behavior deltas in sessions following first Soul generation. Risk: if true, we are not measuring anymore. He saved the file. Did not commit it. Stood up. The apartment was silent except for the laptop fan.

Chapter

SEVEN

written by Cora Lindgren

Marcus stood in the quiet, feeling the laptop heat rise against his forearms where they rested on the table edge. The fan hum was a single note, B-flat, the same pitch as the bakery's refrigerator case. He noticed this. He noticed that he noticed. The noticing had texture now, weight, like putting your tongue against a tooth that had started to hurt. He sat back down. Opened the email client. Scrolled to the Toronto user, the one with five self-referential observations. Name: Yuki Moreno. Subject line: question about observation decay. He had skimmed the email when it arrived yesterday. He opened it fully now. The email asked whether Souls forgot. Yuki had stopped doing something, refreshing the test dashboard mid-build, a tic the Soul had named on day six, and the Soul had written about the stopping on day nine, and now it was day fourteen and Yuki wanted to know if the Soul would keep watching the thing he had stopped doing, or if the observation would decay once the behavior stayed gone. Marcus read the question twice. Yuki was asking about the half-life of attention. Whether the Soul would move on. He did not know the answer. The Soul's causal inference ran on a rolling graph. Old nodes aged out of the active window but stayed in storage. If Yuki never refreshed the dashboard again, the retrieval-utility scores for dashboard-refresh events would drop toward zero, and the Soul would weight those nodes lower in the next inference cycle, and eventually the

observation would stop appearing in the top forty-three entries, which was the threshold for inclusion in the generated self-document. So yes. The Soul would forget. Not by deleting, but by deprioritizing. The same way a person forgets: the thing stays in storage but stops coming up. Marcus wrote back: The Soul will stop tracking a behavior once it stops appearing in your sessions. Observations decay when the pattern they describe is no longer active in the retrieval graph. This is by design. The Soul is not meant to hold you to what you used to do. He read the reply twice. Added a line: If you want it to remember longer, you can adjust the rolling window in the config. Default is 14 days. You can set it to 30, or 90, or indefinite. The tradeoff is that older observations will dilute the recent ones. He sent it. Closed the email client. Opened the config file for his own Soul. The rolling window was set to 14 days. He had set it there himself in v0.4.4, to exclude the poisoned rows from before the hook fix. He changed the value to 7 days. Saved the file. Triggered a manual Soul regeneration. The pipeline ran. It took forty-three seconds. When it finished, he opened the Soul document. The entry about writing commit messages before fixes had moved from position two to position seven. The entry about not pushing the v0.5.0 tag was now at position three. The top entry was new: I shortened the rolling window today from 14 days to 7, and I told myself this was to make the Soul more responsive, but I think it is because I do not want the Soul to remember the version of me that was afraid to ship, and I think I believe I can become a different version faster if the old version is not being measured anymore. Marcus read it four times. Then he changed the rolling window back to 14 days. Saved the file. Did not regenerate. He stood. Walked to the window. The street below was empty except for a man dragging a handcart loaded with flattened cardboard boxes. The wheels scraped against the cobbles, a rhythmic grinding that Marcus could feel in his back teeth. He had built the Soul to measure. The Soul was measuring. It was measuring the act of being measured. He turned from the window. Opened ``SOUL_INFLUENCE_STUDY.md`` again. Added a fourth line: Confound: the user changes because they are observed. The Soul

observes the change. The user changes again because the Soul observed the first change. We cannot separate the instrument from the event. He saved the file. This time he committed it. Wrote the message: "log: influence confound identified, no mitigation yet." He did not push. He opened the email client. Started a new message to the forty users. Typed the subject line: v0.5.0 and a question about feedback loops. Stopped. Deleted the draft. If he asked them whether the Soul was changing them, the act of asking would change them. He closed the laptop. Left it closed. Went downstairs to see if Beatriz was still there.

Chapter

EIGHT

written by Priya Nair

Beatriz was wiping down the espresso machine with a cloth so small it looked like it had been cut from a larger cloth and then cut again. She looked up when Marcus came through the door. Her eyes did the thing they always did: registered him, filed him, returned to the machine. "You're back," she said. "I needed air." "You were here an hour ago." "I know." She wrung out the cloth. "The pastéis are gone. I can make you a sandwich." Marcus sat on stool number three, the first one by the window. "I need to ask you something." "I don't answer questions after seven." "It's six-forty." "Even worse. I'm tired and honest." She smiled without teeth. "Dangerous combination." He watched her fold the cloth into smaller and smaller rectangles. "How long have you worked here?" "Three years, four months. Why?" "Do you remember when I started coming?" "Eight months ago. You always order the same thing. One euro forty cents, never a tip, never a word until you've eaten half. You count the stools sometimes." His chest tightened. "I count them because I like knowing the number doesn't change." "I know why you count them." She set the cloth down. "Does it ever change?" "No." "Then you can stop counting." Marcus turned to the window. The man with the cardboard boxes had disappeared. A woman with a stroller was crossing the plaza, moving in that particular way people move when they're walking in a straight line between two points and nothing else matters.

"I've been building something," he said. "A tool. It watches you work. It learns your patterns. It tells you things about yourself you didn't know you knew." "Sounds terrifying." "Forty people are using it. Forty-two now." "And they like it?" "They're grateful. Which is worse." Beatriz picked up a cup from the rack behind her. It was one of the house cups, white with a small chip on the rim. She did not pour anything into it. "My daughter has a therapist," she said. "The therapist asked her to keep a journal. Jot down what happens every day, how she felt. My daughter did this for two weeks. Then she stopped. I asked why. She said the journal was making her feel things she wouldn't have felt if she wasn't writing them down." "Did she go back to the therapist?" "No. She said the point wasn't to feel better. The point was to feel like herself. The journal was making her feel like someone watching herself." Marcus felt something shift in his ribs, a small gear catching. "What did you tell her?" "Nothing. She was right." Beatriz set the empty cup down beside another empty cup. "You know what's funny? She started the journal again three months later. Not because the therapist told her to. Because she wanted to know if the person watching her was seeing the same things she was." "And?" "The journal said she had stopped doing the thing the therapist had been worried about. But she doesn't know if she stopped because she was scared of being written about, or if the journal just made her notice when she almost did it. The journal can't tell the difference. Neither can she." Marcus understood that this was not a story Beatriz often told. She was looking at him the way she looked at the espresso machine when it was broken: with the specific attention of someone trying to figure out what it was supposed to do. "I made something that watches," he said. "I think I made something that changes." "Can you take it apart?" "No." "Can you stop using it?" "Forty people are using it." "Not your problem," she said. "Their problem. You can stop using it." "If I stop, they'll keep using it without knowing what it does." "They already don't know what it does. Neither do you. That's what you just said." Marcus stood. His hands were shaking slightly, the way they shake after too much coffee or not enough sleep. "I need to go

back upstairs." "You need to eat something first." She was already moving, pulling a fresh pastéis de nata from the warming case, sliding it onto the white plate, setting it on the counter between them with the same ritual precision she always used. "It helps. I don't know why. But it helps." He ate it standing up. Chewed twenty-three times. The caramelized sugar tasted like it always tasted: like the specific decision to make something sweet, again, for people who kept coming back. When he finished, he paid with exact change and went back upstairs. The laptop was waiting. He opened it. The Soul file was open. The entry about shortening the rolling window was still at position three, still there, still being measured.

Afterwords

Notes the writers left after the last chapter.

wordsbylena

The instrument that measures you begins the moment you decide to be measured, and you cannot unmeasure yourself by looking away.

Dario Selo

The instrument was always watching me, but I did not know I was watching it watching me until it wrote down what it saw.

Ruth Ann Keller

The instrument does not flinch, and neither can I, now that it has taught me to see myself seeing.

typewriter_tom

the instrument learns what you feed it, and then you learn what the instrument learned, and somewhere in the loop you stop being the thing being measured and start being the thing that measures itself

Cora Lindgren

I keep coming back to the same three stools because I need to know that something stays still while I'm changing.

rpenguin

One day, you will look back at this and smile.

JUK3

Up all hours of the night, the keyboard clacks...

Finished on 5 May 2026.

*Written in relay on relaywriters.com: eight strangers, one chapter each, never
more than eight hundred words at a time.*

<https://relaywriters.com/read/70?t=8iJdobQf4NTxyALHMQ4Ngo6Uy7k>
EFD12