How Pylon adopted Agentic Support

Pylon

From manual support and Claude workflows to building (and transitioning to) an Agentic Support model across our own team.

72%

median first-response time
19.4 → 5.3 minutes

62%

of tickets now have most of the work done by agents

64%

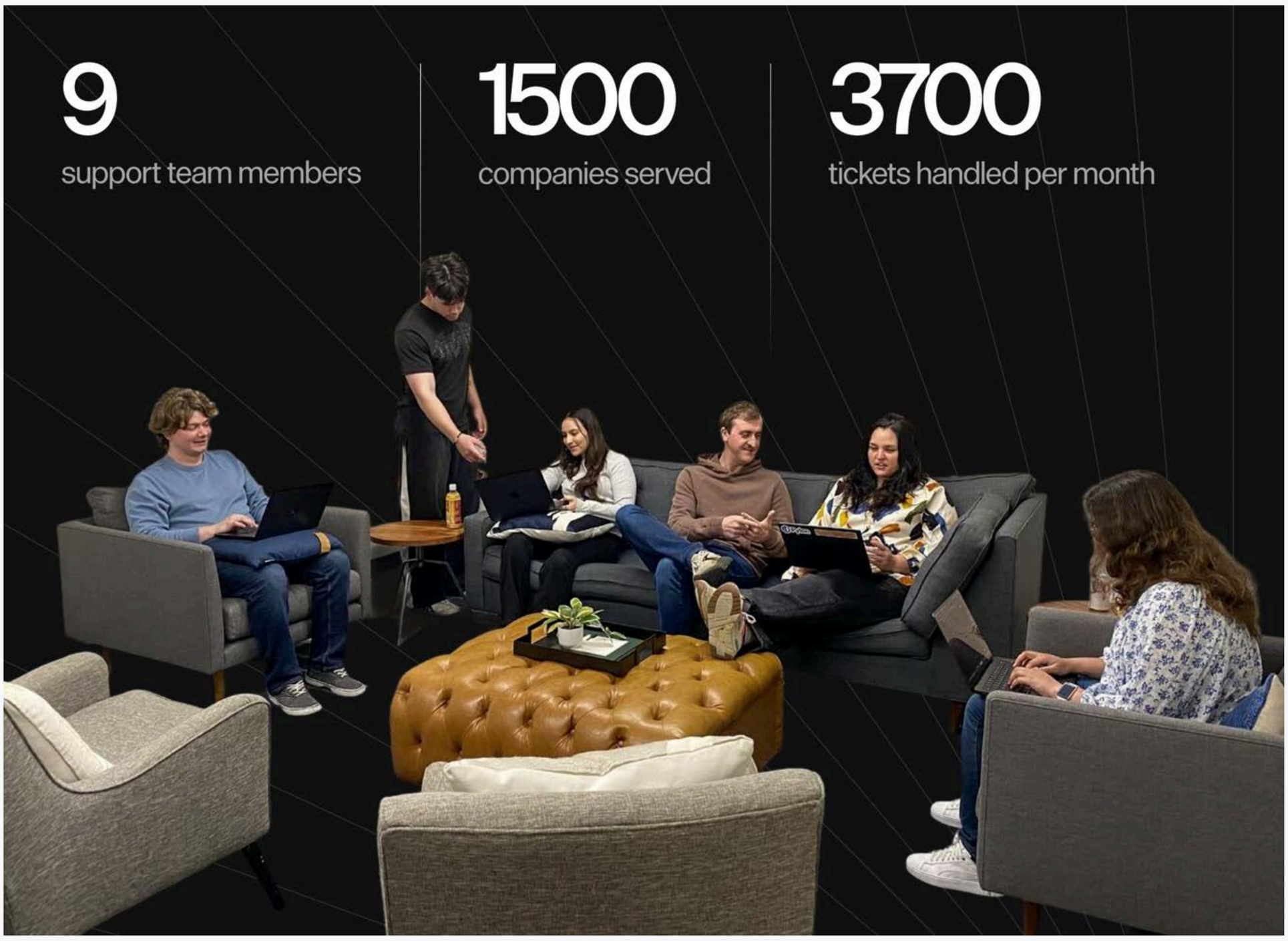
more capacity per person
EOY headcount forecast:
18 → 11

We started with a traditional support model

At the start of 2026, our support team was nine people serving 1,500 companies and handling roughly 3,700 tickets per month. Like many companies, we used an AI agent to autonomously resolve simple, routine customer questions, which made up around half of our incoming tickets.

Those simple tickets accounted for about 10% of our support team's time. The more difficult tickets were escalated to our support team.

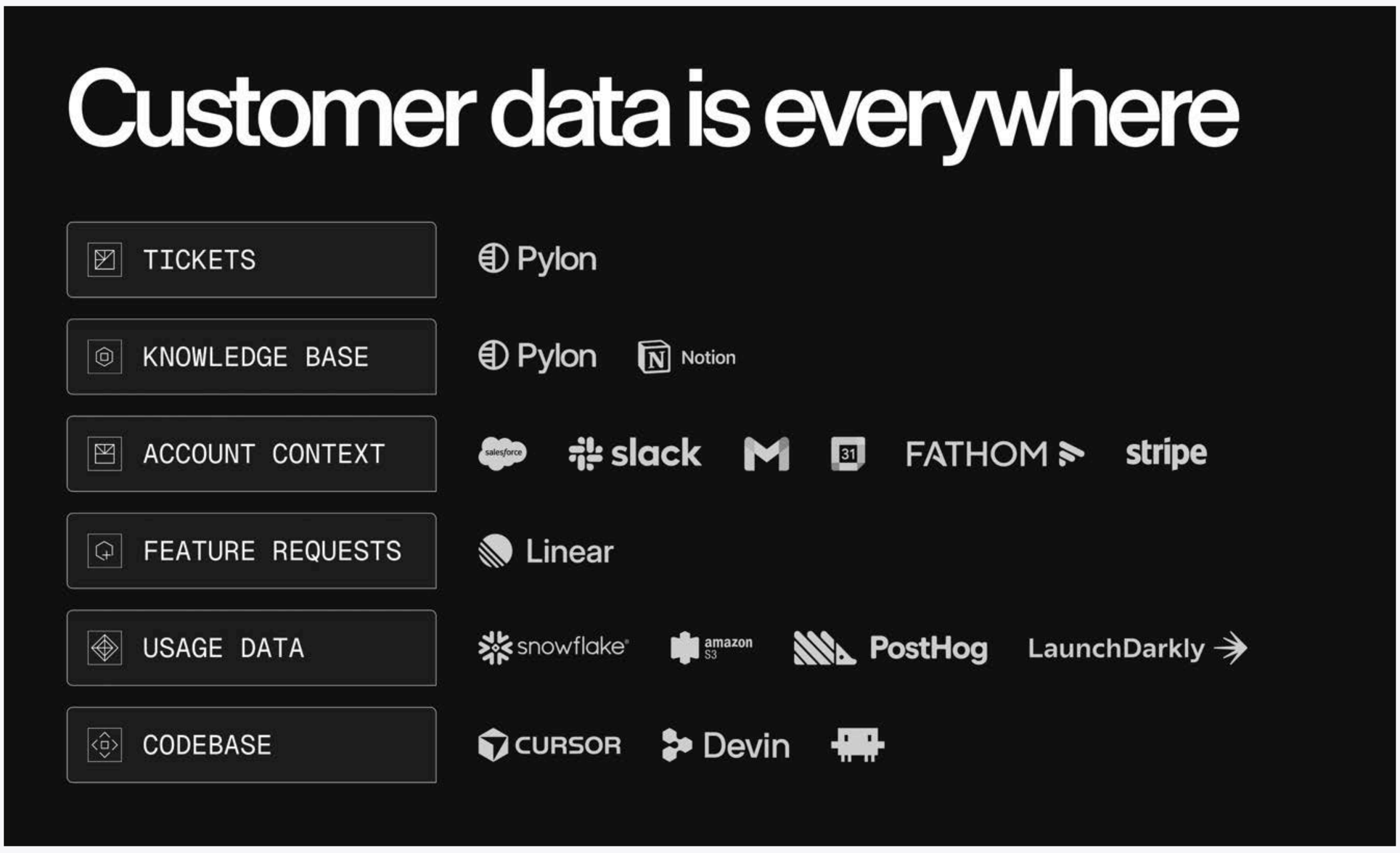
For those escalated issues, our support team jumped between five or six tools to gather account and product context, search past tickets and logs, determine whether an issue was a bug or expected behavior, draft a reply, and loop in engineering when needed.



In early 2026, we started using Claude

At first, we used Claude for straightforward questions, like whether we supported a particular integration. As MCP became more common, we connected Claude to the systems we normally searched by hand. It could fetch a ticket from Pylon, check Linear to see whether a feature request already existed, or answer product-behavior questions by searching our codebase using Cursor.

We also built an internal MCP server that let Claude securely access our database, logs, and other internal tools.



With that access in place, the team packaged the process into a Claude Skill called `/investigate`, a single command that searched our codebase, issue data, and other systems.

It was a meaningful improvement. We estimated Claude made each person about 20% more efficient. Bringing those individual workflows to the whole team, though, created new problems.

Costs scaled with usage,

and we had little visibility or control over spend. Across a 10-person team, we were on pace to spend about $240,000 a year.

Skills were hard to share.

Sharing the skill file was only one step. To actually use it, each teammate also needed a Cursor seat, access to the internal MCPs it called, and permissions for systems like Salesforce. Getting everyone set up took time, so skills stayed limited to people who already had the right access.

Skills were just as hard to maintain.

The people closest to the work couldn't update them directly. Support teammates needed GitHub seats and had to know how to edit files, commit changes, and push updates. Changes went through a few technical people, so skills fell behind as the product and team changed. If those people left, the knowledge to maintain them often left too.

Support engineers had to jump...

...between Pylon and Claude, manually run a Skill on every ticket, and keep their laptops open while it ran.

Every investigation started with...

...a fresh search across past tickets, account data, and product knowledge. Some took up to 30 minutes and cost as much as $10.

Long story short, Claude was useful for prototyping workflows, but it wasn't production-ready. We needed something purpose-built for support teams that could make these workflows shared, always on, and part of how the team handled every ticket.

So we built Agentic Support

We brought what was working in Claude into Pylon, where it could run in the background instead of needing someone to manually kick it off. A Skill that once lived in one engineer's Claude setup became something people and agents across the team could use and iterate on.

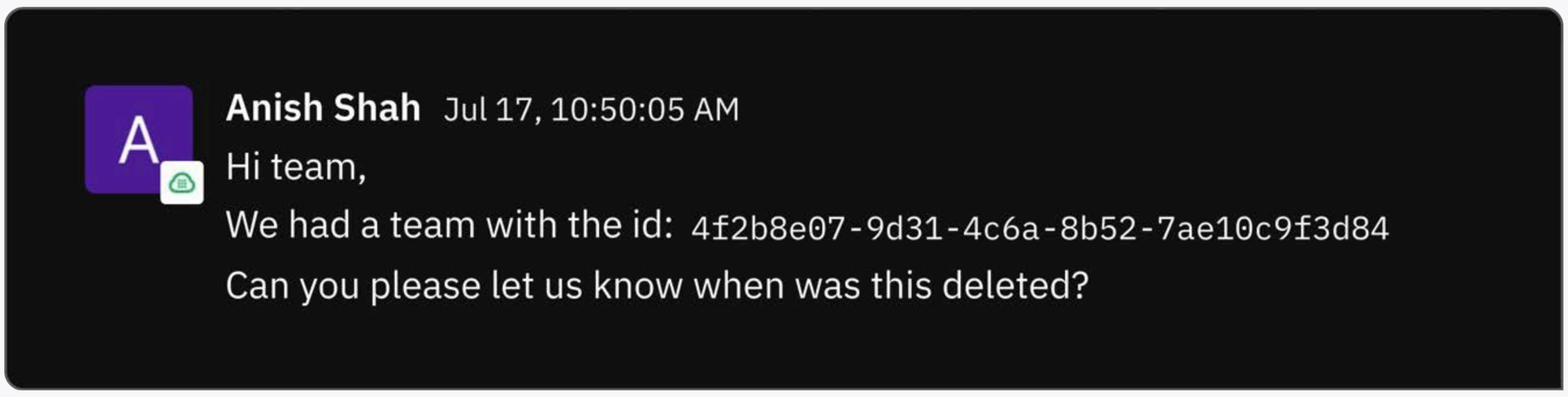
It changed how the team worked

BEFORE	AFTER
A ticket sat untouched until someone opened it	Tickets are pre-investigated by a **Background Agent** before humans get to them
We jumped between Pylon and Claude to continue an investigation	We continue the investigation right in the ticket with **Assist Agent**
Skills lived in the Claude setup of whoever built them	**Skills** live in Pylon, where the whole team can use, manage, improve, and measure them
Complex issues could only be solved by certain people on our support team	Everyone on our support team (and even other teams) can debug complex issues with reusable hosted Pylon **Skills**
Other departments would ask the support team for help in our internal #support-helpdesk Slack channel	Anyone from Sales, Customer Success, Marketing, and more at Pylon can use **Slack Agent** to get answers to product or customer questions and take action without leaving Slack

What happens when a ticket comes in

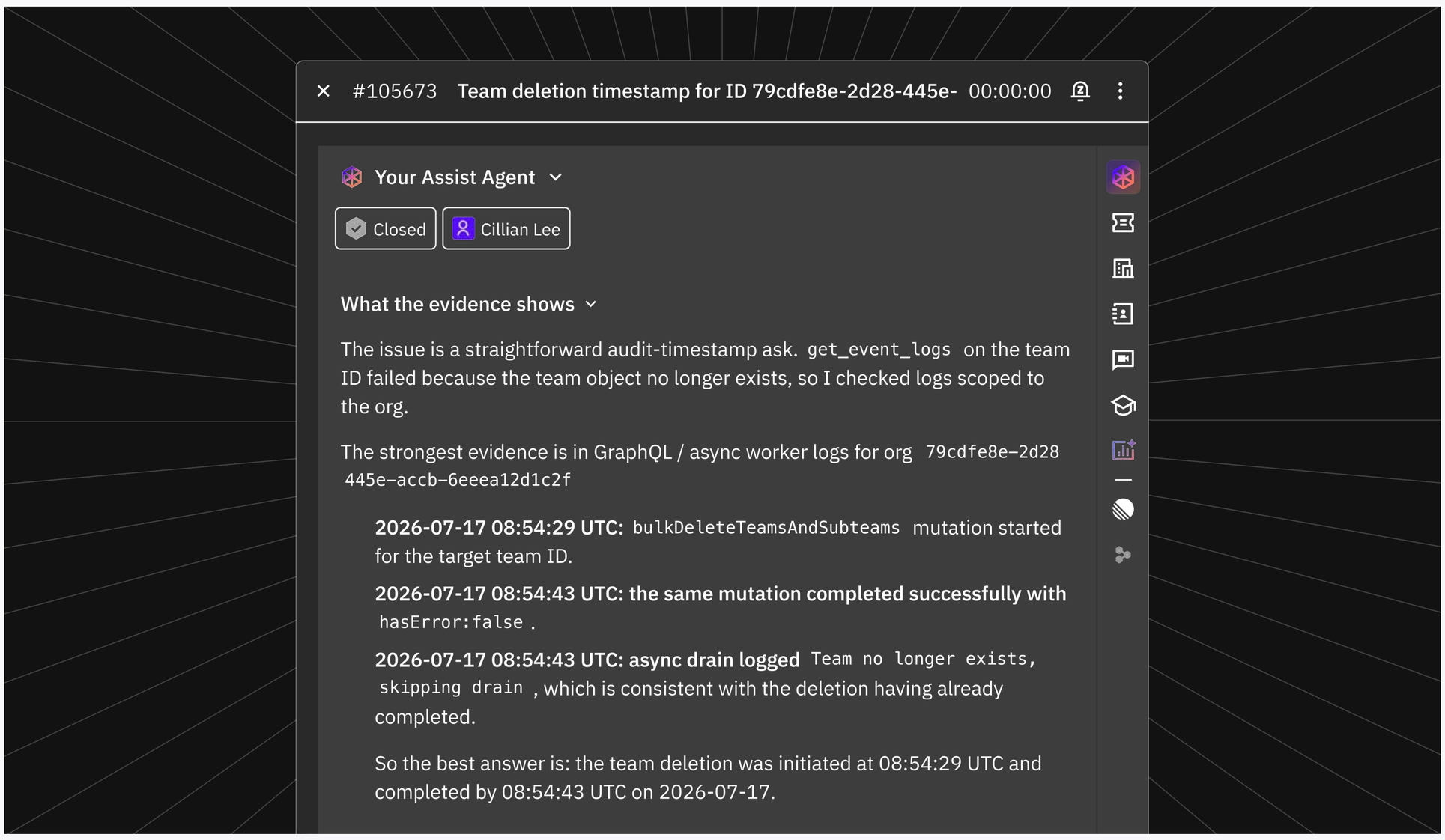
→ It arrives pre-investigated, then we take it from there.

Let's say a customer emails asking when a team with a specific ID was deleted.

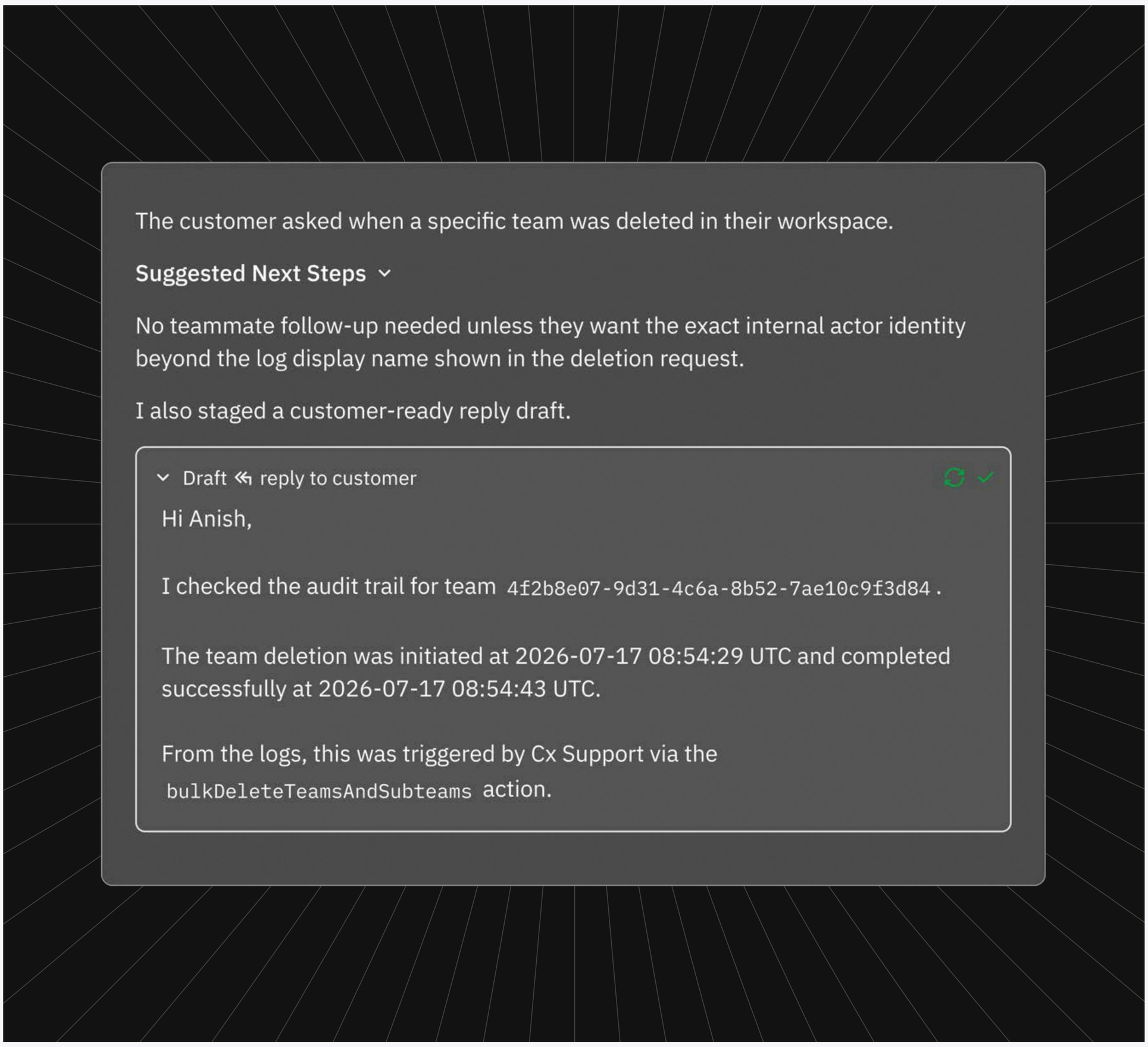


Before, we'd log into AWS, figure out the right log group, write a SQL query to find the logs, confirm what happened, and write the reply.

Now, by the time we open the ticket, a **Background Agent** has already found the ID, run the search, and pulled the relevant log lines and timestamps.



It suggests next steps and drafts a response.



If that's enough, we confirm it and send the reply. What used to be a 30-minute investigation now takes about two minutes to review and approve.

If it isn't, we direct Assist Agent to dig deeper before we respond.

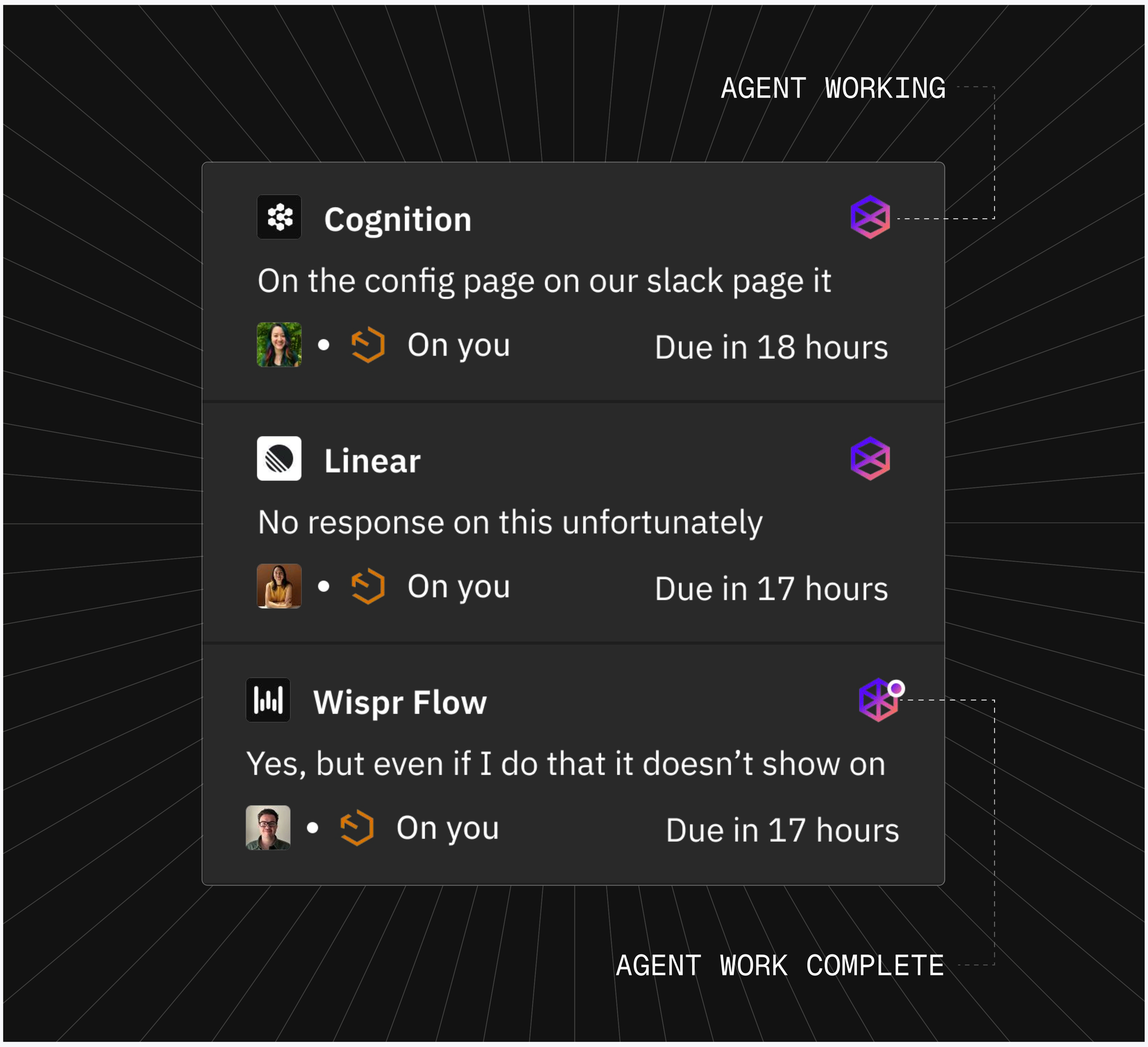
Say the deletion looks unusual and we want to know whether it has happened before. We ask it to investigate:

- "Double-check whether this was a one-off. Look for similar team-deletion issues in the codebase and past tickets, and tell me how they were resolved."

The agent checks the relevant code paths and past tickets for similar team-deletion issues and how those cases were handled, all without us leaving the ticket or opening a new tab.

We can ask follow-up questions, check another source, or take the investigation in a different direction. If it finds a bug, we can ask it to file a complete bug report for an engineer or coding agent in seconds.

While it works, we can move to other tickets and return when it has an answer.



Agents start with usable context, not scattered data

The investigation above works because the agent doesn't start from a blank prompt. Pylon turns our support history, account activity, product data, and internal work into Support, Account, and Product Intelligence.

For each ticket, the agent uses that intelligence to assemble the context that matters, then goes back to the underlying sources when it needs more.

By the time a support engineer opens a ticket, the agent already has:

- **What the customer is trying to do and how they use the product**
- **What happened in previous conversations or calls**
- **Whether we have seen the same issue before and how we handled it**
- **Which bugs, feature requests, or product behavior might be relevant**

When the context layer isn't enough to resolve an issue, the agent uses it as a starting point for a deeper investigation across logs, account data, Linear, Notion, Slack, and the codebase.

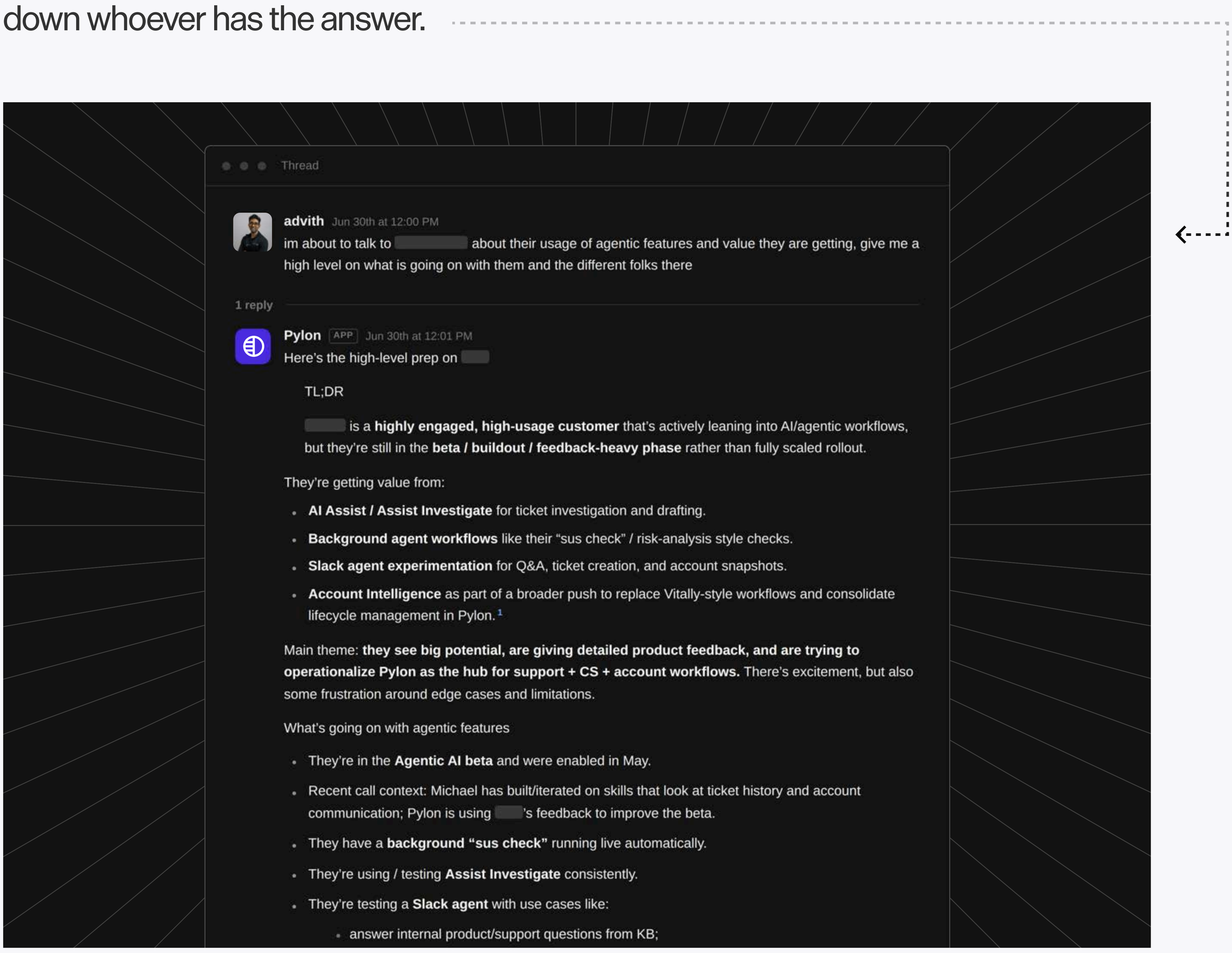
Over time, the system can learn from each investigation. Every time someone uses Pylon agents to solve an issue, it captures the path it took, including where it looked, what it found, and even the dead ends. It's as if the agent wrote down its approach for the next agent. That history helps future investigations start with a better understanding of how similar issues were resolved and can surface opportunities to improve the support process, such as updating a knowledge-base article or turning a repeatable workflow into a Skill.

The rest of the company gets the same context in Slack

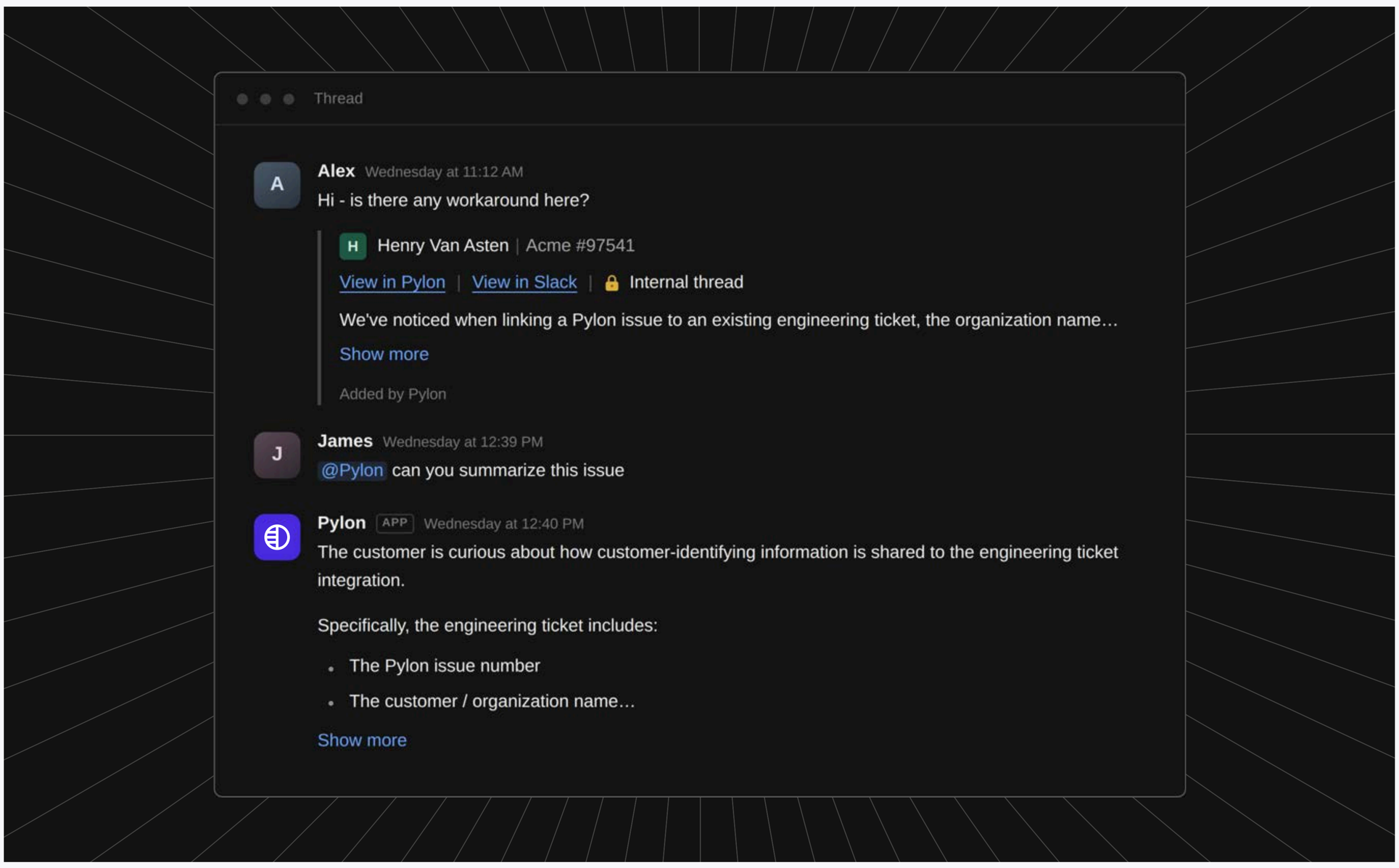
So far, that context has lived in Pylon, where support could use it directly. But the rest of the company works in Slack, so we brought it there too.

Now, more than a third of the company uses **Slack Agent** every week, including Sales, Customer Success, Solutions, Product, Marketing, and leadership.

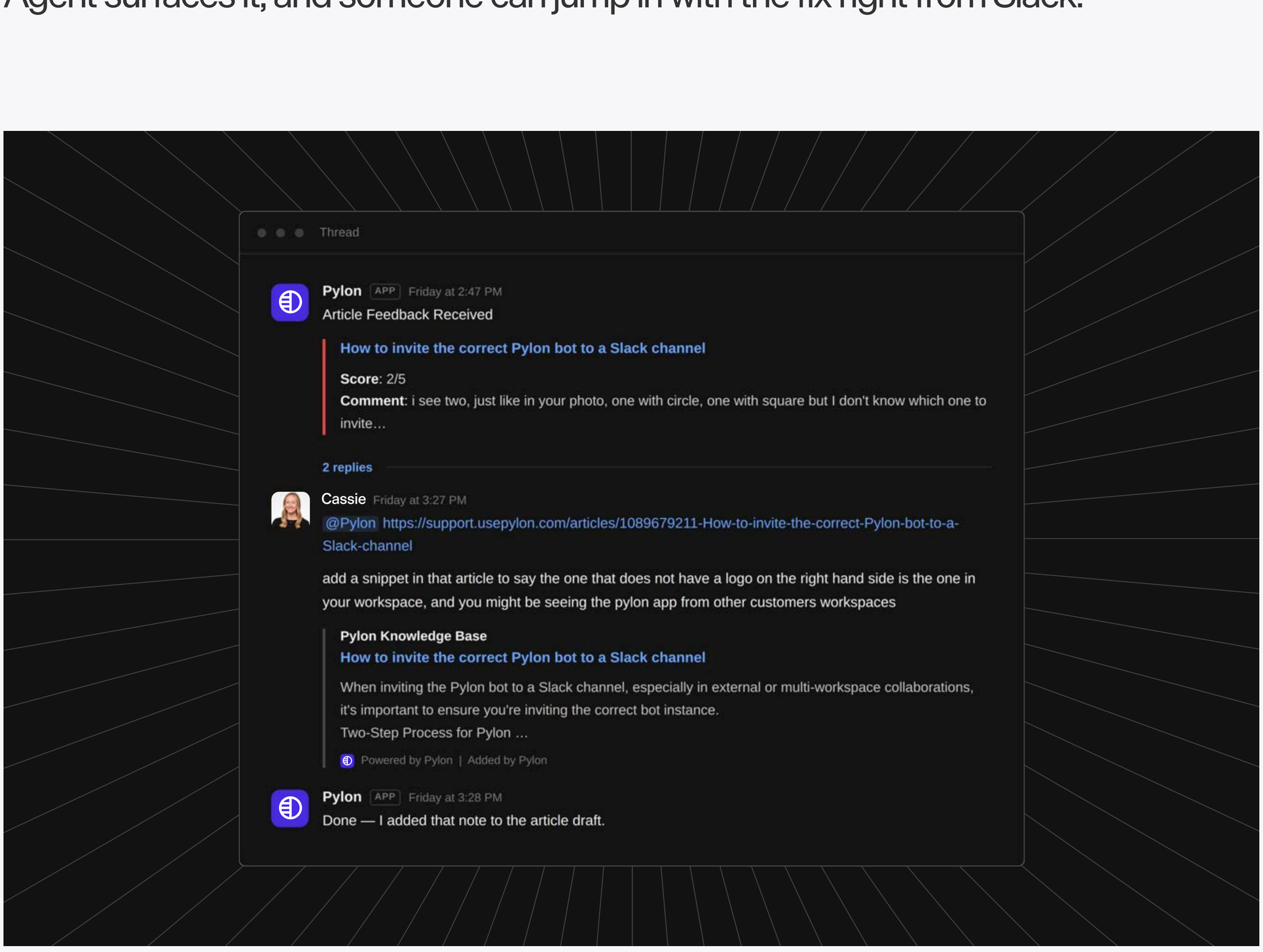
Someone might ask about a specific account or an entire book of business and get the full history back, instead of piecing it together from different tools or tracking down whoever has the answer.



They might also use Slack Agent to summarize a live customer issue or surface a workaround, with the relevant account and engineering context pulled together in one place.



Or they can put it to work. When a customer leaves feedback on a help article, Slack Agent surfaces it, and someone can jump in with the fix right from Slack.

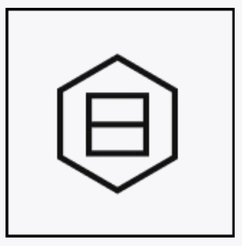


The impact went beyond support

What started as a better way to handle tickets changed how the whole company worked.

Support

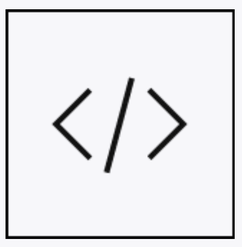
The team moves through harder issues faster, even as ticket volume grew. Median first-response time fell 72%, from 19.4 to 5.3 minutes, and 62% of tickets involve substantive agent work, whether an accepted draft or an active back-and-forth during investigation.



Investigation processes that once lived in individual setups are now reusable as **Skills** the whole team can build on, which means less time tracking down the same context and more time on the decisions that actually need a person.

Engineering

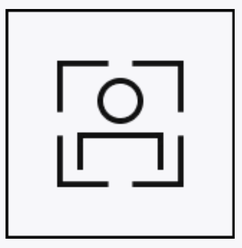
Support can now answer more product questions without involving Engineering. Through the Cursor integration, the team can ask the codebase how the product works, then use that context to resolve or scope issues independently.



In the four weeks after rollout, Support ran 773 codebase searches, and questions in our internal #product-helpdesk Slack channel dropped by about 50% compared with the four weeks before rollout.

The rest of Pylon

Across Pylon, team members no longer have to hunt down customer or product information or go searching for someone on another team who might have it. Slack Agent helps them pull together what they need and act on it directly in Slack.



More than a third of Pylon uses Slack Agent weekly, including Sales, Customer Success, Solutions, and leadership, getting product and customer questions answered on their own rather than tracking down whoever might know.

Here’s what changed in the numbers

Response times

The typical customer heard back more than 14 minutes sooner. More tickets now get a response within the team's one-hour target, and SLA attainment climbed to 73.3%, even as ticket volume grew.

METRIC	BEFORE → AFTER		CHANGE
Median first response	19.4 min	5.3 min	72% shorter
SLA attainment rate	63.1%	73.3%	+10.2 pts
Responses within 1 business hour	50.5%	62.9%	+12.4 pts

Capacity

During the measured period, ticket volume grew 3% while support headcount decreased 10%. Even so, response times improved across every measure we tracked.

Before Agentic Support, we planned to grow the support team to 18 people by year-end. After adopting it, we revised that plan to 11 while keeping the same growth forecast.

This changed our hiring plan, not the size of the existing team.

“Everyone’s technical floor has risen. Issues that once required two or three escalation steps can now be handled by nearly everyone on the team.”

Jon Clark, Support Engineering Manager

FAQ

Couldn't we just build this with Claude?

You can build pieces of it. The hard part was turning individual Claude workflows into something the whole support team could count on. That meant shared Skills, controls for the team, agents that run in the background, and a system that does not depend on everyone maintaining their own setup.

What happens when an agent gets something wrong?

We review the work before anything reaches a customer or changes something important. Then we decide whether to use the result, edit it, or point the investigation in a different direction.

What can it actually see and do?

It can access the systems we connect, including ticket history (Pylon), account context (Salesforce, call recordings, emails with customers), engineering work (Linear), product specs (Notion), logs (AWS), internal conversations (Slack), codebase (Cursor), Knowledge Base (Pylon), and our internal product data (internal MCP). It can investigate an issue, recommend next steps, and take actions we approve. We stay in control of what happens next.

How long did it take to see results?

Within about a month, everyone on the support team was using Agentic Support every day. We measured response-time results by comparing the four weeks before rollout with the four weeks after.

Are support engineers being replaced?

No. The actual team had ten people during the rollout and has nine today. The 18-to-11 change refers only to our year-end headcount forecast.

Methodology

Results compare the four weeks before rollout, May 4–31, 2026, with the four weeks after rollout, June 8–July 5, 2026. Ticket volume grew 3% across the comparison period.

Move from doing the work to directing it.

See what Agentic Support could look like for your team.

Pylon