AI AGENTS AT WORK

From Code to System

How AI agent work in a company improves in three stages, and what a leader can measure at each one.

STAGE 1
Code

→ STAGE 2
Factory

→ STAGE 3
System

SUMMARY

Summary

Most engineering teams now use AI coding agents, and many use several at once, but the leaders who pay for them can rarely say what the agents cost, where the money is wasted or what risks they create. This paper describes three stages that solve those problems in order. In the first stage, Code, each engineer and their team lead can see the cost, waste and security risks of the agents they already use. In the second stage, Factory, a team's work moves from a ticket to a merged change along a fixed path, where the team's own tests check each change and an engineer reviews and merges it. In the third stage, System, every team in the company works with AI agents under the same written workflows, checks, sign-off rules and record.

Each stage answers the question that the stage before it leaves open, and each gives a leader numbers to track. The paper sets out the evidence for each stage and describes Widal's product for each one: Widal.Code, Widal.Factory and Widal.AS (AgentSystem). It ends with a short list of first steps.

THE PROBLEM

Agent use has grown faster than the means to manage it

An AI coding agent is a program that reads code, runs commands and edits files for an engineer, such as Claude Code from Anthropic or Codex from OpenAI. Their use at work rose quickly over the last year. In Stack Overflow's surveys, the share of developers who use agents at work went from 31% in 2025 to 59% in April 2026, and most of the growth was in daily use, which 40% of the developers who answered reported[1]. Anthropic reports that across its enterprise deployments, Claude Code costs on average about $13 per developer per active day, or $150 to $250 per developer per month[2].

Teams also use more than one tool. Sonar found in January 2026 that software teams use four different AI tools on average[3], and in a survey by The Pragmatic Engineer in early 2026, 70% of engineers used two to four tools and 15% used five or more[4]. Each agent vendor offers reports and spending limits for its own agent but not for anyone else's, and several of those reports need an enterprise plan. For Team and Enterprise seats, Anthropic's own documentation says that “usage inside the seat allowance isn't metered in dollars”[2]. A leader who wants one view of cost across the tools a team uses has to build it.

Leaders report the effects in three areas: cost, value and risk.

WHAT LEADERS REPORT, 2025 TO 2026

42%

of engineering leaders name rising AI tool costs as a primary challenge

Jellyfish, June 2026[5]

43%

of technology and security leaders spend more on AI than they budgeted

Retool, August 2026, 101 leaders[6]

86%

of engineering budget holders are unsure which AI tools give the most benefit

DX, October 2025[7]

81%

of security teams lack full visibility into how AI is used in development

Cycode, November 2025, 400+ security staff and leaders[8]

Cost

Jellyfish's usage data shows why budgets run over: the typical user at the median company spent about $5 a month on AI coding tools in August 2025 and $81 in May 2026, sixteen times more[9]. Gartner expects the cost of AI coding to pass the average developer's salary by 2028[10].

Value

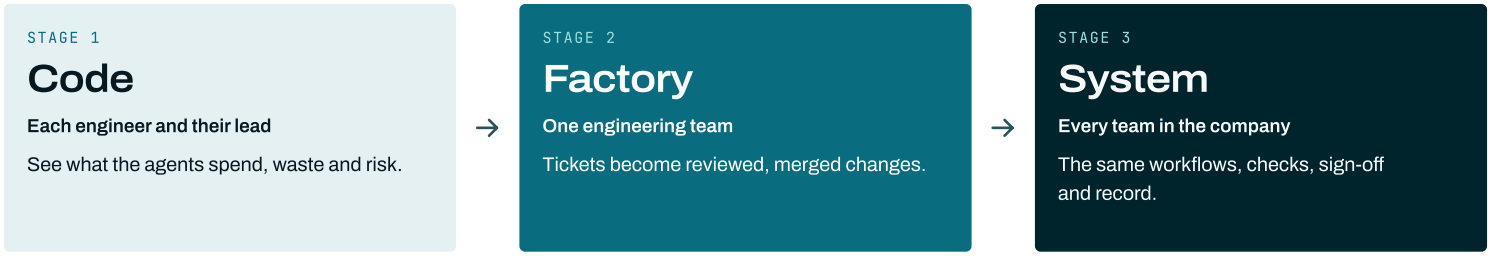
In the same DX survey, 40% of budget holders said they lacked the data on adoption and impact to make a case for the return on their spend[7], and Jellyfish found that only 46% of organizations track AI measures such as adoption and model use[5]. Impressions are a weak substitute for those measures. In a randomized trial by METR in 2025, experienced developers took 19% longer on tasks when they used AI tools, while they believed the tools had made them 20% faster[11]. METR's follow-up in February 2026 said a speed-up is now likely, but that its data is only very weak evidence of how large it is[12].

Risk

In the Cycode survey, 52% of security teams also had no formal framework for governing AI[8], and Sonar found that 35% of developers reach their AI tools through personal accounts that the company does not manage[3].

Our reading of these numbers is that the problems sit at three levels. For the engineer, the question is what the agents cost and where the tokens go (a token is the unit of text that AI models read and write, and the unit in which they are billed). For the team, the question is how agent work gets from a ticket to a change that a person trusts. For the company, the question is how every team can use agents under the same rules, so that security and compliance review one setup instead of many. The three stages below take these questions in that order.

FIGURE 1. THE THREE STAGES



Each stage answers the question the stage before it leaves open, with more control around the agents at each step.

STAGE 1 Code

See what the agents spend, waste and risk

What goes wrong without it

An engineer on a subscription plan sees how much of the plan's limit is left but not what the work would cost in dollars, and each agent reports only its own use. Claude Code now shows each engineer a breakdown of their own recent usage with suggested changes[2], but in our research we found no agent vendor that shows waste across a team or across other vendors' agents.

Most of the tokens go to context that the agent sends again and again. An agent sends its whole conversation with each request to the model, including every file it read and every command output it received. Anthropic, for example, keeps that context in a prompt cache (a short-term store that makes repeated input much cheaper) for 5 minutes or 1 hour. When the cache expires, for example during a break longer than its lifetime, the next request sends the whole context again at the full price, which in a long session can be several hundred thousand tokens[13]. DX's data from more than 500 organizations shows that 94% of the tokens coding agents use are cache reads, the re-reading of context that already exists[14]. Waste in this setting means tokens that bought nothing: an expired cache, a large command output carried through every later request, or a more expensive model than the task needs.

Risk sits in the same place. An agent's settings decide what it may do without asking, such as running any command, reading files that hold keys, or pushing code. Those settings live on each engineer's computer, and the agent's actions are written to local logs that nobody reviews, which is why most security teams say they cannot see how AI is used in development[8].

Regulated companies have one more constraint. Anthropic's business associate agreement (BAA, the contract that HIPAA requires before a vendor may handle protected health information) covers Claude Code only in its local mode, connected directly to Anthropic, and does not extend to plugins, MCP servers or OpenTelemetry export[15]. MCP, the Model Context Protocol, connects an agent to outside tools and data, and OpenTelemetry is an open standard for sending usage data to a monitoring system. A cost tool that copies session text to its own cloud therefore adds one more place where patient data could end up, outside that agreement.

What changes with it

Widal.Code is a free, open source tool (MIT licence) that runs on each engineer's computer. So far it runs on macOS. It reads the session logs of the coding agents the engineer already uses, which today are Claude Code, Codex and Cowork (the agent mode of Anthropic's desktop app), and reports three things.

Spend against the plan

For each account, Widal.Code shows the dollars at list prices by day, project, model and ticket. For an account that pays by the token, that figure is the spend. For a subscription, it is the API value, which is what the same use would cost if it were paid by the token, shown next to the plan's price for the same days. An engineer and their lead can then see whether a plan is too small, too large or about right.

Waste, with the fix

Widal.Code finds expired caches, large tool results carried through later requests, repeated tool calls, and simple work done on an expensive model or at a high effort level (a setting that makes the model think longer). Each finding gives the amount, the cause and the setting that fixes it. For most fixes, Widal.Code replays the engineer's own logs with the change applied to estimate what it would have saved. Each change is applied with one command and undone with another.

Security risks

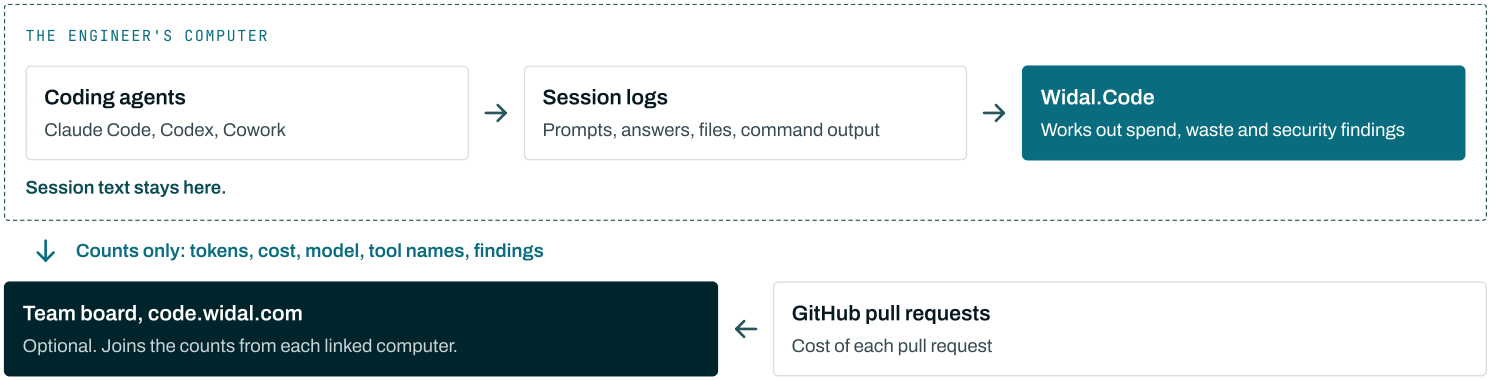
Widal.Code checks the agents' settings, what the agents did and the code they changed. Examples are a mode that skips permission prompts, a read of a file that holds keys, a force push (one that overwrites the shared history of a branch), a secret in the code, or code that turns off the check of a server's security certificate. Each finding names the controls it touches in the frameworks the team turns on, such as HIPAA, SOC 2, HITRUST, 21 CFR Part 11 or the OWASP Top 10 for agentic applications (a list of the main security risks of AI agents). In Claude Code, Widal.Code asks the engineer to confirm before a push while a high risk is open in that repository. Widal.Code checks technical controls and flags risks. It does not certify that a company is compliant.

Session text never leaves the engineer's computer. Widal.Code takes counts, sizes, names and flags from the logs (tokens, cost, model, tool names, the folder and the git branch), and it never keeps the text of a prompt, an answer, a tool result or a file. A security finding records which rule matched, where and when, and never the text that matched, so a finding cannot carry a key or a patient's data. Your security team still makes its own assessment, but it reviews a tool that sends no session text anywhere.

A team can link its computers to the optional team board at code.widal.com. The board receives the same counts from each linked computer and joins them with the team's pull requests on GitHub (a pull request is a proposed change to the code that a person reviews before it is merged). It shows the team's spend, the cost of each pull request, waste and security findings in one place.

The reports cover all three agents. The fixes, the confirmation before a push and the hints while an engineer works act inside Claude Code.

FIGURE 2. WHAT STAYS ON THE COMPUTER



What a leader can measure

MEASURE	WHAT IT TELLS YOU
API value against plan price, per engineer and month	Whether each plan fits the work
Share of spend on expired caches and carried context	How much of the spend bought nothing
Spend per ticket and per merged pull request	What a unit of delivered work costs in agent use
Open high-severity findings, and days until each is fixed	Whether agent settings and actions are getting safer

STAGE 2 **Factory**

Move team work from ticket to merged change

What goes wrong without it

Seeing cost and risk does not change how work moves through a team. When each engineer drives agents by hand on a laptop, the agents produce changes faster than the team can review them, and nobody keeps a shared record of what each agent did. DX's data for the third quarter of 2026 shows that in organizations where 80% or more of the code is written by AI, the median wait for a first review is 82 minutes, and across the same data the median pull request is 122% larger than a year earlier[14]. Developers also say they do not fully trust what the agents write. In Sonar's survey of more than 1,100 developers, 96% did not fully trust AI-written code, only 48% always checked it before they committed it, and 38% said it takes more effort to review than code written by colleagues[16].

Where companies have published how they run coding agents at scale, the agents work inside a fixed path. Anthropic's guide to building agents separates workflows, “systems where LLMs and tools are orchestrated through predefined code paths”, from agents that direct their own process, and it recommends workflows for well-defined tasks because they “offer predictability and consistency”[17]. (LLMs are large language models, the AI models that agents run on.) Stripe wrote in February 2026 that over 1,300 pull requests written entirely by its internal coding agents are merged each week, each one reviewed by a person, and that a change which still fails its automated tests after two rounds goes back to the engineer who started it[18].

What changes with it

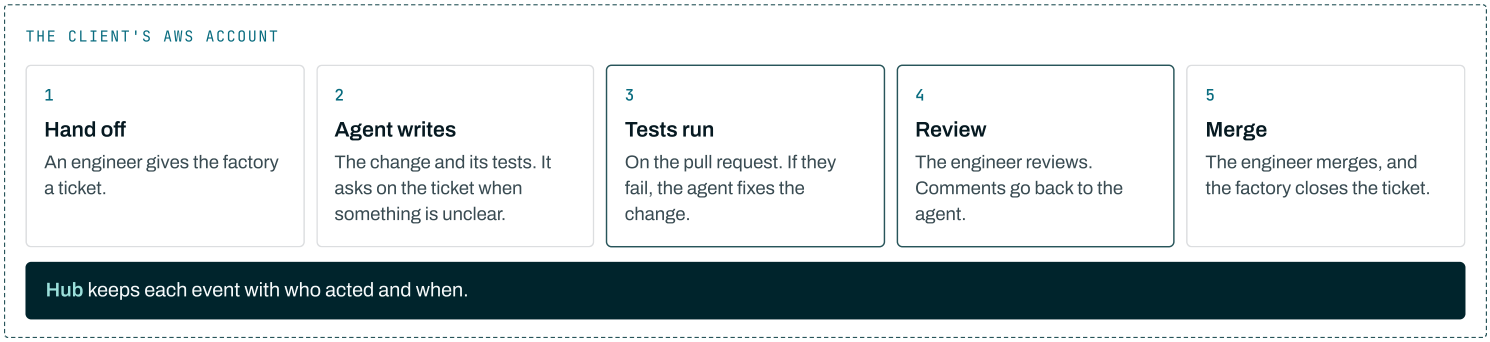
Widal.Factory is the part of Widal's system for engineering teams. An engineer hands a ticket (a work item in a tracker such as Linear, GitHub Issues or Jira) to the factory. A coding agent takes the ticket, reads the code, writes the change and its tests, and asks on the ticket when something is unclear. It opens a pull request, and the team's own automated tests run on it as on any other change. The engineer reviews the pull request, and review comments go back to the same agent, which fixes the change and pushes it again. When the engineer merges, the factory closes the ticket and cleans up.

Each engineer has a factory of their own, and every factory reports to one hub, the team's web page for agent work. The hub shows what needs a person, why each ticket waits, who acts next, and whether the factory is getting faster and better. It keeps an append-only log of events (a record that is only ever added to), so each run, question, check result, review and merge is kept with who acted and when. For each ticket, the hub shows where its time went, stage by stage, and its model cost.

Widal.Factory runs in the client's own AWS account. The agents reach the AI model through the client's own account with the model provider, under the client's own agreements such as a BAA. Widal's engineers deploy, update and monitor the factory there, through access roles that the client creates and can remove at any time.

For a CTO, the factory turns agent work into units that can be counted: each ticket has a start, a merged change and a cost. For a head of security or compliance, agent work now happens in one place the company controls, under the team's existing tests and review rules, with a record of who approved each change.

FIGURE 3. HOW A TICKET MOVES THROUGH THE FACTORY



What a leader can measure

MEASURE	WHAT IT TELLS YOU
Time from ticket to merged change	How fast work moves, including the time it waits for people
Share of changes sent back by review	How often the agent's first attempt is good enough
Share of changes that fail after merge or are reopened	Whether speed costs quality
Model cost per merged change, at list price	The unit cost of delivered work
Time items wait on a person, and on whom	Where people are the bottleneck

STAGE 3 System

The same rules for every team

What goes wrong without it

AI work does not stay in engineering. In a Paubox survey of 151 healthcare leaders whose organizations use AI coding assistants, carried out in July 2026, IT staff build tools with them at 84% of organizations and developers at 58%, and in about two in five organizations no developer is involved at all[19]. 79% of those organizations use MCP servers to connect the tools they build to other systems[19]. When teams outside engineering, such as quality, regulatory or patient access, adopt agents one tool at a time, each setup needs its own security review, its own data agreements and its own way of showing an auditor what happened.

That is where company programs stall. In a Madrona survey of 150 leaders at companies with $500 million or more in revenue, 83% had moved fewer than half of their AI pilots into production in the past year, and the two most common reasons pilots failed were integration complexity (55%) and security, privacy or compliance (52%)[20]. In healthcare, Bessemer, AWS and Bain found in 2025 that 30% of completed proofs of concept reach production[21].

Regulated work also has written rules for records. The HIPAA Security Rule requires “hardware, software, and/or procedural mechanisms that record and examine activity in information systems” that contain or use electronic protected health information[22]. The FDA's rule for electronic records, 21 CFR Part 11, requires “secure, computer-generated, time-stamped audit trails” of the entries and actions that create, change or delete those records[23]. A company that lets agents draft or change such records has to show the same trail for the agent's work as for a person's. Your quality and compliance teams decide how these rules apply to you.

What changes with it

Widal.AS is one agent system for the whole company. Each team gets AI agents, written workflows, fixed checks and sign-off by a named person, and everything runs in the client's cloud with one hub. Widal.Factory is its part for engineering teams, and other teams join the same hub one at a time. Widal's engineers set up each team's workflows with the client and run the system with them.

Each team's work runs the same way:

1. A person hands off the work, such as a ticket, a deviation report or a prior authorization request.
2. An agent does the work. It follows the team's written workflow inside the client's cloud account, and it asks a person when something is unclear.
3. Fixed checks test the result. A check is a small script that gives the same answer every time for the same input, such as “every required field is filled” or “every reference points to an approved document”. If a check fails, the agent tries again up to a set limit, and then a person takes over.
4. A named person signs off at the points the team's procedures require. The record keeps each step, check result and approval, with who did it and when.

TEAM	WORK AN AGENT DRAFTS	A CHECK THE WORK MUST PASS
Engineering	A pull request with tests, from a ticket	The team's tests pass
Quality	A deviation or CAPA (corrective and preventive action) record, from the evidence	Every required field is filled, and every reference points to an approved document
Regulatory affairs	A submission section, with citations	Every statement cites a controlled source
Clinical operations	A check of the trial master file for missing documents	The required documents are present and their versions are current
Patient access	A prior authorization packet	The payer's required fields and deadlines are met

In each case a named person on the team approves before the work is final.

For the company, the change is that security, legal and compliance review one setup with one set of agreements, instead of a new tool for each team, and auditors can follow a piece of work from the request to the sign-off in one record.

What a leader can measure

MEASURE	WHAT IT TELLS YOU
Share of AI work that runs inside an approved workflow	How much unmanaged use is left
Share of results that pass the checks on the first try, per workflow	Where a workflow or the agent's instructions need work
Time from hand-off to sign-off	How fast the whole process runs, with the person included
Audit and review findings about AI work	Whether the record answers the auditor's questions

HOW THEY CONNECT

How the stages build on each other

Each stage depends on what the stage before it provides. A team cannot set a budget for a factory without knowing what its agents cost per ticket, and Code gives that number. A company cannot extend sign-off and records to quality or regulatory work without a working pattern of workflow, checks, person and record, and engineering is a natural first place for that pattern, because the tests and the review step already exist there. The order also follows the level of risk. The first stage only reads what agents did, the second lets agents propose changes that a person merges, and the third lets agents draft records in regulated processes, each with more control around it.

	Code	Factory	System
Who it serves	Each engineer and their lead	One engineering team	Every team in the company
The question it answers	What do our agents cost, waste and risk?	How does agent work get from ticket to merged change?	How does every team use agents under the same rules?
Where it runs	On each engineer's computer, with an optional board at code.widal.com	In the client's AWS account	In the client's cloud, with one hub
What a person does	Reads the findings and decides which fixes to apply	Reviews and merges each change	Signs off where the procedures require it
What it records	Counts: tokens, cost, models and findings	Each run, question, review and merge	Each step, check result and approval, with who and when

A company does not have to start with the first stage, and a team that already knows its agent costs can begin with a factory. Each later stage still needs the information the earlier ones produce, so a company that skips a stage has to collect that information another way.

FIRST STEPS

What to do first

01 Measure one month of agent use on a few engineers' computers, across the agents they already use, before you change anything. Widal.Code is free, and its optional team board joins the numbers from several engineers.

02 Fix the largest waste items and every high-severity settings finding, then check after two weeks whether spend per ticket has changed.

03 Write down the team's rules for agent work: which tickets agents may take, which tests must pass and who may merge.

04 Choose one team and one repository for a factory in your own cloud account, and track the time to merge, the share of changes sent back by review and the cost per merged change for a month.

05 Bring security and compliance in at the start, and agree which data agents may see, which agreements with the model provider apply and where the record is kept.

06 Choose the next team where the procedures and sign-off points are already written down, because those become its workflow and its checks.

ABOUT

About Widal

Widal, Inc. builds AI agent systems for companies in regulated industries, including life sciences, healthcare and health plans. Its product is Widal.AS, one agent system for the whole company, with Widal.Factory as its part for engineering teams. Widal's engineers install and run the system with each client, in the client's own cloud account. Widal.Code is Widal's free and open source tool for engineers who use AI coding agents. It was called Kestrel until 17 November 2026, and its code is at github.com/Widal-Inc/kestrel.

Contact: hello@widal.com.

METHOD

How this paper was compiled

The market figures come from the surveys, reports and documents listed below, published between December 2024 and October 2026 and read on or before 17 November 2026. Most of the surveys were run by companies that sell into this market, so their figures show a direction rather than exact values. The product descriptions come from Widal's product documentation as of 17 November 2026.

Sources

1. Stack Overflow, "Agents on a leash: agentic AI remains mostly monitored at work", 27 May 2026 (pulse survey, April 2026, about 1,100 responses), compared with the Stack Overflow Developer Survey 2025, AI section. stackoverflow.blog/2026/05/27/agents-on-a-leash-agentic-ai-remains-mostly-monitored-at-work and survey.stackoverflow.co/2025/ai
2. Anthropic, "Manage costs effectively", Claude Code documentation, read 17 Nov 2026. code.claude.com/docs/en/costs
3. Sonar, "Shadow AI is already writing your code", 29 Jan 2026. www.sonarsource.com/blog/shadow-ai-is-already-writing-your-code/
4. The Pragmatic Engineer, "AI Tooling for Software Engineers in 2026", 3 Mar 2026 (survey of 27 Jan to 17 Feb 2026, 906 responses). newsletter.pragmaticengineer.com/p/ai-tooling-2026
5. Jellyfish, "In 2026 engineering leaders are shifting from AI adoption to AI accountability", 11 Jun 2026. jellyfish.co/blog/2026-engineering-leaders-shifting-from-ai-adoption-to-ai-accountability/
6. Retool, "How tech leaders are handling AI cost management", 20 Aug 2026 (101 technology and security leaders). retool.com/blog/how-tech-leaders-are-handling-ai-cost-management
7. DX, "How are engineering leaders approaching 2026 AI tooling budgets?", 15 Oct 2025. newsletter.getdx.com/p/how-much-should-you-spend-on-ai-tools-in-engineering
8. Cycode survey of more than 400 CISOs and security staff, as reported by Agile Brand Guide, 5 Nov 2025. agilebrandguide.com/report-shadow-ai-crisis-looms-as-100-of-companies-have-ai-generated-code-but-81-of-security-teams-lack-visibility/
9. Jellyfish, "The State of AI in Software Engineering", research review H1 2026, data through May 2026. jellyfish.co/state-of-ai-software-engineering/
10. Gartner, "Gartner Predicts AI Coding Costs Will Surpass Average Developer Salary by 2028 as Token Consumption Surges", press release, 24 Jun 2026. gartner.com/en/newsroom/press-releases/2026-06-24-gartner-predicts-ai-coding-costs-will-surpass-average-developer-salary-by-2028-as-token-consumption-surges
11. METR, "Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity", 12 Jul 2025 (16 developers, 246 tasks). arxiv.org/abs/2507.09089
12. METR, update on developer speed-up, 24 Feb 2026. metr.org/blog/2026-02-24-uplift-update/
13. Widal.Code documentation, version 0.2.0, 16 Nov 2026. github.com/Widal-Inc/kestrel
14. DX, "New DX data finds 94% of AI token usage is re-reading existing context", Q3 2026 report, 14 Oct 2026 (more than 500 organizations). getdx.com/news/new-dx-data-finds-94-of-ai-token-usage-is-re-reading-existing-context/
15. Anthropic, "Business Associate Agreements for commercial customers", read Nov 2026. privacy.claude.com/en/articles/8114513
16. Sonar, State of Code Developer Survey report, 8 Jan 2026 (more than 1,100 developers). www.sonarsource.com/blog/state-of-code-developer-survey-report-the-current-reality-of-ai-coding/
17. Anthropic, "Building effective agents", 19 Dec 2024. www.anthropic.com/engineering/building-effective-agents
18. Stripe, "Minions: Stripe's one-shot, end-to-end coding agents, part 2", 19 Feb 2026. stripe.dev/blog/minions-stripes-one-shot-end-to-end-coding-agents-part-2
19. Paubox, 2026 benchmark for building with AI in healthcare, August 2026 (survey of July 2026, 151 healthcare leaders, margin of error 8%). www.paubox.com/research/2026-benchmark-building-with-ai-in-healthcare
20. Madrona, "Harnessing Enterprise Value: ROI of AI", August 2026 (survey of May 2026, 150 leaders at companies with $500 million or more in revenue). www.madrona.com/wp-content/uploads/2026/08/Madrona-Research-Harnessing-Enterprise-Value-ROI-of-AI.pdf
21. Bessemer Venture Partners, AWS and Bain, "The Healthcare AI Adoption Index", April 2025 (more than 400 payer, provider and pharma leaders). www.bvp.com/atlas/the-healthcare-ai-adoption-index
22. HIPAA Security Rule, 45 CFR 164.312(b), audit controls. www.law.cornell.edu/cfr/text/45/164.312
23. FDA, 21 CFR 11.10(e), audit trails for electronic records. www.law.cornell.edu/cfr/text/21/11.10