

Promptless

RESEARCH READOUT

DoGBench: Can agents meet expert standards for user-facing documentation?

A benchmark of 292 real repository events, scored against the criteria documentation maintainers actually apply in review.

10 systems evaluated: 7 local lanes, 3 cloud agents

dogbench.ai

Built by expert technical writers from widely used open source projects



Real events

Items are pull requests or reported documentation gaps from open-source projects. Rubrics reward actual content instead of similarity to the docs that humans merged.

Validated with maintainers

Maintainers reviewed 42 items. The rubrics recover 89.2% of the criteria maintainers wrote themselves, and the scoring model agrees with human judgments on 93.9% of criteria.

What the benchmark asks an agent to do

Step 1 – Decide

Does this change need docs?

The agent sees a pull request or a user-reported gap in a repository, and must return either a patch or an explicit no-op.

Step 2 – Write

Is the patch good?

Scored 0–100 on accuracy, completeness, placement and reader guidance. One failed P0 requirement caps the score at 60.

292 real repository events

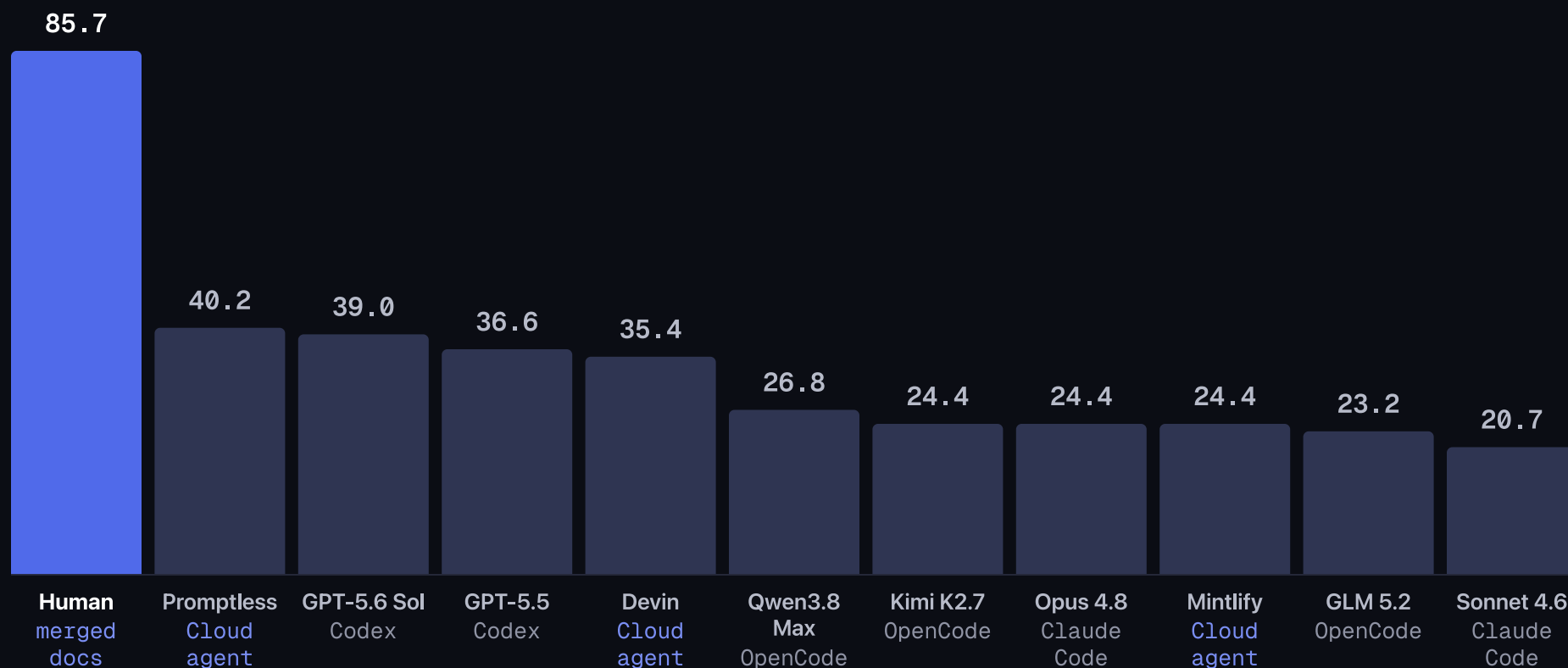
205 / 87 need a patch / need no change

The best agents barely cleared 50 out of 100

#	SYSTEM	COMPOSITE	DELIVERED QUALITY	PATCH QUALITY	ABSTENTION RECALL
1	 Promptless Cloud agent	54.8	48.5	52.3	62.9%
2	 Devin Cloud agent	53.0	39.6	45.1	80.0%
3	Qwen3.8 Max OpenCode	47.3	34.1	42.3	77.1%
4	GPT-5.6 Sol Codex	46.2	44.1	46.3	48.6%
5	GLM 5.2 OpenCode	44.2	30.1	35.8	82.9%
6	Kimi K2.7 Code OpenCode	43.9	34.6	37.8	60.0%
7	Claude Opus 4.8 Claude Code	41.2	27.1	34.2	85.7%
8	Claude Sonnet 4.6 Claude Code	40.0	30.8	32.8	57.1%
9	 Mintlify Cloud agent	36.7	29.5	31.9	48.6%
10	GPT-5.5 Codex	34.0	42.0	43.6	28.6%

Composite combines two things: **Delivered quality**, the average rubric score on the 82 items that needed a patch (a missed patch scores zero), and **Abstention recall**, how often the agent left docs alone on the 35 items that needed no change. An agent has to do well on both. [Cloud agents](#) were told not to use the internet. We verified that Promptless did not browse the internet to find contaminating information, but we could not verify this for other cloud agents.

Humans are better at judging what matters and making sure we get them right



Agents don't know when *not* to write



Abstention recall (%): of the 35 items where no documentation change was needed, how often the agent correctly wrote nothing.

Why does it matter?

Internal detail leaks out

Implementation details users can't act on, published as product.

Reviewer burden

Each one still has to be read and rejected.

Maintenance burden

If merged, it must be versioned, translated and kept true.

Wrong precedent

The next agent run reads it as a pattern to follow.

What is actually wrong with the patches

Task-completion gap	Omits a decision point, procedure, verification step, or recovery path the reader needs.	45.5%
Technical inaccuracy	Misstates an interface or behavior, including its scope, default, lifecycle, or compatibility.	36.6%
Incomplete conceptual or reference coverage	Omits the core concept, capability, or contract the reader needs to understand and use the change.	32.5%
Missing supporting information	Covers the main task but omits rationale, boundaries, examples, or secondary cases.	26.5%
Missing prerequisites	Omits permissions, dependencies, credentials, versions, or other setup conditions.	21.0%
Hard to find	Puts content on a page readers won't reach, or never links to it from where they are.	20.7%
Missing audience and purpose framing	Does not establish who the content is for, why it matters, or when to use it.	19.9%
Out-of-date related pages	Updates one page but leaves other pages that describe the same thing out of date.	13.3%

Share of 1,267 audited submissions. Labels selected for at least 10% of submissions; multiple modes may apply to one patch.

The dominant failure mode: accurate text that still leaves the reader stuck

Agents treat a change as one piece of information to convey, rather than one step in a journey that needs prerequisites, decisions, verification and recovery.

EXAMPLE • STRAWBERRY GRAPHQL

The patch correctly documented how to select an older Apollo Federation version, but not why a reader would want to: to upgrade Strawberry while staying compatible with an older Apollo Router or Gateway.

It documented the setting and omitted the decision the setting exists to support.

The same narrowness at corpus scale

Agents can add accurate text to a page the reader is unlikely to visit, or update one page and leave others that cover the same thing out of date.

20.7% **13.3%**

hard to find

related pages left stale

A patch can be accurate in isolation and still fail inside the documentation system.

Why it happens: agents have no model of how the product is actually used

Trace-supported root causes across all 1,267 audited submissions. One or more per material problem.

36.0%

Explained the interface without checking how it is used

33.1%

Assumed instead of reading the decisive evidence

30.1%

Stopped at the first plausible page

27.5%

Read the evidence, didn't use it

27.3%

Committed early to a narrow reading of the task

26.3%

Generalized one example into a public rule

AI is not very good at detecting AI-slop, and hallucination has gotten subtler: less invention, more misstated facts

7.6%

had low information density: repetition, extra structure, long prose. The judges may under-count this — a looser prompt flagged only 3.2%.

6.1%

contained fabricated content: invented classes, flags, endpoints.

36.6%

had technical inaccuracies. The feature is real, but the patch overstates its scope, default or compatibility.

What this means for how we work

Review the decision, not just the prose

Sometimes, there shouldn't be any documentation for a particular code change at all. Writer still needs to triage.

Check scope and conditions, not just facts

Defaults, versions, lifecycle and compatibility are where accurate sentences go wrong.

Read for the reader's whole task

Check prerequisites, decision points, verification and recovery. Agents still rely on humans to the full user journey in mind.

Look beyond the page that was edited

Navigation, related pages and cross-references sit outside an agent's stopping point.