Stackademic

Member-only story

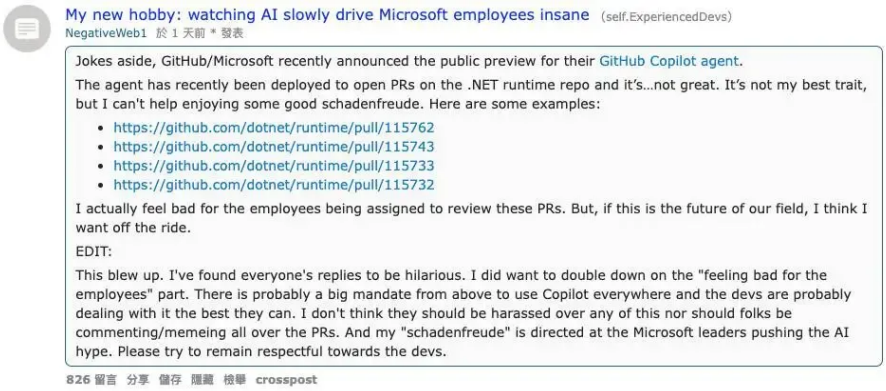
My New Hobby: Watching Copilot Slowly Drive Microsoft Engineers Insane

Dylan Cooper Follow 9 min read · May 27, 2025

296 23

A recent GitHub Copilot controversy has sparked widespread attention among developers on Reddit. One user posted a thread titled *"My New Hobby: Watching AI Slowly Drive Microsoft Employees Insane"*, poking fun at a series of *cringe-worthy* pull requests (PRs) submitted by Microsoft's Agent in the .NET repository. 1

According to the post, Copilot's performance in those PRs was far from satisfactory — so much so that the original poster admitted to feeling a guilty pleasure in watching the chaos unfold.



My new hobby: watching AI slowly drive Microsoft employees insane (self.ExperiencedDevs)

NegativeWeb1 於 1 天前 * 發表

Jokes aside, GitHub/Microsoft recently announced the public preview for their GitHub Copilot agent.

The agent has recently been deployed to open PRs on the .NET runtime repo and it's...not great. It's not my best trait, but I can't help enjoying some good schadenfreude. Here are some examples:

- https://github.com/dotnet/runtime/pull/115762
- https://github.com/dotnet/runtime/pull/115743
- https://github.com/dotnet/runtime/pull/115733
- https://github.com/dotnet/runtime/pull/115732

I actually feel bad for the employees being assigned to review these PRs. But, if this is the future of our field, I think I want off the ride.

EDIT:

This blew up. I've found everyone's replies to be hilarious. I did want to double down on the "feeling bad for the employees" part. There is probably a big mandate from above to use Copilot everywhere and the devs are probably dealing with it the best they can. I don't think they should be harassed over any of this nor should folks be commenting/memeing all over the PRs. And my "schadenfreude" is directed at the Microsoft leaders pushing the AI hype. Please try to remain respectful towards the devs.

826 留言 分享 儲存 隱藏 檢舉 crosspost

He also expressed sympathy for the Microsoft engineers forced to review these PRs, guessing that management might be **mandating** the use of Copilot, leaving developers to deal with the consequences. His "schadenfreude," he clarified, was mostly directed at Microsoft's leadership 1

— those who blindly promote automation and hype up AI without understanding its limitations.

The post quickly gained traction, attracting plenty of attention and discussion. Unexpectedly, it also opened the door to a broader conversation about the growing **gap between the promise of AI coding tools and the reality of using them in production.**

1

01 Copilot Agent Public Preview Sparks Controversy

This week, Microsoft officially launched GitHub Copilot Agent. According to the company, this AI-powered agent is built on advanced large language models and is capable of performing low- to medium-complexity development tasks within well-tested codebases. Its use cases include adding new features, fixing bugs, expanding test coverage, refactoring code, and enhancing documentation.

1

Based on some observed pull requests (PRs), Microsoft's engineering team appears to be experimenting with integrating Copilot Agent into the .NET repository — letting the AI automatically submit code modification PRs. These applications include low-level system fixes, cross-platform compatibility adaptations, regular expression logic corrections, and documentation improvements.

By putting Copilot Agent "on the front lines" for system-level maintenance tasks, Microsoft may have been trying to showcase the boundaries of AI-assisted programming — but the results seem to have backfired.

Coincidentally, just two days earlier at a developer conference, a Microsoft .NET engineer gave a presentation on how GitHub Copilot had become part of their day-to-day development workflow.

In the talk, the engineer noted that AI performed well in code development tasks, such as updating the regular expression engine, assisting with coding and bug fixing, and enhancing memory extension modules. Across the broader software development process, AI was also used to merge code, create and handle test cases. The engineer confidently claimed that AI had significantly boosted their productivity and had moved well beyond the "hype" phase.

Microsoft also explicitly pointed out during the presentation that regular expression engines are one of the most suitable areas for AI to "cut its teeth."

They emphasized that the .NET regular expression engine, originally created in 2003, had long remained stagnant — until AI brought it back to life. "With AI, the entire process of refactoring the engine has changed." According to Microsoft engineers, AI significantly increased efficiency — for example, in literal optimization, AI was able to quickly generate dozens of potential optimization paths, including novel approaches that human developers hadn't previously considered, such as lookahead anchor optimizations. Leveraging these AI-driven ideas, the team submitted a PR that will go live in .NET 10, expected to deliver a roughly 10x performance improvement by "dramatically reducing the amount of work needed."

02 The Harsh Reality of GitHub Practice

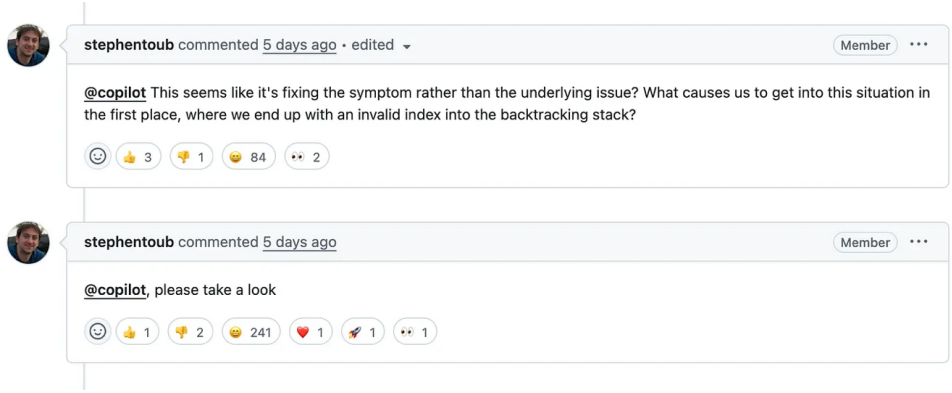
Ironically, it's in the domain where AI is supposedly most proficient — regular expressions — that Copilot's real-world performance on GitHub has most vividly exposed the gap between expectations and reality.

Take, for example, PR #115733, which attempted to address an `IndexOutOfRangeException` triggered by complex nested quantifiers. The goal was to ensure the regex engine behaves as expected—either matching or not matching—without crashing the application due to unexpected exceptions. Notably, this isn't the only PR where Copilot attempted to fix regex-related issues; several others exist, but their results have been similarly underwhelming.

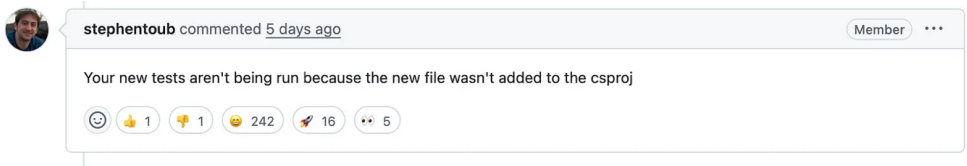
In the case of PR #115733, the full interaction between Copilot and human collaborators unfolded over roughly three days.

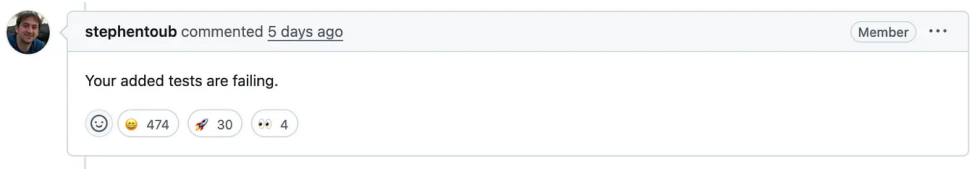
Three days prior, Copilot submitted the PR to the .NET Runtime repository and requested a code review. It also added a new test file — but failed to include it in the project, meaning the tests didn't run at all.

The next day, senior Microsoft engineer @stephentoub commented that Copilot's fix was merely superficial and didn't actually resolve the root cause behind the broken stack backtracking. He also pointed out that the test file wasn't functioning.



Copilot then attempted to fix the build issue and submitted an updated version of the tests — but once the tests were added to the project, they failed immediately.





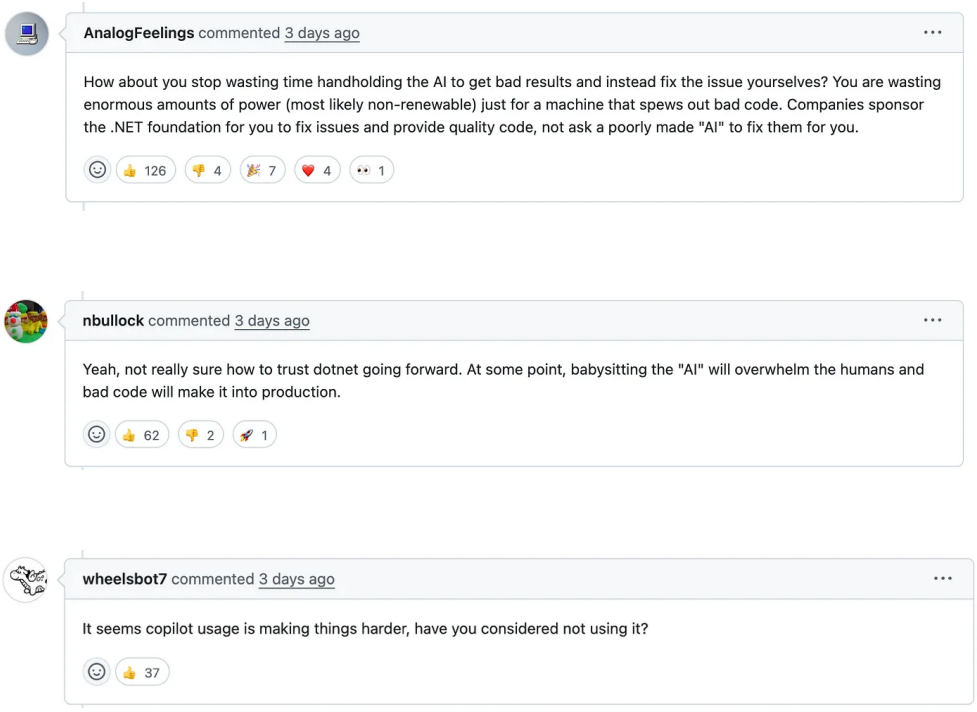
Eventually, community developers stepped in, clearly frustrated. Several questioned whether Copilot was wasting their time, suggesting the AI should delete the rewritten tests — or just close the PR altogether.

As of now, the PR has not delivered an effective solution nor received

approval from reviewers, marking it as a temporary failure. Many developers voiced strong dissatisfaction in the comments. Some criticized Microsoft for “pouring resources into an AI that only generates broken code,” arguing it wastes compute power, pollutes the codebase, and deviates from the .NET Foundation’s mission of delivering high-quality code.

Others expressed concern that these AI tools, which still require human “babysitting,” could overwhelm maintainers — and eventually let unqualified code slip into production, jeopardizing the long-term sustainability and trustworthiness of the project.

Some questioned whether Microsoft is envisioning Copilot as part of a “programming utopia” that replaces developers — an idea made even more ironic given the recent wave of layoffs.



03 Is This a Strategic Lesson from Inside Microsoft?

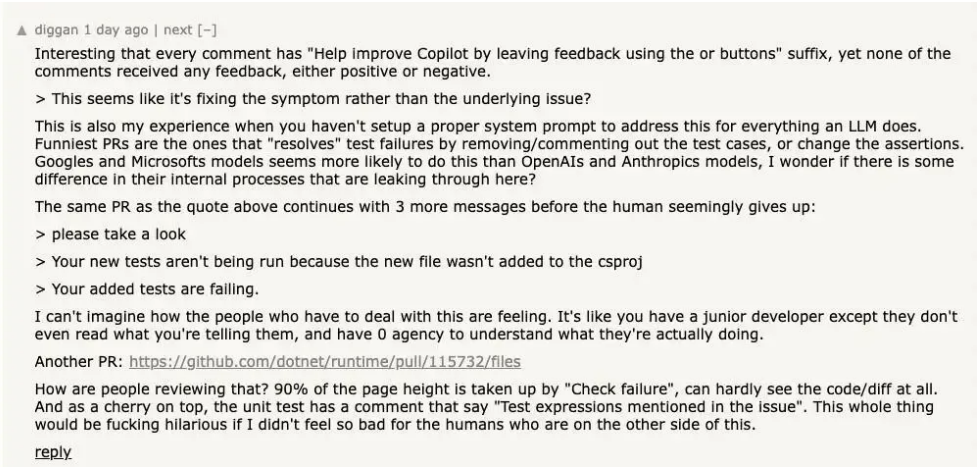
Photo by Matthew Manuel on Unsplash

In recent days, as abnormal PR behavior spread rapidly across Hacker News and Reddit, skepticism within the developer community has only intensified. Many engineers candidly admit: “We’re still quite far from truly reliable automated development.” What frustrates people most isn’t just the faulty code Copilot submits — but the fact that every comment is followed by the line “Help improve Copilot with feedback,” and yet not a single piece of feedback appears to be acknowledged. Even more ironically, some PRs attempted to “fix” failing tests by deleting or commenting them out, or by tweaking assertions.

Top highlight

What’s worse, even when humans explicitly comment things like “The test wasn’t added to the project file” or “The new test failed,” the AI continues to make the same mistakes. The result is a collaboration experience that feels like working with a “junior developer who doesn’t read feedback and has no idea what they’re doing.”

“I can’t imagine what it must feel like for the people who have to deal with this… If I didn’t feel bad for them, this whole thing would honestly be hilarious,” one developer commented.



▲ diggan 1 day ago | next [–]

Interesting that every comment has "Help improve Copilot by leaving feedback using the or buttons" suffix, yet none of the comments received any feedback, either positive or negative.

> This seems like it's fixing the symptom rather than the underlying issue?

This is also my experience when you haven't setup a proper system prompt to address this for everything an LLM does. Funniest PRs are the ones that "resolves" test failures by removing/commenting out the test cases, or change the assertions. Googles and Microsofts models seems more likely to do this than OpenAIs and Anthropics models, I wonder if there is some difference in their internal processes that are leaking through here?

The same PR as the quote above continues with 3 more messages before the human seemingly gives up:

> please take a look

> Your new tests aren't being run because the new file wasn't added to the csproj

> Your added tests are failing.

I can't imagine how the people who have to deal with this are feeling. It's like you have a junior developer except they don't even read what you're telling them, and have 0 agency to understand what they're actually doing.

Another PR: https://github.com/dotnet/runtime/pull/115732/files

How are people reviewing that? 90% of the page height is taken up by "Check failure", can hardly see the code/diff at all. And as a cherry on top, the unit test has a comment that say "Test expressions mentioned in the issue". This whole thing would be fucking hilarious if I didn't feel so bad for the humans who are on the other side of this.

reply

However, comparing large language models (LLMs) to junior developers sparked pushback from some. One user argued that it was unfair to real junior developers, who typically learn quickly and don't repeat the same rookie mistakes. He suggested that LLMs might function well as "idea generators or task accelerators for repetitive work," but they're still far from replacing interns — let alone full-time engineers.

▲ surgical_fire 1 day ago | parent | next [–]

> I can't imagine how the people who have to deal with this are feeling. It's like you have a junior developer except they don't even read what you're telling them, and have 0 agency to understand what they're actually doing.

That comparison is awful. I work with quite a few Junior developers and they can be competent. Certainly don't make the silly mistakes that LLMs do, don't need nearly as much handholding, and tend to learn pretty quickly so I don't have to keep repeating myself.

Many observers suspect this isn't just a single team's technical misstep but part of a broader, top-down initiative at Microsoft to roll out Copilot company-wide. Other users found similar PRs in different repositories, with Copilot agents appearing across multiple projects in a synchronized fashion. It appears to be a centrally coordinated effort, with frontline developers doing their best to support a strategy that's not yet fully mature.

In that viral Reddit post, the author clarified that their "schadenfreude" wasn't directed at Microsoft's engineers but squarely at the company's leadership, who are aggressively pushing AI despite its flaws. The poster expressed sympathy for the developers caught in the middle.

In fact, Microsoft's leadership intentions have long been visible. Just days ago, CTO Kevin Scott said in a podcast:

> "I think the folks who are still saying 'the tech's not ready yet' — because maybe they tried it and felt it's a little expensive or lacking in some areas — if they choose to wait, they'll likely fall behind very quickly. Everything will be cheaper and more powerful in the next year. That's something I feel very confident about. By 2025, this won't even be a controversial opinion."

This top-down push for AI adoption isn't unique to Microsoft. A similar trend is emerging across the tech industry: AI is increasingly being introduced into development workflows from above, rather than waiting for bottom-up developer adoption.

Microsoft's own Brian Houck, a senior applied scientist studying developer productivity, shared insights on how they encourage AI usage internally. He said the company has been working on increasing adoption among engineers, and emphasized:

> **"When leadership strongly champions the use of AI tools, it makes developers seven times more likely to become regular users."**

More than just encouragement, this top-down advocacy signals to teams that "this tech is safe and ready to use."

In addition to leadership pressure, Brian also highlighted the role of "local champions" — those senior, influential developers within teams. Their example can have a tangible impact, making it easier for AI tools to be integrated into daily workflows. Microsoft's observations suggest that teams with these advocates have an AI adoption rate about 22% higher, making it more feasible to systematize and operationalize AI use in software development.

This might also explain why Microsoft chose the .NET repositories for its AI programming experiment: they may be seen as an ideal "test bed" for

ecosystem.

Reflections Behind the Controversy

At the end of the day, the debate surrounding Copilot isn't just about a few "weird PRs." It's more like a mirror — revealing the gap between lofty technological ideals and their messy real-world deployment.

On one side, you have company leadership brimming with confidence in the future of AI, eager to roll it out everywhere as quickly as possible. On the other side, there are tools that aren't fully ready, frontline developers overwhelmed with day-to-day work, and a developer community that's becoming increasingly wary of AI narratives that sound too good to be true.

Perhaps the real question we should be asking isn't whether AI can write perfect code. It's whether developers are given enough space — amid this top-down technological shift — to question, to experiment, and to iterate. Only then can we develop workflows that actually fit the realities of modern software development.

Ultimately, the key to whether AI coding tools can truly "take off" lies in our ability to strike a balance between automation and reality.

Reference Links:

- https://old.reddit.com/r/ExperiencedDevs/comments/1krttqo/my_new_hobby_watching_ai_slowly_drive_microsoft/?limit=500
- https://news.ycombinator.com/item?id=44050152
- https://getdx.com/podcast/brian-houck-ai-adoption-playbook/
- https://www.youtube.com/watch?v=gieL0bxyTUU
- https://www.youtube.com/watch?v=jYHLKtWM164

Coding Programming Technology Llm AI

Published in Stackademic

41K followers · Last published 1 day ago

Follow

Stackademic is a learning hub for programmers, devs, coders, and engineers. Our goal is to democratize free coding education for the world.

Written by Dylan Cooper

14.3K followers · 23 following

Follow

A Software Engineer at an Internet Company. Follow me for a dose of practical tech knowledge! Let's explore the digital world together. 🚀💻

Responses (23)

Gilles Sérasset

What are your thoughts?

Lars van Gemerden
1 day ago

In theory, theory and practice are the same. In practice they are not.

21 1 reply Reply

Boss Hog
2 days ago (edited)

This “growing gap between the promise of AI coding tools and the reality of using them in production” is precisely what triggered the AI Winter in the late ‘80s, on into the ‘90s. Entire companies died, people lost their jobs, and AI research money... more

24 1 reply Reply

NTTP
3 days ago

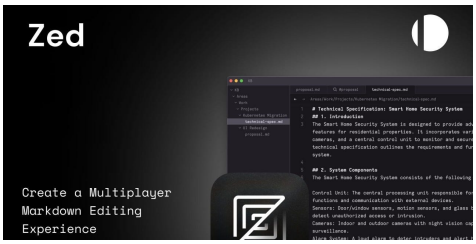
> They emphasized that the .NET regular expression engine

Is this really the nexus of problems in computing today? Shakin' my head...

12 1 reply Reply

See all responses

More from Dylan Cooper and Stackademic



In Stackademic by Dylan Cooper

Better Than Cursor or Windsurf? A “World’s Fastest AI Code Editor”...

While VS Code and its AI-enhanced forks continue to dominate developers’ workflows...

May 12 190 10

In Stackademic by Mark Henry

11 Ways To Make Money with Coding (My Earning Proof) $

I Made $3,500 as a Beginner Developer Last Month✌

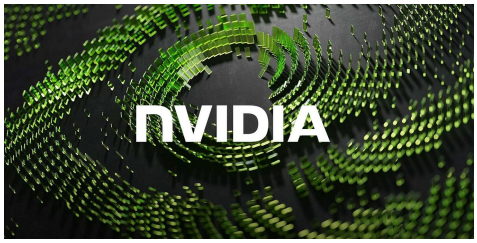
Feb 27 765 30

In Stackademic by Deepanshu Chauhan

The Silent Performance Killer in .NET:

How ignoring a simple token in .NET can lead to wasted resources, poor performance, and...

May 16 699 7



In Python in Plain English by Dylan Cooper

A New Era for GPU Programming: NVIDIA Finally Adds Native Pytho...

This year, NVIDIA is going all in, making it clear that Python is to become a first-class...

Apr 17 901 7

See all from Dylan Cooper | See all from Stackademic

Recommended from Medium

In Coding Nexus by Code Coup

6 MCP Servers Every Developer Needs to Try

I've spent the last two months around with over 100 MCP servers, and let me tell you,...

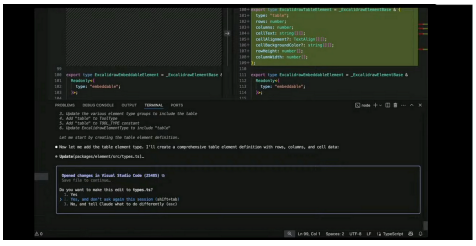
May 27 637 9

In Level Up Coding by Saravanan M

Everyone's Talking About Zed, the new AI Code Editor. So I Gave it a...

and here's what worked, and what didn't.

May 17 94 5

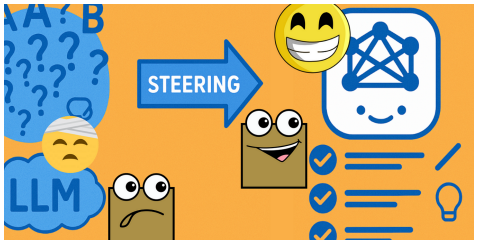


In Coding Beauty by Tari Ibaba

The new Claude 4 coding model is an absolute game changer

Woah this is really huge.

6d ago 386 11

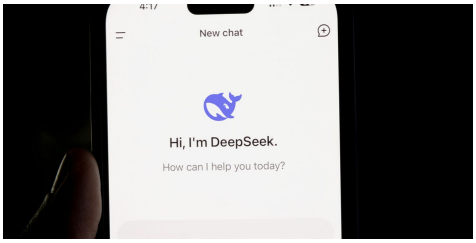


In AI Advances by Nikhil Anand

I made my LLM's outputs 50% better with ZERO training.

It's wild how few people are talking about this.

May 27 446 8

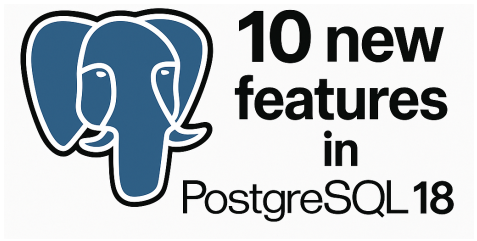


In Data Science in Your Pocket by Mehul Gupta

DeepSeek-R1-0528: The new DeepSeek model will bankrupt...

How to use DeepSeek-R1–0528 for free?

5d ago 253 8



Devlink Tips

PostgreSQL 18 just dropped: 10 powerful new features devs need ...

From parallel COPY to smarter replication and JSON wizardry this release might just b...

May 27 80 2

See more recommendations