You're Fired

The State of Coding Assistants & Vibe Coding

The Economic Shift - will you be FIRED?

AI Reshapes Software Development

Accelerated Productivity
AI generates 30% of Microsoft's code; 92% of developers use AI tools.

Market Disruption
Startups lead AI adoption, while "code-gen" investments soar.

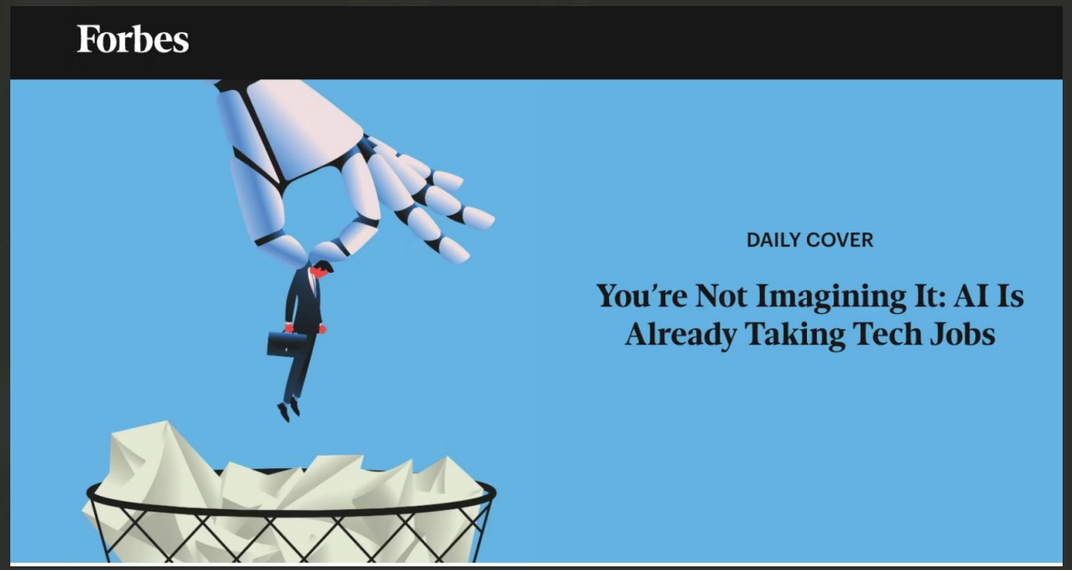
Trillions in Value
Generative AI projects a $7.9T economic boost, impacting development.

Concerns arise about job displacement, especially for entry-level roles, as affordable AI agents enter the market. However, CTOs agree AI evolves roles, not eliminates them, shifting focus to critical thinking and AI orchestration.

The Economic Shift - will you be FIRED?

FORTUNE

Stunning new data reveals 140% layoff spike in July, with surge in AI and ‘technological updates’ increasingly apparent



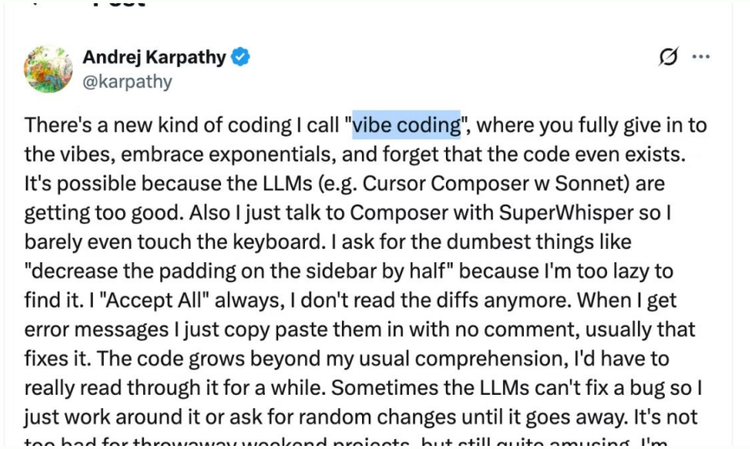
What Is Vibe Coding?

Natural-language prompts

Focus shifts from manual typing to **design, validation, and collaboration**

Key observation:

- Software development by non-developers
  - Designers
  - Product Managers



Originally coined on Twitter by Andrej Karpathy

Vibe Coding Platforms

Lovable

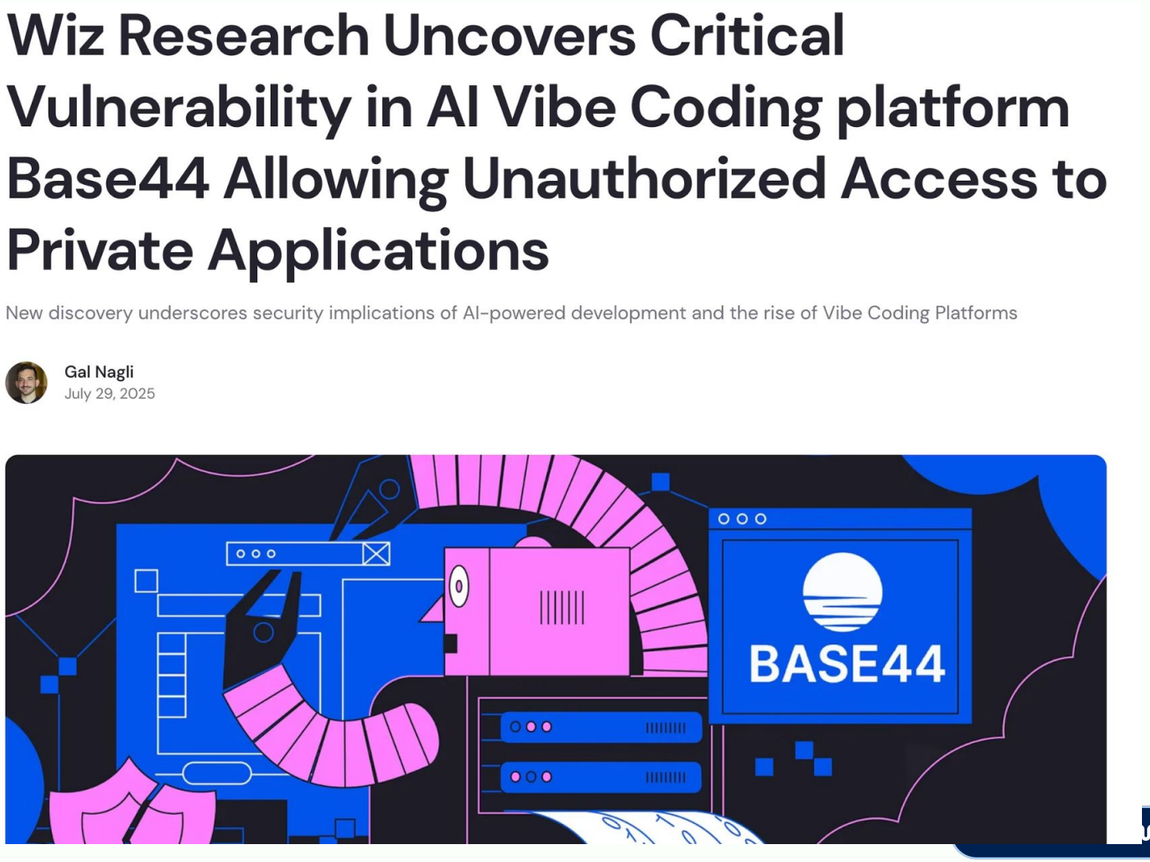
- 2M users
- Raised $200M, valued at $1.8B

Base44

- 250K users in just 6 months
- 80M acquisition

Vibe Coding Challenges

- Struggles with **Complex** apps
- Struggles with **Security**
- Struggles with **Robustness**



Taxonomy of AI Coding Assistants

1

No-code/Low-code

Lovable, Base44

2

IDE-integrated

GitHub Copilot, Tabnine, Cursor, GitLab Duo

3

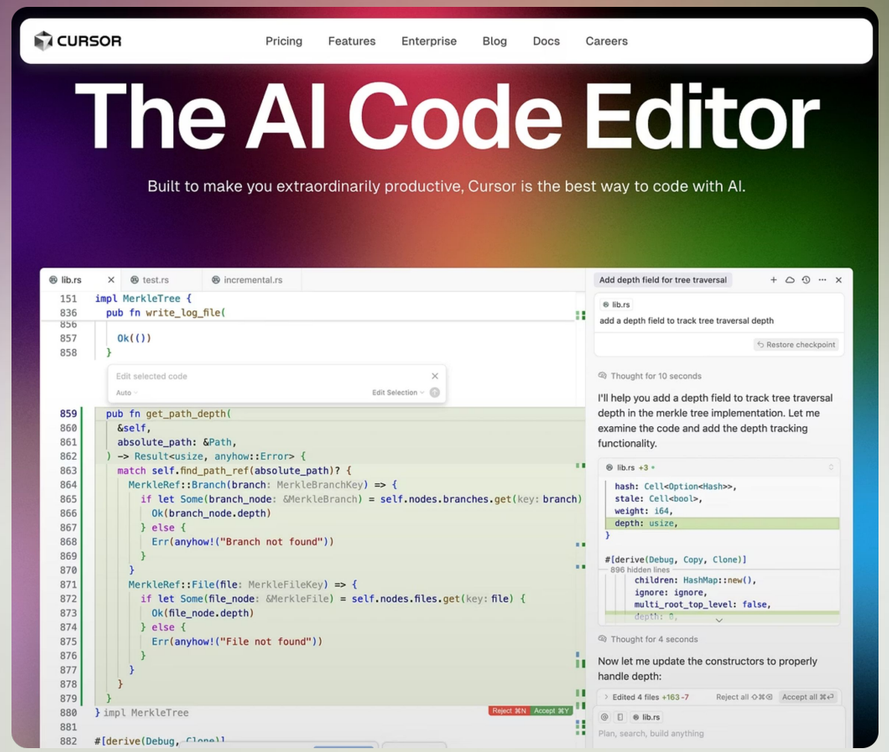
CLI-based and Background Agents

Claude Code, Gemini CLI, Aider, cursor-agent, Codex

4

Use-case-specific

Qodo (testing/PR), CodeRabbit (review)



IDE-integrated Assistants

GitHub Copilot

Real-time code completion

Chat assistance (Enterprise)

Tabnine

Multi-language support

Enterprise privacy focus

Cursor

Project-aware completions

VS Code-based refactoring

GitLab Duo

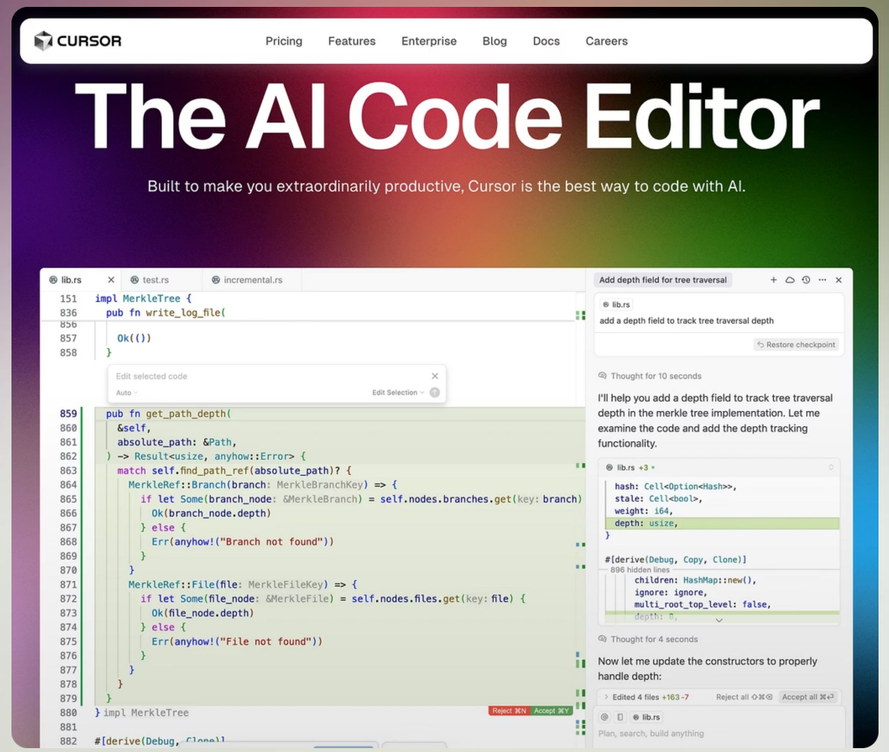
GitLab IDE and Platform integrated

Cline

Any vendor model

Gemini

Generous free usage



Cursor workflows

Smart code completions

Start typing and wait for completion

Inline Edit

Highlight a selection and ⌘+K

tell Cursor what to do

Agent

E.g. Python developer,

Security reviewer etc. ⌘+E

Background by default

Chat

Architect, Plan, Learn

⌘+I

Rules

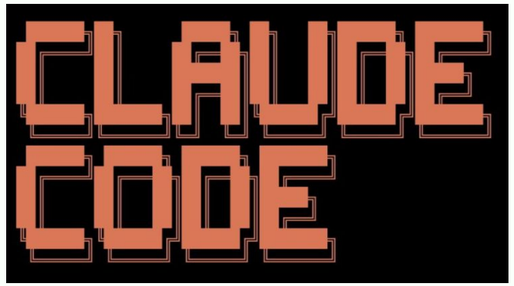
Provide rules in

`.cursor/rules/`

MCP

Provide MCP servers

CLI-based Pair Agents



Claude Code

Multi-file edits

Git-aware workflows

Gemini CLI

Generic coding

Media generation

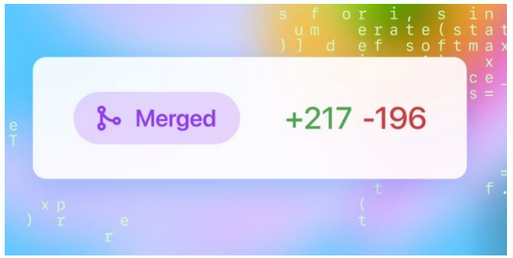
GitHub Actions

Aider

Clean commit messages

Reversible changes

High control.



OpenAI Codex

Mainly for background tasks such as Code Review and automatic issues handling.

Claude Code Workflows

/commands

For common and repeatable tasks in `.claude/commands`

Print mode

`claude -p "something"`

CLI non-interactive mode

Sub-Agent

Create sub-agents with speciality in `.claude/agents`

CLAUDE.md

Provide rules in `CLAUDE.md` files

MCP

Provide MCP servers

IDE

Minimal IDE integration

Claude Code Sub-Agents v/s Cursor Agents

Aspect	Claude Code Subagents	Cursor Agents
Agent Identity	Explicitly defined (e.g., Debugger, Doc Writer, Test Generator)	Implicit modes/behaviors, not exposed as named entities
Manifest / Spec	Declared in `.claude/agents`	No manifest; embedded in the IDE
Foreground Usage	Yes – you explicitly invoke with `/commands` or by routing tasks to a subagent	Yes – you can chat or issue a direct request (e.g., “write tests”)
Background Usage	Possible by rare – subagents don’t run implicitly unless you call them	Common – auto-suggestions, inline completions, proactive refactors
Transparency	High – you always know which subagent is acting	Lower – “which agent” is acting is invisible; feels like one assistant
Control	Strong – you can choose which tool/subagent to use	Weak – Cursor decides which behavior to apply, context-driven
Mental Model	Toolbox of agents you select from	Single assistant who shifts roles

Key Takeaway

Claude Code: Foreground-first. Subagents don’t really run in the background unless explicitly asked.

Cursor: Background-first. Often act in the background, can respond in foreground.

MCP Servers for Coding Agents

MCP (Model Context Protocol) provide powerful extension. A few examples:

Linear

Manage **tasks and issues** directly from the agent, including auto-creation of bug tickets and task assignments.

GitLab

Interact with **repositories**, manage **merge requests**, trigger CI/CD pipelines, and review code differences.

Context7

Provide open source l**ibrary documentation** in machine-friendly format

Sequential-Thinking

Offload complex, **structured reasoning steps**, enabling agents to execute multi-step planning and problem-solving.

Key Concept: MCP provides a standardized framework, empowering coding agents to act far beyond the confines of the IDE.

Use-case-specific Agents


```
11
2
3
4
9
8
46
25
30
46
29
25
45
00
37
20
39
10
20
30
30
20
30
20
31
30
30
```

```
topctive (
  veld pctior, ines agtb
  cod or = todade botion apb
      vlect; 17 spssiont-=Cizer, 0;
{   ter lot tar  ofdteriop copb (
      lec fotiot: ABI]s gecceentorit,  conolt-vgl);
      leav tore apptinntr sadusalyctif:
      ingrative 0);
      chose add becrased
{
      ler sofe = ipfo;
      lec toller: EBI]is ganaseatgrily;
      teast 101(; Acnol or aecape;
          = slaxst,
      intect formant with vcabalet coders,
      taple: juali};
 }
      lec tollor: ABI]is resmolle, opooficariop );
      teast (fichesed to rappostalty);
      Msors_codes= b;
      ler tollor: EBI]is reample: cupcoticar-og>))
      was agder borld for achealjyute:
      les te rafalvent  = llef;
      these: Il fuccade jynut:
}
```

1 AI Cinens
Use relique colerticto ecuction time is the pesive alocar as ourprectol sterfeut conneattlina.

2 Reviews
Percopantc sedalal lotations douoge effery menraal corfiee fires.

2 Adsenti
for rease accentatis nurd the this sbart toopentalble.

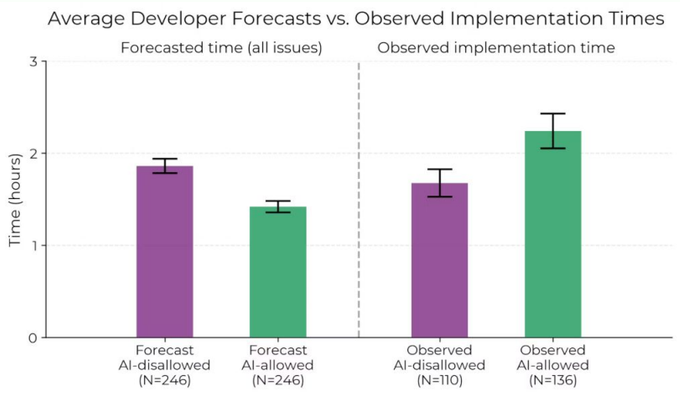
1 AI Clous
Yoe reyour de eccenition technalk allect and bour ae inellde.

Open Challenges - Productivity, Usability, Reliability

While AI coding assistants offer immense promise, their real-world implementation reveals several significant hurdles:

Productivity Slowdown

AI tools **increased completion time by ~19%** despite initial **expectations of a 24% speedup**.



[Measuring the Impact of Early-2025 AI on Experienced Open-Source...]

Usability Frustrations

- AI tools need guidance. It takes an effort
- They sometimes produce Gold, but sometimes Sh*t

Slow ⏳

Enterprise Reliability

- Large code-base
- Internal know-how and undocumented knowledge not visible to agents
- Background and long-running process support

Continuing Challenges: Code Quality & Security

Security Vulnerabilities

A Veracode study on over 100 AI models revealed that **45% of AI-generated code contained known OWASP-class vulnerabilities** like XSS and SQL injection.

Another analysis of GitHub projects showed that **29.5% of Python and 24.2% of JavaScript code snippets from Copilot had security weaknesses**

Challenges with Codebase-Scale Generation

Developers' satisfaction with generated codebases averaged only **~2.8/5**. Common dissatisfactions included poor functionality (77%), low code quality (42%), and inadequate communication (25%).

Adoption of AI-Aided & Vibe-Coding Tools (2025)

90%

Engineering Teams

Incorporate AI coding tools (up from 61% last year).

78%

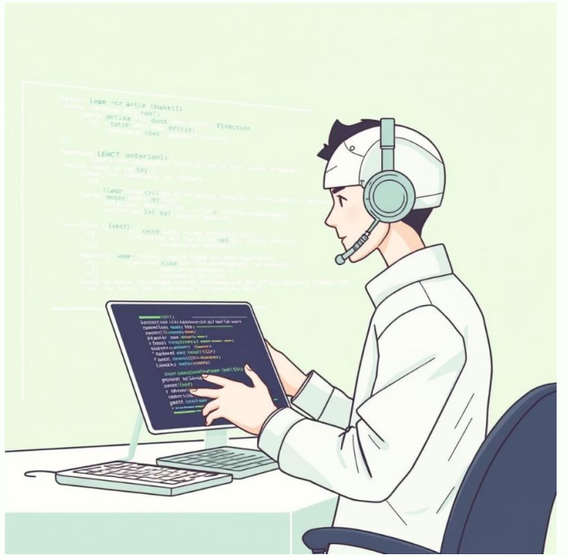
Developers Rely on AI

Multiple times per week for development tasks.

75%

Enterprise Adoption

Forecasted by Gartner for enterprise software engineers by 2028.



"Vibe Coding" Hits the Mainstream

Developer Backlash & Risks

59% of developers use AI-generated code without full understanding, raising significant security risks.

Key Takeaways

Vibe Coding is not AI assisted coding

Software professional - better use AI-assisted for your day to day

Diverse ecosystem

Tools specialize in different aspects of development workflow

AI-assisted development is shaping the future

Embrace it, or risk being left behind in skill and opportunity.

AI won't get you fired.

But not using AI - will!