Muhammad Asad · AI Implementation & Growth

Case Study · N° 09

PRODUCT / MCP INFRASTRUCTURE SOLO

My own MCP platform, live on npm

A SaaS plus stdio MCP server that keeps my private AI-context library out of every client repo, pulled in over MCP and removed clean when the engagement ends.

Solo · end to end

DISCIPLINE	Product / MCP Infrastructure
STACK	Next.js • TypeScript • MCP (stdio, 10 tools) • npm @asad120414/mcpcall • Postgres • sha256-keyed sync
LIVE / VERIFY	team-agents-platform-web.vercel.app ↗

MY ROLE

Built solo, end to end: the platform, the stdio MCP server, the data model, the migrations, and the published npm package. Live free beta.

10	1	0
MCP TOOLS	COMMAND TO PULL	FILES LEAKED TO CLIENT

THE PROBLEM

Across client projects I bring years of my own AI-context files (CLAUDE.md, agents, skills, rules, research). The manual workaround is to copy them in and .gitignore them, but .gitignore is committed, so even the file names leak, and there is no clean off-boarding.

WHAT I DID

- Built a SaaS that stores context per workspace (one per client) and a stdio MCP server (10 tools) that pulls it into any local checkout in one command.
- Hid pulled files via .git/info/exclude, the per-repo user-local ignore file that is never committed, so the client sees neither the files nor the rule, unlike .gitignore.
- Made every operation sha256-keyed: a pull writes a .incoming sibling on a hash mismatch instead of clobbering, and cleanup refuses to delete a file whose local hash has drifted.
- Shipped it on npm as @asad120414/mcpcall (npx, no install), tokens hashed at rest with timingSafeEqual, symlink-defended, with a self-host URL override.

THE RESULT

Live free beta with open signup, published on npm. One cleanup call leaves a client repo bit-identical to before. I run my own context library on it day to day.

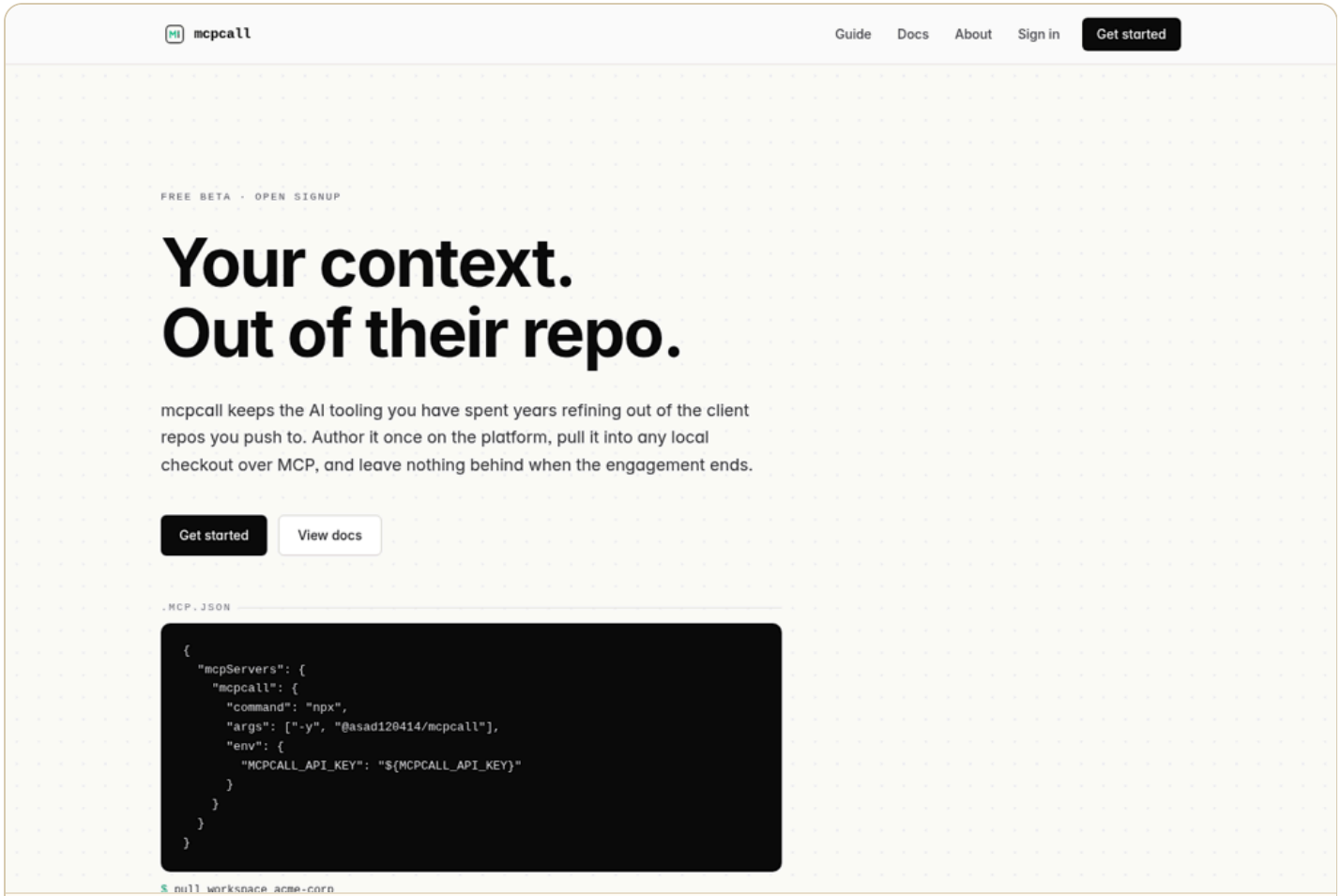
THE JUDGMENT CALL · WHAT THE AI COULDN'T DO

Everyone reaches for .gitignore to hide local files. But .gitignore is committed, so it leaks your file names to the client. I used .git/info/exclude instead, the per-repo ignore file git never tracks, so the client sees neither the files nor the rule. Knowing which ignore file git actually commits is the difference between working and leaking.

PROOF

Live platform: team-agents-platform-web.vercel.app ↗

SEE IT



The mcpcall platform, live in free beta.