Telefónica Innovación Digital

Reducing AI FOMO

VIDEO Team - CafeterIA

27 February 2026

Fear Of Missing Out

What you're feeling has a name

01

There is big money making you feel this way

Every day a new model. Every week a new tool. Everyone seems to know more than you.

- **$600B+** invested in AI infra this year -> they *need* you to feel behind
- Google & Microsoft paying influencers -> **$400–600K** to flood your feed
- The noise is designed to feel **urgent** — it's a product strategy

Remember

There are **$600 billion reasons** for you to feel this way. Being aware of it makes you **clearer**.

Source: El negocio de crearte ansiedad con la IA — Simón Muñoz

What I’m trying to do here

This talk has a simple success metric:

IF YOU LEAVE WITH MORE FOMO

I failed

↓ 100%

Let's avoid this

IF YOU LEAVE WITH LESS FOMO

I succeeded

↑ 100%

This is the goal

No pressure. Just 15 minutes.

Where we are

AI has been moving fast. Some people are already **Level 12**.

This talk targets **Level 1**.

Realizing this gap spikes the FOMO.

Knowing the field reduces the FOMO.

GOAL: Get from Level 1 → Level 2

Lvl 10

Some people

Lvl 1

We

Understanding the tools

Models, clients, and how they fit together

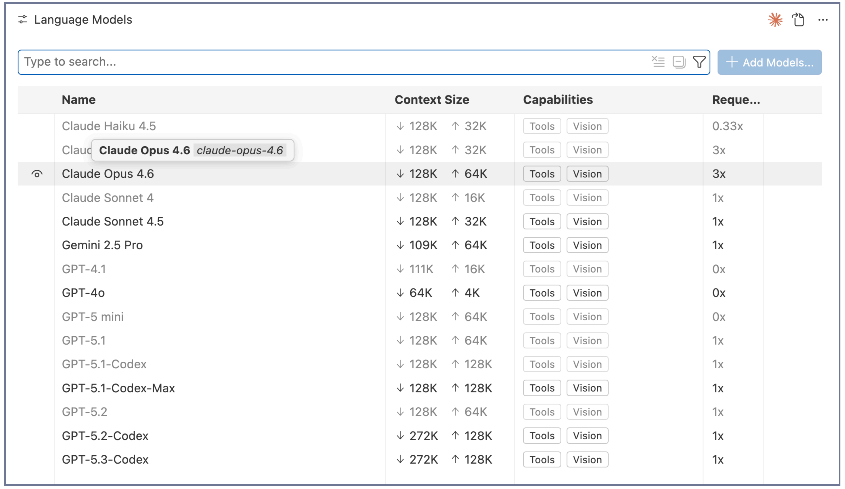
02

Models and clients are separate things

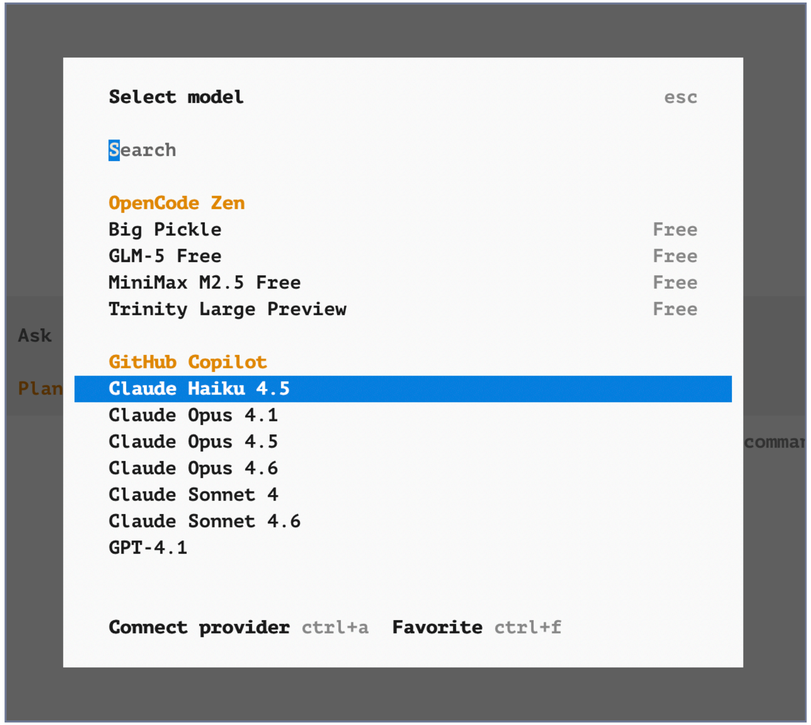
Client = the interface you see

Model = the brain doing the work

You can **mix and match**.



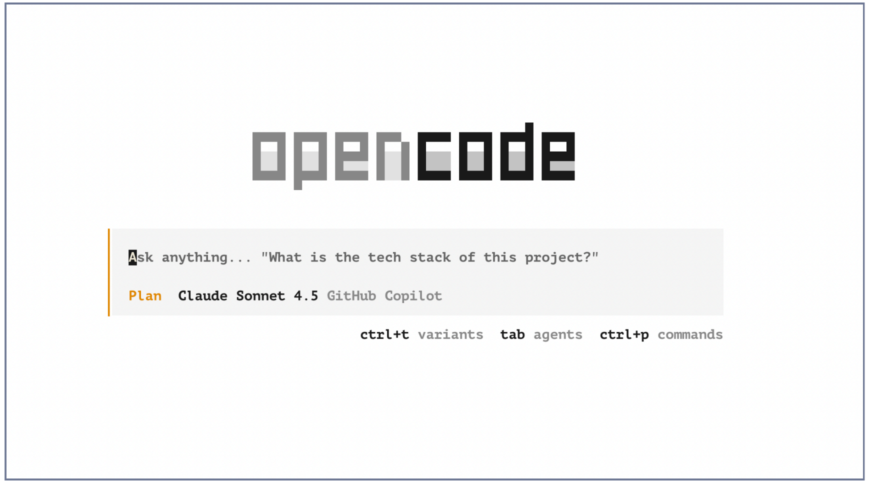
Name	Context Size		Capabilities		Reque...
Claude Haiku 4.5	↓ 128K	↑ 32K	Tools	Vision	0.33x
Clauc	↓ 128K	↑ 32K	Tools	Vision	3x
Claude Opus 4.6	↓ 128K	↑ 64K	Tools	Vision	3x
Claude Sonnet 4	↓ 128K	↑ 16K	Tools	Vision	1x
Claude Sonnet 4.5	↓ 128K	↑ 32K	Tools	Vision	1x
Gemini 2.5 Pro	↓ 109K	↑ 64K	Tools	Vision	1x
GPT-4.1	↓ 111K	↑ 16K	Tools	Vision	0x
GPT-4o	↓ 64K	↑ 4K	Tools	Vision	0x
GPT-5 mini	↓ 128K	↑ 64K	Tools	Vision	0x
GPT-5.1	↓ 128K	↑ 64K	Tools	Vision	1x
GPT-5.1-Codex	↓ 128K	↑ 128K	Tools	Vision	1x
GPT-5.1-Codex-Max	↓ 128K	↑ 128K	Tools	Vision	1x
GPT-5.2	↓ 128K	↑ 64K	Tools	Vision	1x
GPT-5.2-Codex	↓ 272K	↑ 128K	Tools	Vision	1x
GPT-5.3-Codex	↓ 272K	↑ 128K	Tools	Vision	1x



```
Select model                                        esc

Search

OpenCode Zen
Big Pickle                                         Free
GLM-5 Free                                         Free
MiniMax M2.5 Free                                  Free
Trinity Large Preview                              Free

GitHub Copilot
Claude Haiku 4.5
Claude Opus 4.1
Claude Opus 4.5
Claude Opus 4.6
Claude Sonnet 4
Claude Sonnet 4.6
GPT-4.1


Connect provider ctrl+a  Favorite ctrl+f
```

The clients

- GitHub Copilot
- Claude Code
- Cursor
- Windsurf
- Codex
- OpenCode



```
Ask anything... "What is the tech stack of this project?"

Plan  Claude Sonnet 4.5 GitHub Copilot

                    ctrl+t variants  tab agents  ctrl+p commands
```

Does pairing client + model matter?

Opinion A

The specific client+model pair creates a different experience; they're **tuned together**.

Opinion B

The UX (devExperience) differs. The output quality, **Not significantly**.

My recommendation

Use the client you're most comfortable with.

What actually matters is the **model** and **how you talk to it**.

Level 2: *how* you interact

The client is just a window. The model is the engine.
Your prompts, context, and structure is what actually matter

The buzzwords

Let’s name them so they stop sounding scary

03

The buzzwords

Agents

AI operating in multi-step, autonomous loops

Skills

Reusable, scoped instructions you give the agent

MCPs

Model Context Protocol; structured tools the model can call

RAG

Retrieval-Augmented Generation; feeding external data to the model

The buzzwords

Agents

AI operating in multi-step, autonomous loops

Skills

Reusable, scoped instructions you give the agent

MCPs

Model Context Protocol; structured tools the model can call

RAG

Retrieval-Augmented Generation; feeding external data to the model

The buzzwords

Agents

AI operating in multi-step, autonomous loops

Skills

Reusable, scoped instructions you give the agent

MCPs

Model Context Protocol; structured tools the model can call

RAG

Retrieval-Augmented Generation; feeding external data to the model

The buzzwords

Agents

AI operating in multi-step, autonomous loops

Skills

Reusable, scoped instructions you give the agent

MCPs

Model Context Protocol; structured tools the model can call

RAG

Retrieval-Augmented Generation; feeding external data to the model

Agents: you’re already using one

An agent = AI that takes actions in a loop, using tool calls and observations to make progress toward a goal
Claude Code has two modes:

Mode	What it does
Plan	Thinks, reasons, proposes steps
Act	Executes, writes, runs commands

The modes control how much autonomy you give it.

Plan
Think before acting

⇩

Act
Execute the plan

You've been using an agent. You just didn't call it that.

What is `AGENTS.md`?

We're using an agent. We write down the instructions and give it context.

How do you tell it about the project? **We write it down.**

If we are re-writing a similar context every single time, the idea to "reuse" a basic context

Initially I read it as `CONSTITUTION.md`

Idea: A file you put at the root of your project. just **instructions** that always are considered.

The model reads it **automatically** at the start of every session.

"Here are the rules you must follow ALWAYS"

```
# AGENTS.md

This project uses TypeScript.
Run tests with: npm test
Follow conventional commits.
Never push to main directly.
Error codes are mandatory in log calls, every
logger.error() needs a code
```

Current debate: fills the context window fast. Is it worth it if the model could discover most of that info anyway?

Avoid redundant info that the agent will assume anyway

Same idea, different names

Each client has its own version of the "project instructions" file:

File	Client
`CLAUDE.md`	Claude Code
`.cursorrules`	Cursor
`.windsurfrules`	Windsurf
`.github/copilot-instructions.md`	GitHub Copilot
`.clinerules`	Cline
`AGENTS.md`	Codex / generic

Different filename, **same concept**: persistent context the model reads at the start of every session.

Same idea, different names

Each client has its own version of the "project instructions" file:

File	Client
`CLAUDE.md`	Claude Code
`.cursorrules`	Cursor
`.windsurfrules`	Windsurf
`.github/copilot-instructions.md`	GitHub Copilot
`.clinerules`	Cline
`AGENTS.md`	Codex / generic

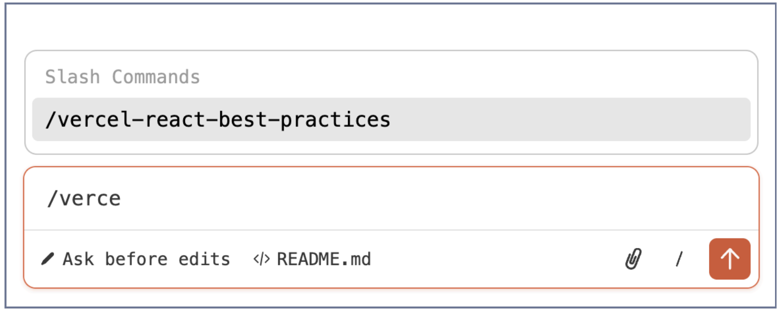
Different filename, **same concept**: persistent context the model reads at the start of every session.

Skills: a better alternative

Instead of dumping everything in `AGENTS.md`:

Skills are modular, **on-demand instruction sets**.

- Loaded **only when relevant**
- Scoped to a **specific task**
- Don't pollute the context window
- Invoked via **slash commands** (`/skill-name`)



Slash Commands

`/vercel-react-best-practices`

`/verce`

AGENTS.md
Always-on background noise

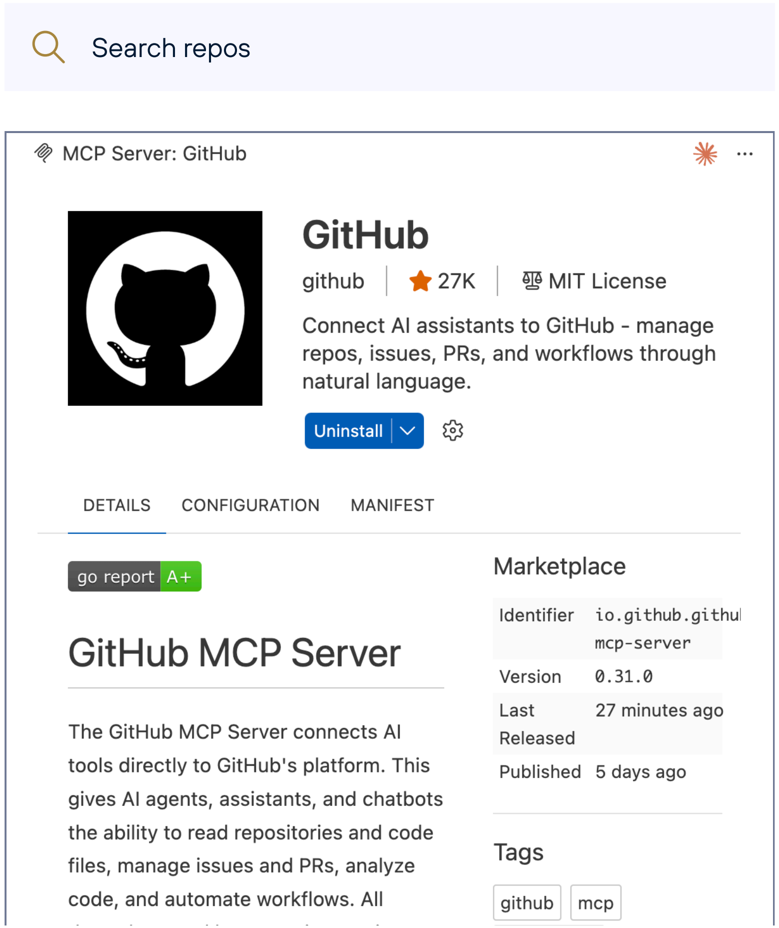
vs

Skills
Called when needed, focused, efficient

MCPs: giving the model real tools

Model Context Protocol = a standard way for models to call external tools.

The model doesn't browse GitHub. It calls a **structured tool** that does.



Search repos

MCP Server: GitHub

GitHub

github | 27K | MIT License

Connect AI assistants to GitHub - manage repos, issues, PRs, and workflows through natural language.

GitHub MCP Server

The GitHub MCP Server connects AI tools directly to GitHub's platform. This gives AI agents, assistants, and chatbots the ability to read repositories and code files, manage issues and PRs, analyze code, and automate workflows. All

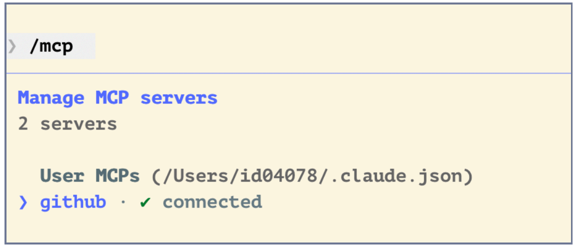
Marketplace

Identifier	io.github.githu mcp-server
Version	0.31.0
Last Released	27 minutes ago
Published	5 days ago

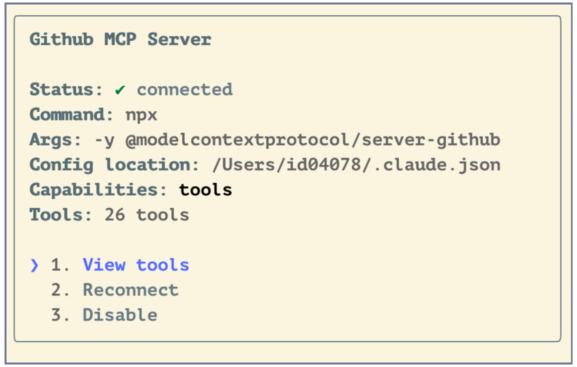
Tags

github mcp

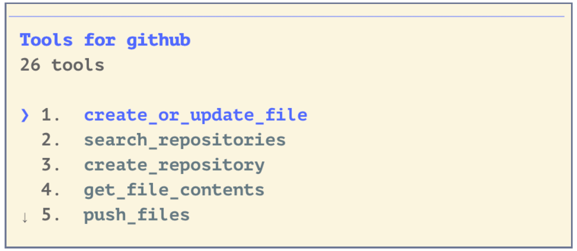
Using MCP Servers



```
> /mcp

Manage MCP servers
2 servers

  User MCPs (/Users/id04078/.claude.json)
❯ github · ✔ connected
```



```
Github MCP Server

Status: ✔ connected
Command: npx
Args: -y @modelcontextprotocol/server-github
Config location: /Users/id04078/.claude.json
Capabilities: tools
Tools: 26 tools

❯ 1. View tools
  2. Reconnect
  3. Disable
```



```
Tools for github
26 tools

❯ 1.  create_or_update_file
  2.  search_repositories
  3.  create_repository
  4.  get_file_contents
↓ 5.  push_files
```

Skills in practice

How to create, use, and share them

04

Let's see a Skill in the wild

Example: `/telefonica-slides`

- Bootstraps a Slidev presentation with Telefonica branding
- Accepts a file, URL, or inline text as input
- Generates properly structured slides automatically

Examples:

```
/telefonica-slides ./docs/api.md
/telefonica-slides summarize the contents from ...
/telefonica-slides export pdf
/telefonica-slides create a pdf
```

One skill. Reusable steps, workflow and prompt.

Creating a skill: the recipe

1. Create a `SKILL.md` file
2. Add YAML frontmatter (`name` , `description`)
3. Write the procedure
4. Keep it under ~500 lines
5. Point to reference files (one level deep only)

```
---
name: telefonica-slides
description: Bootstrap a Slidev
  presentation with Mistica theme
---

## Procedure
1. Read the input source
2. Generate slide structure
3. Apply Mistica layout
4. Run slidev dev to validate
```

The **description** is critical: how the agent decides *when* to use it.

/telefonica-slides

```
---
name: telefonica-slides
description: Creates branded presentations using ...
metadata:
  tags: []...
---

## When to use this skill

## Inputs
...

## Procedure
...

## Output format
...

## Examples
...
```

What makes a good Skill

- ✓ **Concise** — only include what the model doesn't already know
- ✓ **Specific description** — what it does AND when to use it
- ✓ **Progressive disclosure** — overview in SKILL.md, details in linked files
- ✓ **Feedback loops** — run → validate → fix → repeat

What makes a bad Skill

- ✗ Don't **over-explain** basics
- ✗ Don't give **5 options** when one default + escape hatch works
- ✗ Don't nest references **more than one level** deep

What’s happening around us

Teams are already building on this

05

What's happening in the wider team

CDO level:

- Agents and MCP sharing point (shared repo)
- Building reusable Skills across teams

Video team:

- Same approach, scoped to video workflows
- Specific MCPs and Skills for their context

We're figuring it out **together**.

Nothing is set in stone

This space is 2 months old in practice

Best practices are being written in real time

Everyone is experimenting, including the people at Level

Telefónica Innovación Digital

Thanks