



Agentes de IA para pesquisa em ciências sociais

uma introdução ao Claude Code

- 01 **Do chat ao agente:** o que muda na prática
- 02 **Claude Code na prática:** instalação, CLAUDE.md e skills
- 03 **Gerenciamento de contexto**
- 04 **Aplicações práticas:** demos ao vivo
- 05 **Para além do curso:** limites, ética e recursos

Do chat ao agente

Você provavelmente já usa IA

- Interface de chat: você digita, o modelo responde
- Cada conversa começa do zero: sem memória de sessões anteriores, sem acesso ao seu ambiente de trabalho
- Interações **transacionais**: úteis para tarefas pontuais, mas sem continuidade

O chat tem limites para pesquisa

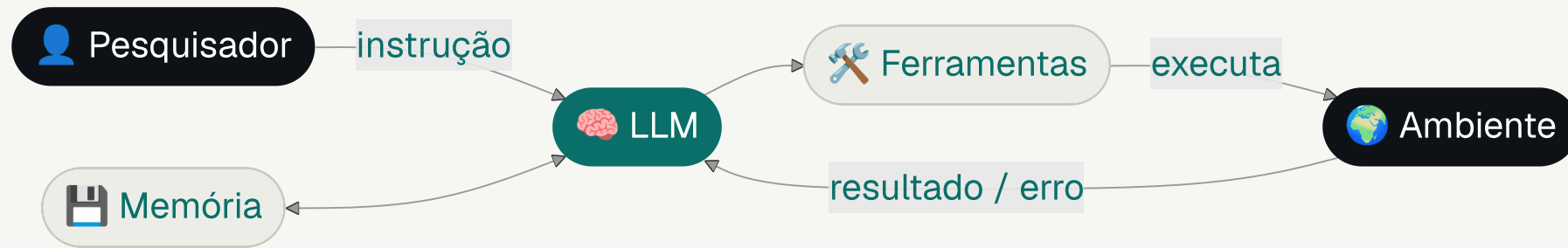
- Sem acesso ao seu ambiente de trabalho (arquivos, código, terminal)
- Sem memória entre sessões
- Cada etapa do ciclo de pesquisa (coleta, análise, documentação, comunicação) exige sua coordenação manual

O que é um agente de IA?

O agente decide o próximo passo, roda o código e corrige sozinho quando dá erro. Você entra só pra revisar.

Chat	Agente
Responde	Age
Uma troca	Múltiplas etapas
Você executa	Ele executa

Anatomia de um agente



 contexto da sessão + CLAUDE.md  terminal · arquivos · APIs · web

Claude Code

- CLI que roda no terminal, ou integrado ao VSCode e ao Positron
- Lê e escreve arquivos, executa código, acessa o `git` e o shell
- É, hoje, o agente de IA mais maduro para uso em pesquisa

Roda o script, encontra o erro, corrige e re-executa. Você nem chega a abrir o terminal.

Outros agentes de IA

Ferramenta	Empresa	Diferencial
Claude Code	Anthropic	CLI + IDE; forte em raciocínio e texto
Codex	OpenAI	Equivalente da OpenAI para terminal
Cursor	Cursor AI	IDE própria centrada em IA
Cline	Open-source	Extensão VSCode; suporta vários modelos

A escolha depende do modelo que você prefere e de como trabalha. **O raciocínio de hoje vale pra qualquer uma delas**

Claude Code na prática

Instalação

Baixar o app em **claude.com/download**, disponível para Mac e Windows

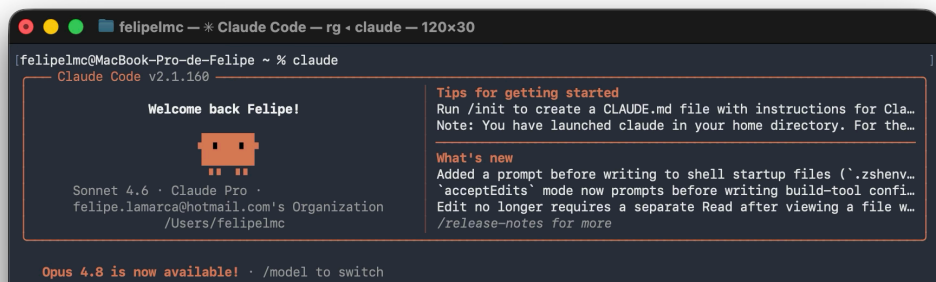
Depois, abrir o Claude e pedir:

“Me ajude a instalar o Claude Code no meu computador”

O próprio Claude guia a instalação: detecta o sistema operacional e resolve as dependências sozinho

Integrações no dia a dia

Terminal



```
felipelmc — * Claude Code — rg - claude — 120x30
[felipelmc@MacBook-Pro-de-Felipe ~ % claude
Claude Code v2.1.160

Welcome back Felipe!

Sonnnet 4.6 · Claude Pro ·
felipe.lamarca@hotmail.com's Organization
/Users/felipelmc

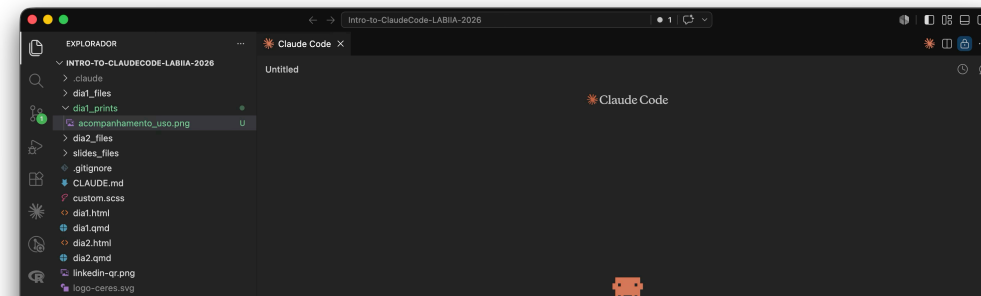
Opus 4.8 is now available! · /model to switch

Tips for getting started
Run /init to create a CLAUDE.md file with instructions for Cla...
Note: You have launched claude in your home directory. For the...

What's new
Added a prompt before writing to shell startup files (.zshenv...
"acceptEdits" mode now prompts before writing build-tool confi...
Edit no longer requires a separate Read after viewing a file w...
/release-notes for more
```

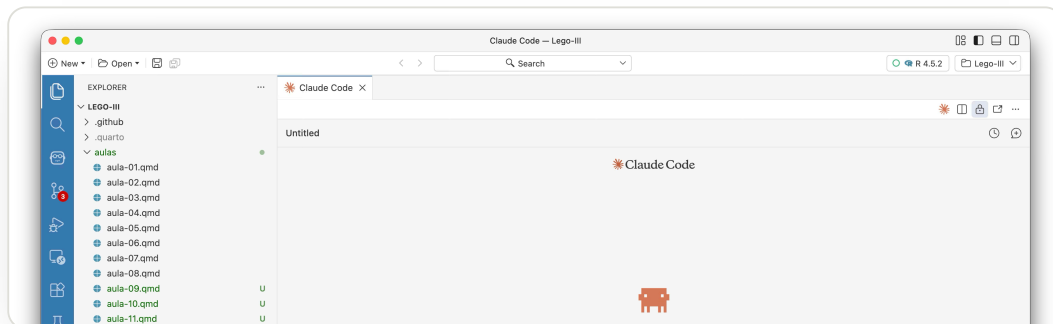
```
cd projeto/ && claude
```

VSCode



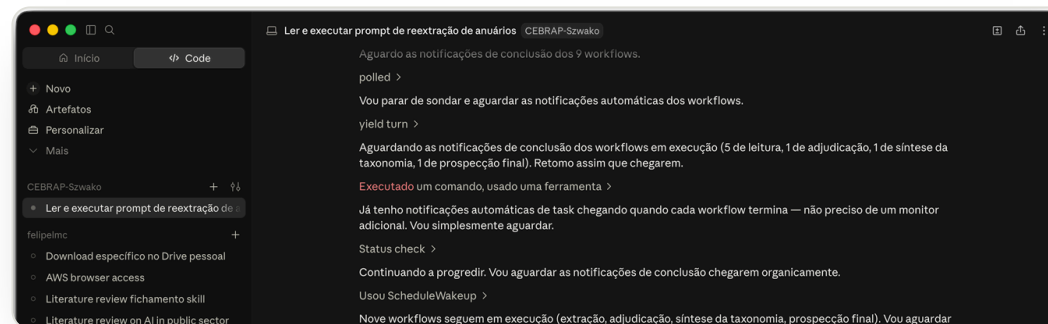
Extensão “Claude Code” na barra lateral

Positron



Para quem usa R: mesmo mecanismo

App desktop



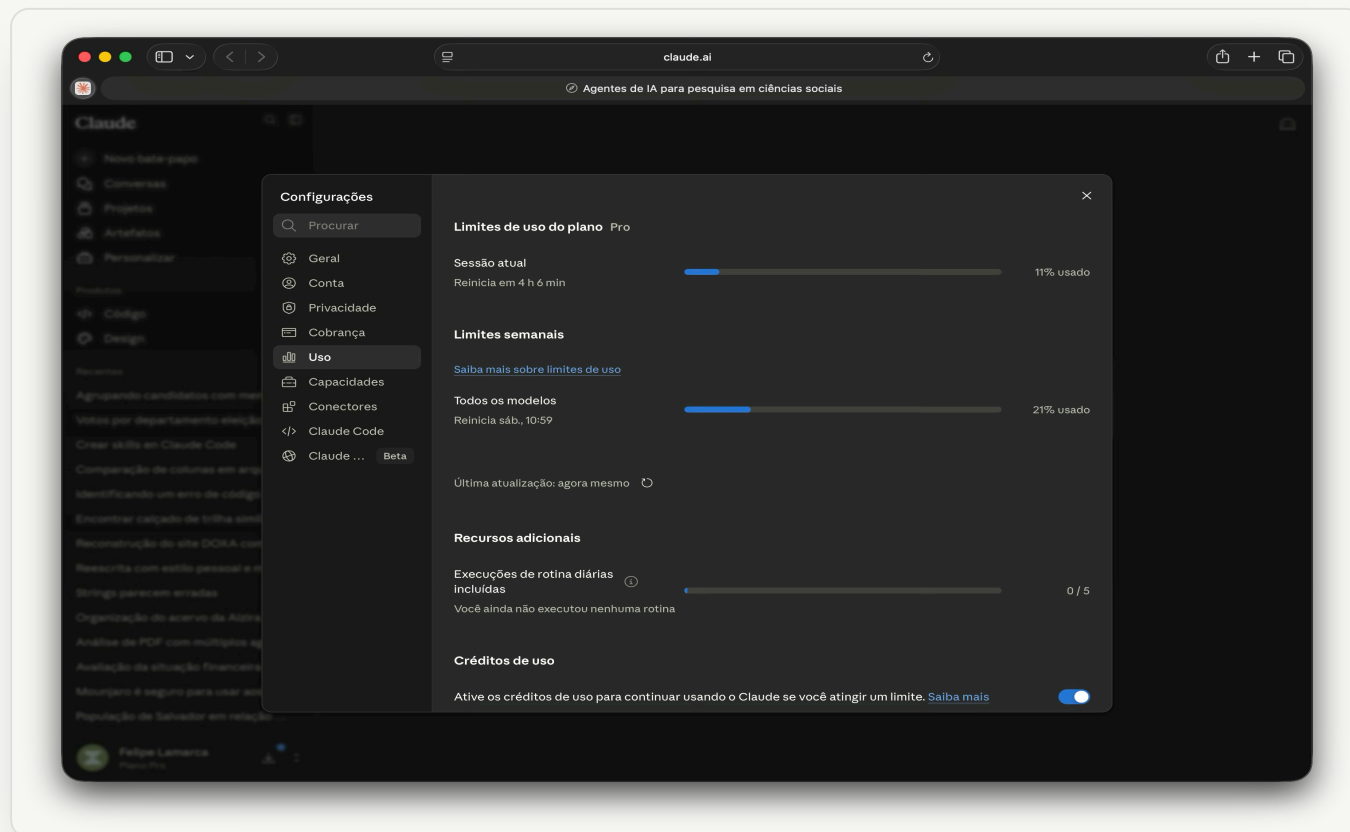
Aba “Code”: sessões organizadas por projeto

Quanto custa? 💰

Plano	Preço	Para quem
Pro	US\$ 20/mês	Uso individual
Max 5×	US\$ 100/mês	Uso intensivo
Max 20×	US\$ 200/mês	Uso muito intensivo
API	Por token	Automação e uso programático
Team / Enterprise	A partir de US\$ 30/usuário	Organizações

O plano Pro tem limite de uso, mas dá pra render bem numa rotina de pesquisa acadêmica

Acompanhando o gasto



No dashboard da Anthropic, em tempo real: uso de tokens e gasto por período, com alerta quando bate o limite

CLAUDE.md: instruções persistentes

Arquivo de instruções persistentes, lido automaticamente pelo agente a cada nova sessão

O que colocar:

- Contexto do projeto e objetivos
- Convenções técnicas (linguagem, estrutura de pastas)
- **Decisões analíticas já tomadas**, e por quê

É a **memória permanente** do agente no projeto: o que sobrevive ao `/clear`

Um CLAUDE.md para pesquisa

```
1 # CLAUDE.md
2
3 ## Contexto
4 Projeto de monitoramento de votações nominais na Câmara dos Deputados.
5 Fonte: API de Dados Abertos da Câmara -- não usar scraping do site.
6
7 ## Convenções
8 - Usar R 4.3+ com tidyverse e ggplot2
9 - Salvar gráficos em `output/figures/`
10 - Cada análise em script separado
11
12 ## Decisões analíticas
13 - Unidade de análise: deputado × votação nominal
14 - Hipótese: alinhamento ao governo varia por bloco partidário
15 - Considerar só a legislatura atual (57ª)
16 - Excluir votações simbólicas (sem registro nominal)
```

A pasta `.claude`

```
projeto/
├── CLAUDE.md          ← instruções permanentes do projeto
└── .claude/
    ├── commands/      ← suas skills customizadas
    │   ├── voice.md
    │   └── summarize.md
    └── settings.json   ← permissões e configurações
```

E na sua máquina, fora de qualquer projeto:

```
~/.claude/
└── commands/          ← skills globais, disponíveis em todo projeto
    └── voice.md
```

Skills: comandos customizados ⚡

`/nome-da-skill` executa um *prompt* reutilizável, definido num arquivo `.md` em `.claude/commands/`

- 1 Você vai receber um texto acadêmico em português.
- 2 Produza um resumo estruturado com:
- 3 1. Pergunta de pesquisa (1 frase)
- 4 2. Método principal (1 frase)
- 5 3. Resultado central (2 frases)
- 6 4. Limitações reconhecidas pelos autores (bullet points)

Para usar: chame `/summarize` e cole o abstract ou o artigo inteiro. É portátil entre projetos: copie a pasta e as skills vão junto

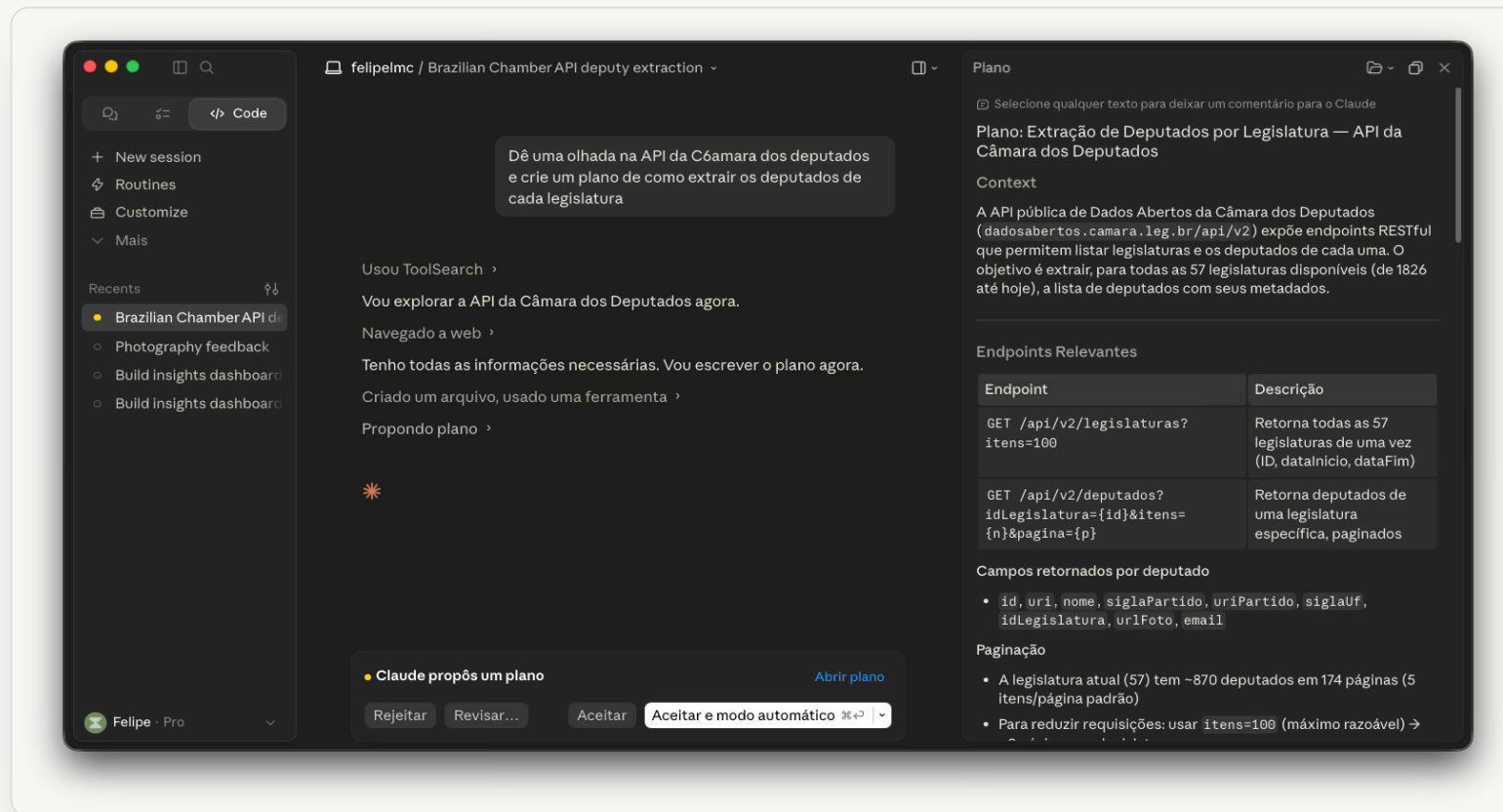
Skills de pesquisa

Skill	O que faz
<code>/voice</code>	Reescreve texto no seu estilo de escrita
<code>/summarize</code>	Resume artigos: pergunta, método, achado, limites
<code>/referee-2</code>	Revisão em instância nova do terminal: “não peça ao mesmo Claude que escreveu o código para revisá-lo”
<code>/blindspot</code>	Identifica análises que você não pensou em fazer

`/referee-2` e `/blindspot` são do workflow de Scott Cunningham
(github.com/scunning1975/MixtapeTools)

Plan mode: revise antes de executar

Antes de qualquer tarefa não-trivial: entrar em **plan mode** com **Shift+Tab**

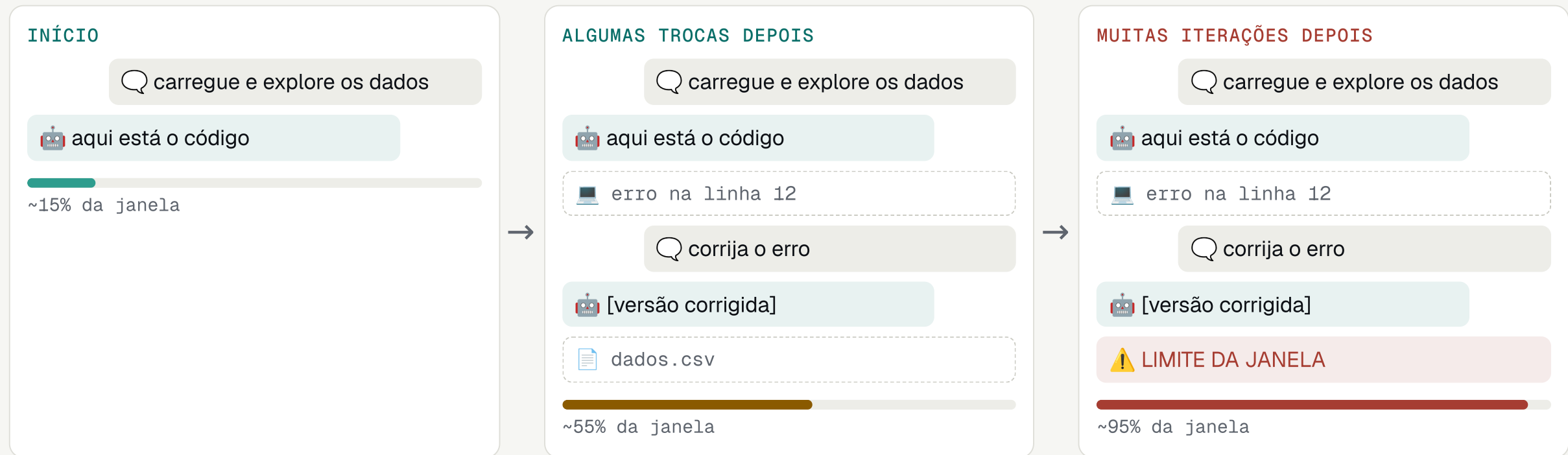


O agente mostra o plano antes de tocar em qualquer arquivo. Você aprova, ajusta ou cancela

Gerenciamento de contexto

A janela de contexto

Tudo que o modelo “vê” ao mesmo tempo cresce a cada iteração



Estratégias de gerenciamento

Estratégia	O que faz
<code>/clear</code>	Reinicia a sessão, preserva o CLAUDE.md
Compaction automática	O Claude resume o contexto sozinho quando necessário
Sessões focadas	Uma tarefa por sessão
CLAUDE.md	Memória que sobrevive ao <code>/clear</code>

Tarefas grandes = múltiplas sessões pequenas e focadas

Aplicações práticas

As demos de hoje



Pesquisa em paralelo

Bancada ambientalista: literatura e votações no Congresso



Site pessoal

Do currículo ao deploy

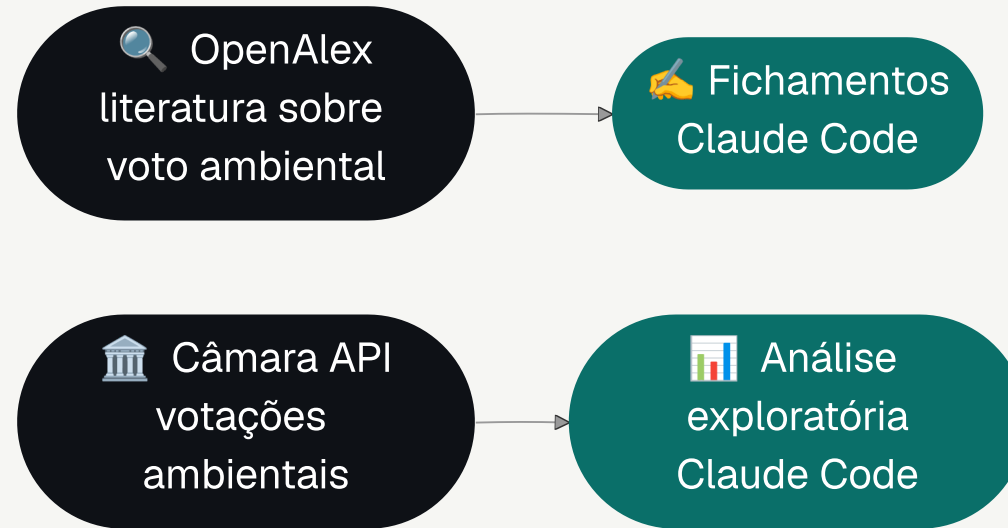
Demo: bancada ambientalista, literatura e dados

O problema

Revisão de literatura e coleta de dados legislativos normalmente rodam em processos separados – primeiro se lê, depois, semanas depois, se coleta

Hoje as duas rodam sobre o mesmo tema: como a bancada ambientalista vota no Congresso

O pipeline



Buscando a literatura

A **OpenAlex** indexa ~250 milhões de trabalhos acadêmicos com API gratuita e sem autenticação

```
1 import requests
2
3 params = {
4     "search": "bancada ambientalista Congresso Nacional votação",
5     "filter": "publication_year:2015-2025",
6     "sort": "cited_by_count:desc",
7     "per_page": 25
8 }
9 resp = requests.get("https://api.openalex.org/works", params=params)
10 works = resp.json()["results"]
```

Claude Code lê os abstracts retornados e gera um fichamento estruturado de cada um

APIs de dados abertos no Brasil

API	O que oferece
Câmara dos Deputados	Deputados, votações, proposições, despesas
Portal da Transparência	Gastos do governo federal, contratos, servidores
IBGE	PNAD, censo, indicadores socioeconômicos
TSE	Resultados eleitorais, candidatos, financiamento

A API da Câmara dos Deputados

Base URL: `dadosabertos.camara.leg.br/api/v2`

```
1 library(httr2)
2 library(purrr)
3
4 deputados <- request("https://dadosabertos.camara.leg.br/api/v2/deputados") |>
5   req_url_query(idLegislatura = 57, ordem = "ASC", ordenarPor = "nome") |>
6   req_perform() |>
7   resp_body_json(simplifyVector = TRUE) |>
8   pluck("dados")
```

Retorna JSON. Com `simplifyVector = TRUE`, o `httr2` já entrega um data frame pronto

Achando as votações ambientais

Proposições têm um código de tema – meio ambiente é `codTema = 48`, e a lista completa está em `/referencias/proposicoes/codTema`

```
1 proposicoes <- request("https://dadosabertos.camara.leg.br/api/v2/proposicoes") |>
2   req_url_query(codTema = 48, itens = 100) |>
3   req_perform() |> resp_body_json(simplifyVector = TRUE) |> pluck("dados")
4
5 # ex.: PL 2159/2021, a Lei do Licenciamento Ambiental (id 257161)
6 votacoes <- request("https://dadosabertos.camara.leg.br/api/v2/proposicoes/257161/votaco
7   req_perform() |> resp_body_json(simplifyVector = TRUE) |> pluck("dados")
```

Cada proposição pode ter várias votações – de emendas, destaques, do texto principal. Escolher qual votação usar é decisão da pesquisa, não do código

Endpoints úteis

Endpoint	Traz
<code>/proposicoes?codTema=48</code>	Proposições sobre meio ambiente
<code>/proposicoes/{id}/votacoes</code>	Votações de uma proposição específica
<code>/votacoes/{id}/votos</code>	Como cada deputado votou
<code>/deputados</code>	Partido, estado, foto

Duas saídas, mesmo tema

- Da literatura: fichamentos estruturados, um por trabalho, salvos em `/fichamentos/`
- Dos dados: um pedido direto – “faça uma análise exploratória dessas votações” – que vira um relatório simples de pesquisa: quem votou o quê, por partido e por estado

Não é redação automática de um paper. O Claude Code ajuda a explorar os dados; o que vira achado é decisão sua

Demo: site pessoal a partir do currículo

Do currículo ao site pessoal

Input

- Currículo em texto simples ou PDF
- Uma instrução ao agente

Output

- Página HTML/CSS completa e responsiva
- Publicável no GitHub Pages sem configuração manual

O mesmo raciocínio vale para páginas de projeto: o gap entre ter um artigo com código e ter uma página sobre ele é menor do que parece

Para além do curso

Problemas parcialmente contornáveis

Problema	Mitigação
Erros e alucinações	Revisão sistemática; agentes paralelos para verificação cruzada
Degradação do contexto	Sessões focadas por tarefa; compactação automática
Deriva de escopo	CLAUDE.md com restrições analíticas explícitas
Reprodutibilidade	Documentar decisões, não apenas o código gerado

O que não delegar ⚠️

- A **interpretação substantiva** dos achados é do pesquisador
- As **decisões de design**: o que medir, como operacionalizar, quais hipóteses testar
- A **validação**: o agente sabe se o código roda; você verifica se o achado faz sentido
- A **responsabilidade autoral**: o pesquisador assina, inclusive pelo que o agente produziu

O Claude Code acelera a parte mecânica da pesquisa. O raciocínio científico continua sendo seu trabalho

O que dizem os periódicos

Internacionais

Padrão dominante: disclosure obrigatório
+ IA não pode ser autora

- [AJPS](#): proíbe usar IA para redigir o manuscrito
- [APSR](#): exige disclosure
- [Journal of Politics](#): proíbe uso na redação
- [Political Analysis](#): disclosure com ferramenta, versão e uso

Brasileiros

- [CNPq – Portaria nº 2.664/2026](#): disclosure obrigatório em todas as fases; pesquisador é integralmente responsável
- [Opinião Pública / Unicamp](#): nota de rodapé com ferramentas, fases e seções

Recursos úteis

Workflows de pesquisadores que valem a visita. Os repositórios são públicos:

Scott Cunningham github.com/scunning1975/MixtapeTools

Manoel Galdino github.com/mgaldino/agents-workflow

Pedro Sant'Anna github.com/pedrohcgsc/claude-code-my-workflow

A pasta `.claude` deles é pública: dá pra copiar, adaptar e construir em cima

Obrigado!

Felipe Lamarca

IESP-UERJ · MAPE · DOXA · NECON

✉ felipe.lamarca@hotmail.com

🌐 felipelamarca.com

📁 github.com/felipelm/Presentations/tree/main/AgentesIA-FMMAPE-2026