



Introdução aos agentes de IA na pesquisa acadêmica

Dia 2

Felipe Lamarca

IESP-UERJ · MAPE · DOXA · NECON

Feito com Claude Code 😊

3 de junho de 2026

- 01 **Recap**
- 02 **Demo:** fichamento de literatura
- 03 **Demo:** análise da Cota Parlamentar (CEAP)
- 04 **Demo:** API da Câmara dos Deputados
- 05 **Limites e ética**

Recap

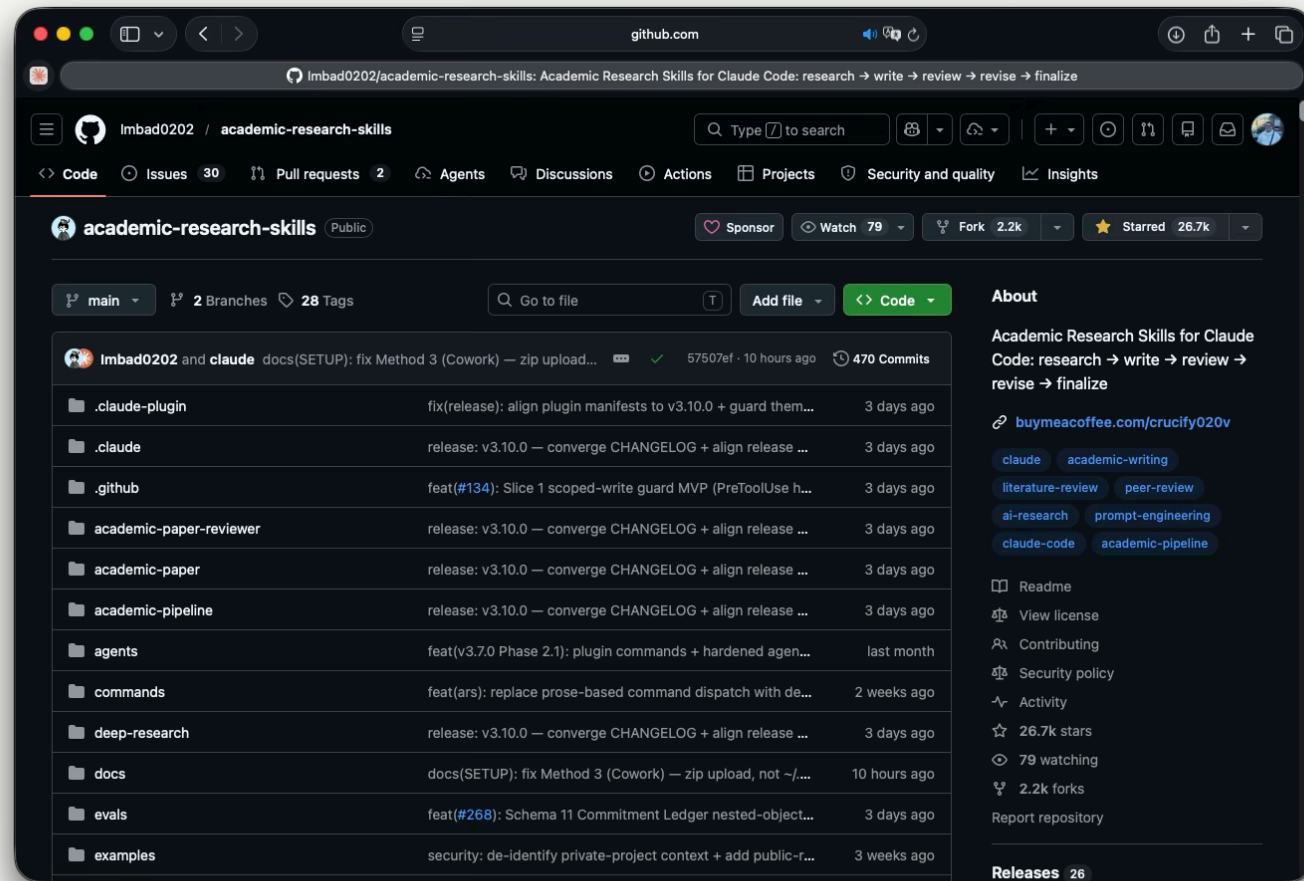
O que vimos no Dia 1

- Claude Code age no seu ambiente: lê arquivos, executa código, acessa o git
- Instalação e integração ao VSCode, Positron e terminal
- **Pasta `.claude`**: CLAUDE.md, skills (`/voice`, `/summarize`, `/referee-2`), settings.json
- Skills encapsulam seu vocabulário — `/nome` chama qualquer uma delas

Hoje vamos usar tudo isso para pesquisa de verdade

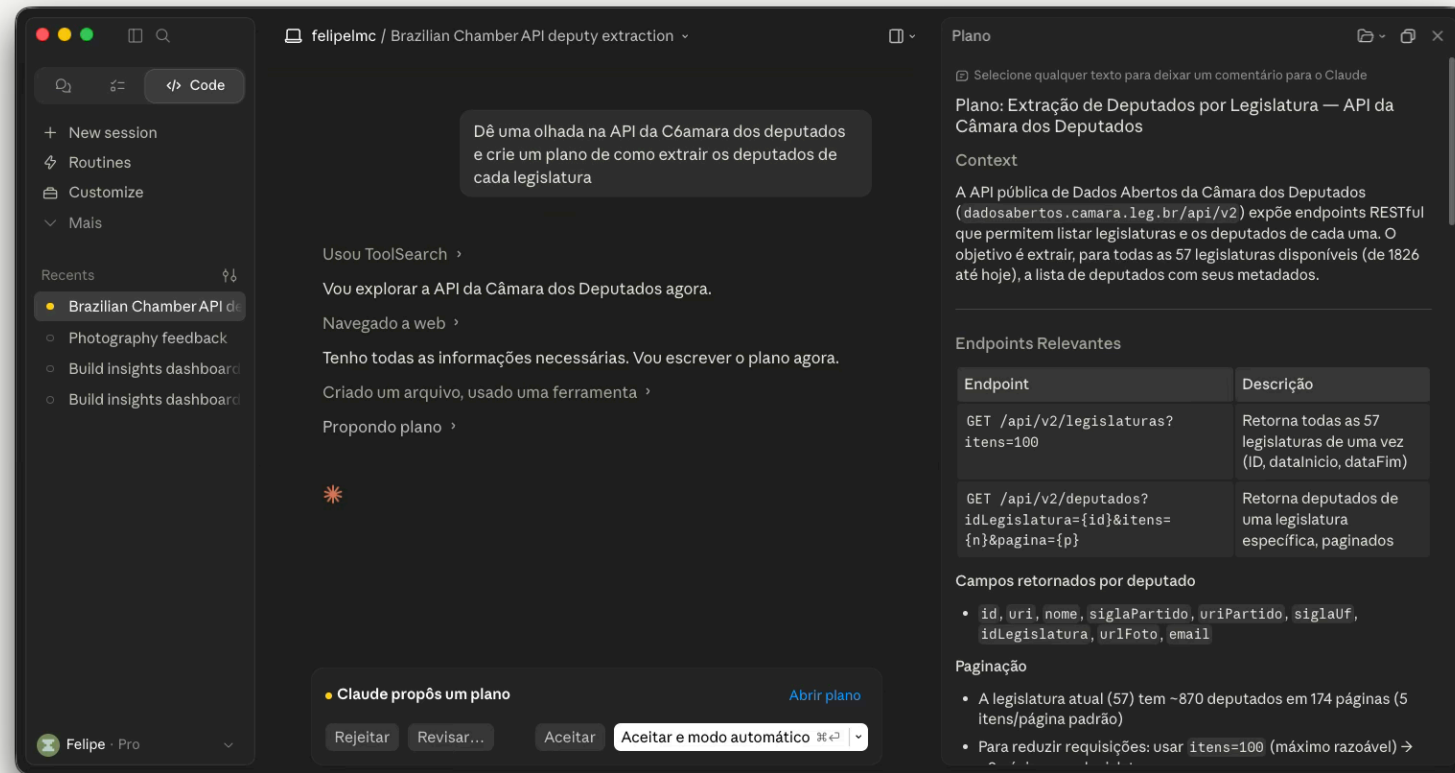
Skills para pesquisa acadêmica

Repositório com skills prontas para download: github.com/Imbad0202/academic-research-skills



Plan mode

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



O agente apresenta o plano — você aprova, ajusta ou cancela antes de qualquer arquivo ser tocado

Demo 1: fichamento de literatura

O problema

Mapeamento de literatura é custoso: horas de leitura para identificar o que é relevante, o que conversa com o quê, o que ainda não foi feito

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

Hoje: IA no setor público brasileiro — `"inteligência artificial" "setor público"` ·
idioma: pt · 2019–2025

O pipeline

Etapa	Ferramenta	Output
Busca	OpenAlex API	lista de papers + metadados
Filtragem	citações, ano, idioma	subset relevante
Download	PDFs em acesso aberto	arquivos locais
Sumarização	Claude Code	<code>autor_ano.md</code> por paper
Geração	metadados OpenAlex	<code>bibliography.bib</code>

Para papers sem PDF disponível: Claude sumariza a partir do abstract

A query

```
1 import requests
2
3 params = {
4     "search": '"inteligência artificial" "setor público"',
5     "filter": "language:pt,publication_year:2019-2025",
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"]
```

Roteiro da demo ao vivo

1. Query OpenAlex → inspecionar os 50 papers retornados
2. Filtrar: manter os 20 mais citados com abstract disponível
3. Claude Code lê cada abstract e gera fichamento estruturado
4. Para papers com PDF: Claude expande o fichamento
5. Salvar em `/fichamentos/` + gerar `bibliography.bib`

O CLAUDE.md do projeto vai estar no repositório – vocês acompanham no próprio computador

Demo 2: análise da Cota Parlamentar

O que é a Cota Parlamentar?

A **CEAP** (Cota para o Exercício da Atividade Parlamentar) é a verba de gabinete dos deputados federais

- Cada deputado recebe até **~R\$ 45 mil/mês** para despesas do mandato
- Categorias: passagens aéreas, alimentação, hospedagem, combustível, consultorias...
- **Todos os gastos são públicos** — Portal da Câmara disponibiliza os dados

Em 2023: mais de **R\$ 220 milhões** em gastos totais

Como baixar os dados

Acesso direto no Portal da Câmara:

<https://dadosabertos.camara.leg.br/swagger/api.html>

Ou baixar o CSV diretamente:

```
1 import pandas as pd
2
3 url = "https://www.camara.leg.br/cotas/Ano-2023.csv.zip"
4 df = pd.read_csv(url, sep=";", encoding="latin-1", compression="zip")
5 df.head()
```

Roteiro da demo ao vivo

Com o Claude Code + dataset CEAP 2023:

1. Carregar e inspecionar os dados
2. Análise exploratória: distribuição de gastos, categorias mais comuns
3. Gastos por partido e por estado — padrões regionais
4. Identificar os maiores gastadores e as categorias mais polêmicas
5. Visualizações prontas para comunicar os achados

Vocês acompanham no próprio computador — o CLAUDE.md do projeto vai estar no repositório

Demo 3: API da Câmara dos Deputados

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

API da Câmara dos Deputados

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

```
1 import requests
2
3 # Listar deputados da legislatura atual
4 resp = requests.get(
5     "https://dadosabertos.camara.leg.br/api/v2/deputados",
6     params={"idLegislatura": 57, "ordem": "ASC", "ordenarPor": "nome"}
7 )
8 deputados = resp.json()["dados"]
```

Retorna JSON bem estruturado — fácil de transformar em DataFrame com `pd.json_normalize()`

Endpoints úteis

Endpoint	O que traz
<code>/deputados</code>	Lista com partido, estado, foto
<code>/deputados/{id}/despesas</code>	Gastos individuais (mesmo que o CEAP)
<code>/votacoes</code>	Votações nominais na Câmara
<code>/votacoes/{id}/votos</code>	Como cada deputado votou
<code>/proposicoes</code>	PL, PEC, MP e outros

Roteiro da demo ao vivo

Conectando a API com o que já analisamos no CEAP:

1. Buscar a lista completa de deputados via API
2. Enriquecer o dataset CEAP com informações de partido e estado
3. Cruzar padrão de gastos com padrão de votação
4. Gerar visualização integrada

Limites e ética

Problemas parcialmente contornáveis

Problema	Mitigação
Erros e alucinações	Revisão sistemática do output; rodar o código antes de confiar
Degradação do contexto	Sessões focadas por tarefa; CLAUDE.md com decisões-chave
Deriva de escopo	Instruções explícitas; revisão incremental
Reprodutibilidade	Documentar decisões — não só 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 é um acelerador, não um substituto do raciocínio científico

Reprodutibilidade na pesquisa com IA

O código gerado pelo agente roda — mas o código sozinho não documenta **por que** você fez cada escolha

Boas práticas:

- Registrar no CLAUDE.md as decisões analíticas e suas justificativas
- Comentar no código as escolhas não óbvias
- Tratar o repositório como um diário de pesquisa, não só como um arquivo de scripts

O que dizem os periódicos (internacionais)

Padrão dominante: **disclosure obrigatório** + IA não pode ser autora + responsabilidade é do pesquisador

- [AJPS](#) — proíbe uso para redigir o manuscrito ou seções substanciais
- [APSR](#) — disclosure obrigatório; alinhado à política Cambridge
- [Journal of Politics](#) — proíbe uso na redação; revisores não podem usar IA para avaliar
- [Political Analysis](#) — disclosure com ferramenta, versão, data e descrição do uso

O que dizem os periódicos (brasileiros)

- [CNPq — Portaria nº 2.664/2026](#) — disclosure obrigatório em todas as fases; IA não pode ser autora; pesquisador é integralmente responsável pelo conteúdo final
- [Opinião Pública / Unicamp](#) — nota de rodapé especificando ferramentas, fases e seções

Para além do workshop

Recursos úteis

Documentação oficial: claude.ai/code

Seções recomendadas:

- *How Claude Code uses memory* — entender o CLAUDE.md em profundidade
- *Claude Code hooks* — automatizar ações recorrentes
- *Model Context Protocol (MCP)* — conectar a APIs e fontes externas

Obrigado!

Felipe Lamarca

IESP-UERJ · MAPE · DOXA · NECON

✉ felipe.lamarca@hotmail.com

🌐 felipelamarca.com

📁 github.com/felipelmc/Presentations/tree/main/Intro-to-ClaudeCode-LABIIA-2026



🔗 [LinkedIn](#)