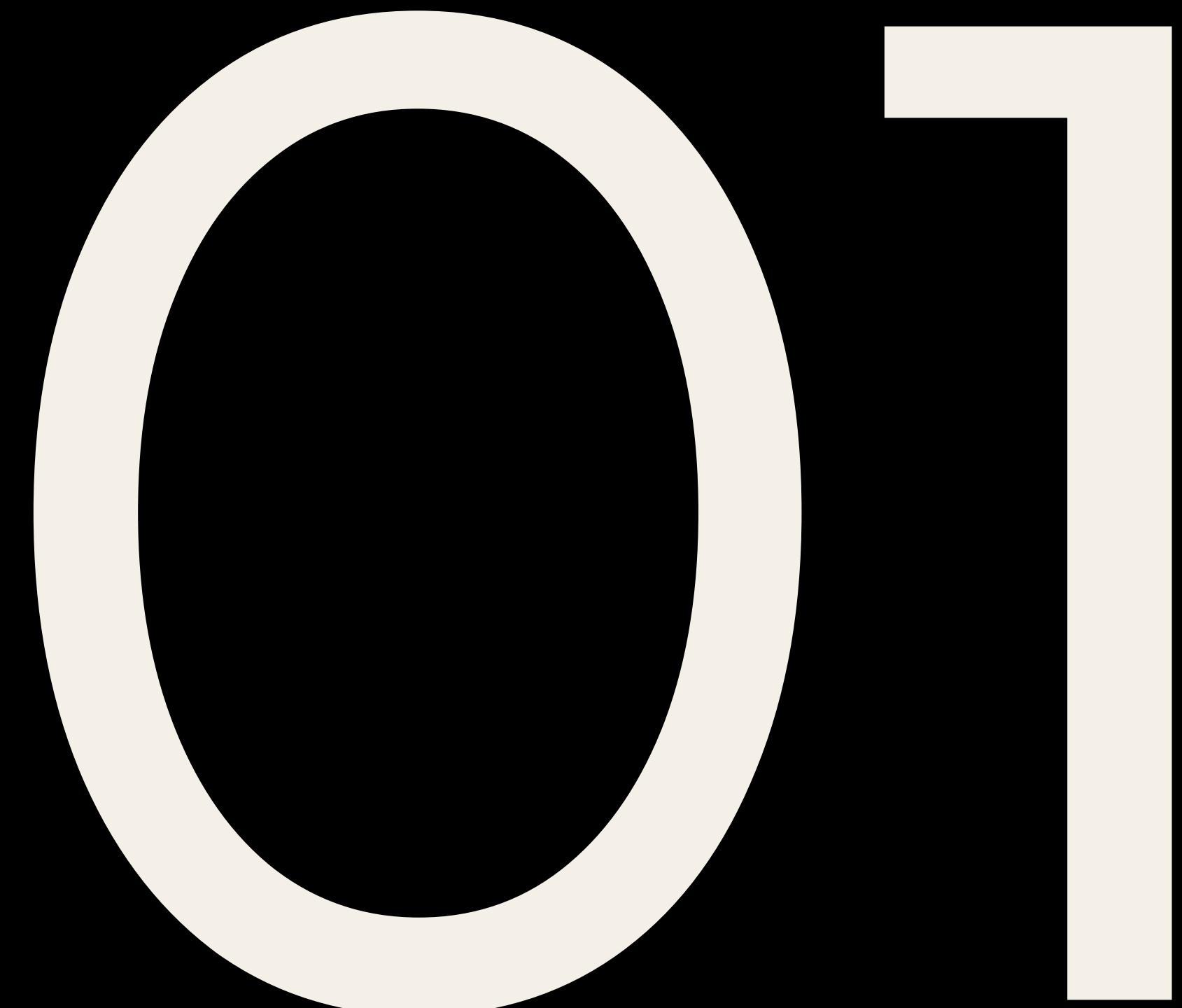


**Desenvolvimento  
Acelerado com  
Agentes de IA**

# Agenda:

1. Abertura
2. Contexto
3. IA, LLM e Assistentes e Agentes
4. Ferramentas e Stack
5. Boas Práticas e Limites
6. MCP
7. Desenvolvimento Acelerado na Prática
8. Proposta de Uso

# Abertura



A IA está entre nós, mas  
produzir código ≠  
desenvolver uma solução  
de qualidade.

# O que está além de produção de código?

## ● ARQUITETURA DE SOFTWARE:

Como o sistema é construído: um bloco gigante (monolito), várias partes pequenas (microserviços) ou sem se preocupar com servidores (serverless).

## ● DESIGN PATTERNS

Receitas prontas para resolver problemas de código comuns.

## ● PRINCÍPIOS SOLID

Cinco regras para escrever um código limpo e fácil de mexer.

## ● CLEAN ARCHITECTURE

Organiza o código para proteger as regras do seu negócio de mudanças tecnológicas.

## ● DDD (DOMAIN- DRIVEN DESIGN)

Desenvolve software focado na linguagem e nos processos do negócio.

## ● VISÃO DE PRODUTO

O objetivo final do produto, o que ele faz e para quem ele é.

# 3 insights para refletir

- **SE ESPECIALIZAR**

Nunca foi tão necessário ser realmente bom no que você faz para que a IA seja uma ferramenta valiosa e não um medo de ser substituído por ela.
- **USAR IA COMO FERRAMENTA**

Integrar a IA como ferramenta de produtividade é uma necessidade para manter a nossa entrega de valor como desenvolvedores.
- **CODAR NÃO É TUDO**

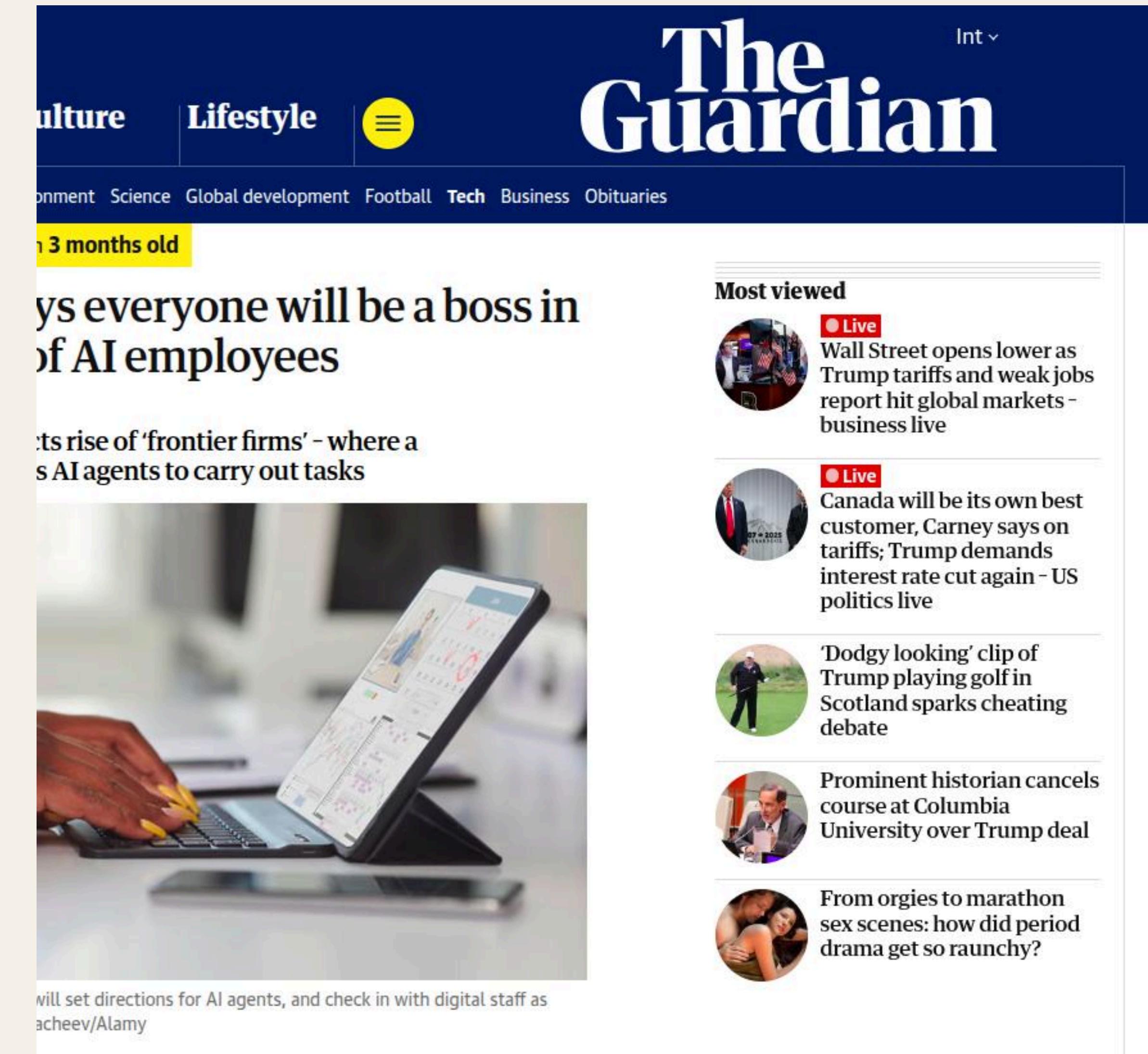
Codar não é mais diferencial competitivo porque código é parte da estratégia.

# Contexto



Pegando como referência o post no blog do The Guardian que cita o relatório anual da Microsoft de 2025.

# “A integração da IA no desenvolvimento de software pode ser dividida em 3 etapas:



The screenshot shows a news article from The Guardian's technology section. The header reads "Microsoft says everyone will be a boss in the future of AI employees". Below the headline, there is a sub-headline: "It's rise of 'frontier firms' - where a boss will set directions for AI agents, and check in with digital staff as needed". A small image shows a person's hands typing on a keyboard next to a smartphone displaying a calendar application. The article is dated "3 months old". To the right, there is a sidebar titled "Most viewed" with several live news feeds.

Int ▾

Culture | Lifestyle | ☰

Environment Science Global development Football Tech Business Obituaries

3 months old

## Microsoft says everyone will be a boss in the future of AI employees

It's rise of 'frontier firms' - where a boss will set directions for AI agents, and check in with digital staff as needed



will set directions for AI agents, and check in with digital staff as needed

Live

Wall Street opens lower as Trump tariffs and weak jobs report hit global markets - business live

Live

Canada will be its own best customer, Carney says on tariffs; Trump demands interest rate cut again - US politics live

Dodgy looking' clip of Trump playing golf in Scotland sparks cheating debate

Prominent historian cancels course at Columbia University over Trump deal

From orgies to marathon sex scenes: how did period drama get so raunchy?

(<https://www.theguardian.com/technology/2025/apr/25/microsoft-says-everyone-will-be-a-boss-in-the-future-of-ai-employees>)

## Fase 1: humanos com assistentes

- Ferramentas reativas (ex: ChatGPT).
- Dependem de comandos humanos.
- Não tomam iniciativa.
- Ex.: Pedir algo ao ChatGPT, copiar o código e colar na database.

## Fase 2: equipes humano-agente

- Agentes como "colegas digitais".
- Executam tarefas específicas com autonomia, sob supervisão.
- Mantêm contexto entre interações.
- Ex.: Agente que desenvolve funcionalidades a partir de especificações de tarefas, escreve o código-fonte, executa testes e submete para revisão humana.

## Fase 3: orquestrado por humanos, operado por agentes

- Humanos definem a estratégia.
- Agentes operam processos de forma contínua.
- Trabalham em rede (sistemas multi-agentes).
- Ex.: Um time de agentes gerencia o ciclo completo de um pedido de cliente: desde o recebimento do briefing até a entrega do produto e acompanhamento de feedback.

IA, LLM,  
assistentes  
e agentes



## IA TRADICIONAL

**Foca em entender, classificar, prever ou tomar decisões com base em dados existentes.**

**Tarefas:**

- Recomendação de vídeos (YouTube, Netflix)
- Reconhecimento facial
- Detecção de fraude
- Previsão de demanda
- Tradução automática
- Análise de sentimentos
- Classificação de e-mails (spam ou não)

## IA GENERATIVA

**Foca em criar um novo conteúdo a partir de um input e de dados previamente treinados.**

- **Tarefas:** Gerar novos conteúdos, como texto, imagem, código ou vídeo.
- **Exemplos:**
  - ChatGPT (texto)
  - GitHub Copilot (código)
  - DALL·E, Midjourney (imagem)
  - Sora (vídeo)
  - Geração de música com IA

# Alguns tipos de IA Tradicional

## ● RECOMENDAÇÃO PREDITIVA

- Recomenda conteúdos com base no comportamento anterior do usuário.
- Exemplos: YouTube, Netflix, Spotify.

## ● RECONHECIMENTO FACIAL

- Identifica rostos em imagens ou vídeos.
- Exemplos: Desbloqueio facial do celular, segurança em aeroportos.

## ● VISÃO COMPUTACIONAL

- Entende e interpreta o conteúdo de imagens e vídeos além de rostos.
- Exemplos: Detecção de objetos, leitura de placas de carro.

## ● TOMADA DE DECISÃO AUTÔNOMA

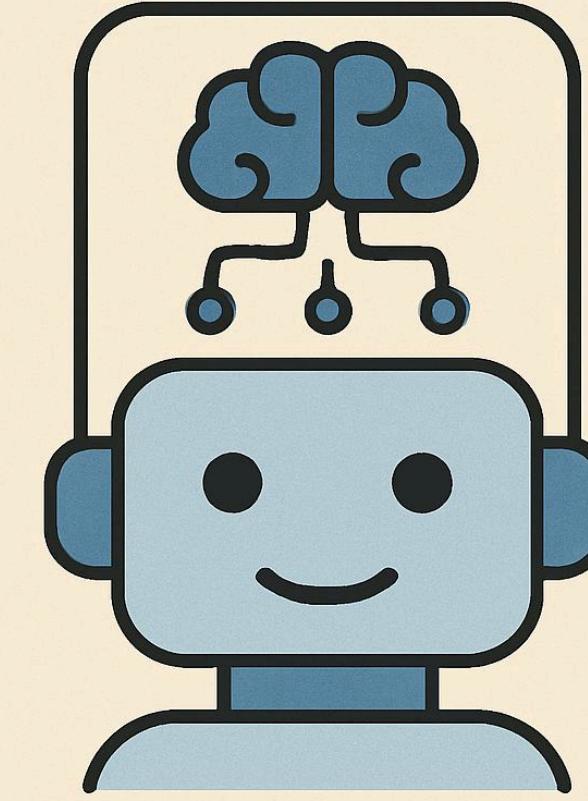
- Toma decisões e executa ações sem intervenção humana.
- Exemplos: Carros autônomos, robôs, IA que joga jogos sozinha.

## ● PREVISÃO

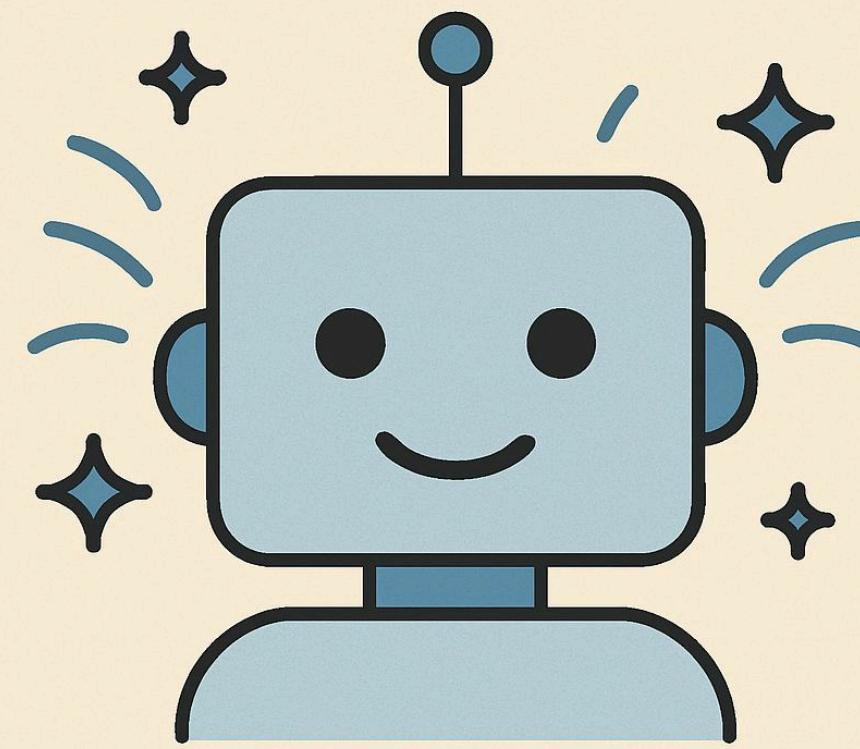
- Faz previsões numéricas baseadas em dados históricos.
- Exemplos: Previsão de vendas, previsão do tempo, demanda de produtos.

## ● DETECÇÃO DE ANOMALIAS

- Detecta padrões incomuns ou comportamentos estranhos.
- Exemplos: Sistemas antifraude, manutenção preditiva em máquinas.



**IA TRADICIONAL**



**IA GENERATIVA**

Aluno em uma prova de  
múltipla escolha.

Responde o que já foi  
ensinado

Aluno em uma prova  
dissertativa.

Responde com base no  
que foi ensinado  
construindo algo novo.

# CÉREBRO DA IA

As IAs, assim como os seres humanos, precisam de um "cérebro" para funcionar.

Assim sendo, o cérebro das iA **tradicionais** são os **modelos clássicos de machine learning**, que tomam decisões com base em padrões extraídos de dados.

## Cérebros da IA Tradicional:

- **Árvore de Decisão:** toma decisões com base em perguntas sucessivas (tipo um jogo de 20 perguntas).
- **Regressão Linear/Logística:** prevê valores ou categorias com base em variáveis numéricas.
- **SVM (Máquina de Vetores de Suporte):** separa dados em categorias usando fronteiras matemáticas.
- **Redes Neurais Clássicas (MLPs):** detectam padrões em dados numéricos, imagens ou texto simples.
- **Modelos baseados em regras:** seguem instruções explícitas definidas por humanos (como “se A, então B”).

# LLM

Já no caso da **IA generativa**, o cérebro são os **Modelos de Linguagem de Grande Escala (LLMs)**, como o GPT (da OpenAI), que não apenas entendem linguagem natural, mas também são capazes de **gerar novos conteúdos**.

## Se o ChatGPT é uma IA Generativa quem é o seu LLM?

IA Generativa (assistente)	LLM utilizado como "cérebro" principal	Desenvolvido por	Open Source?
ChatGPT	GPT-4 (ou GPT-3.5, ou GPT-4o)	OpenAI	✗ Não
Gemini	Gemini 2.5 Flash	Google DeepMind	✗ Não
Claude	Claude 3 (Haiku, Sonnet, Opus)	Anthropic	✗ Não
Grok	Grok-1 (modelo proprietário)	xAI (empresa de Elon Musk)	✗ Não
Copilot (Microsoft)	GPT-4 (via OpenAI API)	Microsoft + OpenAI	✗ Não
LLaMA Chat (Meta)	LLaMA 3	Meta (Facebook)	✗ Não
Mistral (Le Chat)	Mixtral 8x7B, Mistral 7B	Mistral AI	✓ Sim
DeepSeek Chat	DeepSeek-V2	DeepSeek AI	✗ Não

## ASSISTENTE DE IA

- Uma aplicação que usa um **LLM como seu núcleo** para interagir com um usuário.
- É uma **ferramenta reativa**, espera uma solicitação, processa com o LLM e gera um conteúdo como resposta.
- O ChatGPT é um assistente. **Ele não age sem sua instrução direta.**

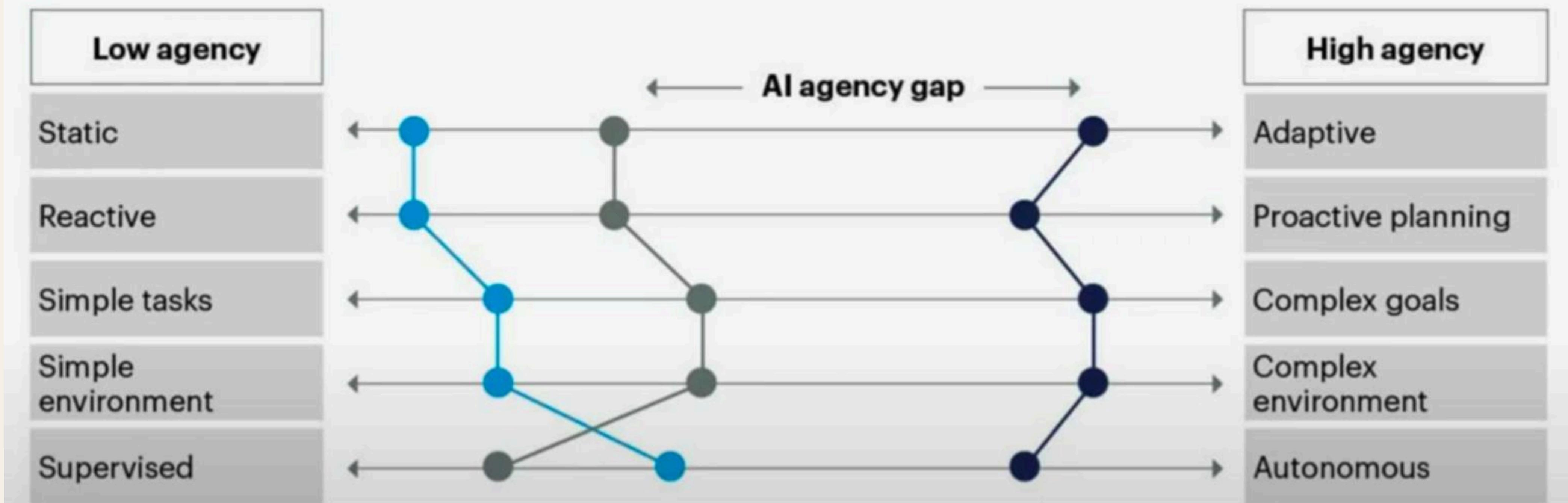
## AGENTE DE IA

- Diferente dos assistentes, os agentes são aplicações que não só respondem, mas **executam tarefas específicas de forma autônoma**, sempre sob alguma supervisão humana. Eles podem pegar um objetivo, planejar passos e agir para cumprir essa tarefa, sem precisar que você dê instruções o tempo todo.

Ser agêntico = ter  
capacidade de tomar  
decisões e executar  
ações deliberadas para  
alcançar um objetivo.

# Mind the AI Agency Gap

● Human agency   ● Deterministic chatbots   ● LLM-based assistants



Source: Gartner

© 2024 Gartner, Inc. and/or its affiliates. All rights reserved. 3176213

**Gartner**

# NÍVEL AGÊNTICO DE CADA ENTIDADE

No contexto de agentes de IA, ser **agêntico** (ou "possuir agência", do inglês *agentic*) significa ter a capacidade de agir de forma autônoma com base em objetivos, informações e contexto



← →  
POUCO MUITO

# Ferramentas e Stack



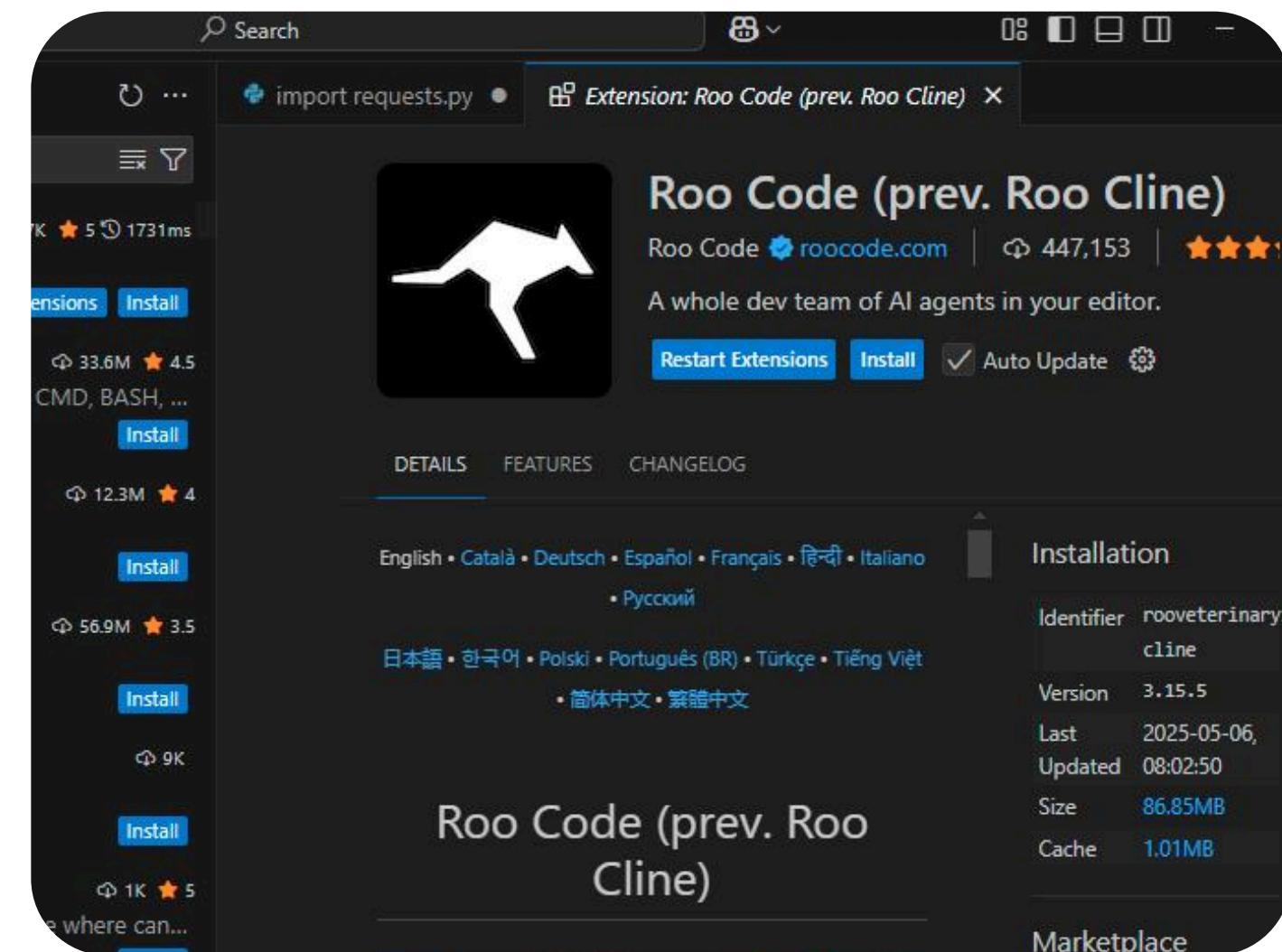
## IDE com Agentes/Chat:

- Cursor
- Windsurf
- VSCode



## Extensões para IDE:

- Cline
- Kilo Code
- Roo Code



## Agentes CLI:

- Claude code
- Gemini CLI
- Codex



# Boas Práticas e Limites



## REGRAS

As regras definidas em **.cursor/rules/\*.mdc** servem como guias para o modelo de IA seguir ao gerar ou revisar código. Elas funcionam como **pontos de partida**, oferecendo diretrizes sobre arquitetura, estilo, práticas recomendadas, workflows, entre outros.

## DOCUMENTOS

Os arquivos em **/docs/\*.mdc** servem como **documentação viva** do sistema. Incluem decisões de arquitetura, stack, fluxos, autenticação, endpoints, ambientes, testes e outros aspectos essenciais do projeto.

# Sugestão de Documentos

## ● **adrs/arquivo.mdc**

- O que foi decidido?
- Por que foi decidido isso?
- Outras opções consideradas
- Consequências da decisão

## ● **overview.mdc**

- Explica o propósito do projeto, objetivos, público-alvo e escopo geral.

## ● **architecture.mdc**

- Descreve a arquitetura geral (ex: monolito, microsserviços), diagramas, camadas, responsabilidades e decisões de alto nível.

## ● **deployment.mdc**

- Explica como fazer deploy do sistema (manualmente ou via CI), ferramentas utilizadas (ex: Vercel, Railway, Docker, GitHub Actions).

## ● **auth.mdc**

- Explica como o sistema autentica usuários (ex: JWT, OAuth, Supabase Auth), níveis de permissão, e fluxo de login/logout.

## ● **endpoints.mdc**

- Lista e documenta todos os endpoints expostos (REST, GraphQL etc), com verbos HTTP, parâmetros, exemplos de request/response.

## ● **environment.mdc**

- Documenta todas as variáveis necessárias em .env, seus propósitos e exemplos (sem expor dados sensíveis).

# Sugestão de Documentos

## ● **glossary.mdc**

- Lista termos técnicos, siglas e conceitos usados no projeto com suas definições para facilitar entendimento por novos devs.

## ● **workflow.mdc**

- Como abrir PRs
- Branching strategy (main/dev, trunk-based etc)
- Revisão de código
- Versionamento

## ● **state.mdc**

- Exemplo: state.local.md
- Adicionar ao .gitignore
- Dicas de como personalizar sem afetar o repositório principal

## ● **guidelines.mdc**

- Nomenclatura de arquivos/variáveis
- Estrutura de pastas
- Convenções de código (ex: async/await obrigatório, ESLint, Prettier)

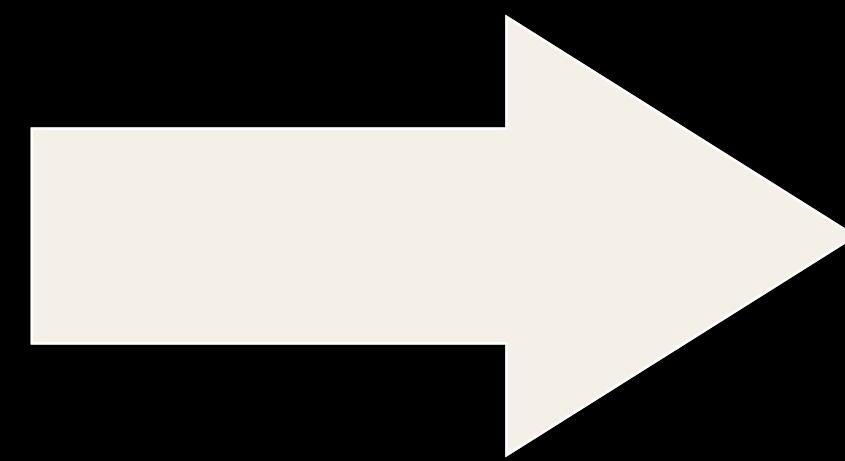
## ● **stack.mdc**

- Libs / Frameworks utilizados
- Por que foram escolhidos
- Links para documentação oficial
- Outras opções consideradas (se houver)

## ● **testing.mdc**

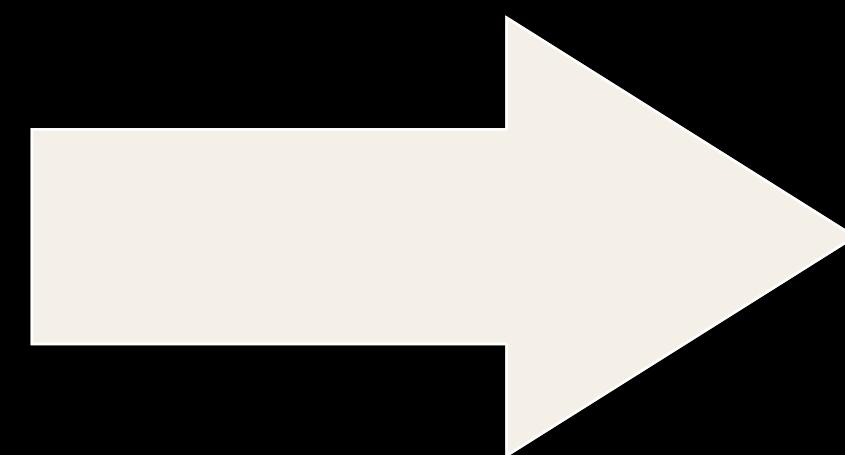
- Descreve a estratégia de testes (unitários, integração, e2e), bibliotecas utilizadas, padrões para nomeação e estrutura dos testes.

Regras de codificação  
e padrões obrigatórios



.cursor/rules

Documentação  
explicativa, contextual  
e de referência



/docs

# MCP



# MCP (Model Context Protocol)

O agente tem uma base “core” de compreensão que é o LLM. É o que fornece a capacidade fundamental do agente.

O MCP é basicamente uma forma do agente se conectar com o mundo exterior, com o que está além do LLM.

As IDE's com agente são clientes para os servidores MCP.

## MCPs interessantes:

- Supabase
  - Doc: <https://supabase.com/docs/guides/getting-started/mcp>
- Context 7
  - Site: <https://context7.com/>
  - Doc: <https://github.com/upstash/context7>

# Exemplos de prompts para Context 7

Use context 7 e me diga qual é a melhor forma de configurar headers com axios.

Verifique se a forma como estou lidando com headers no Axios segue boas práticas de segurança e refatore se necessário, respeitando as regras da pasta `@.cursor/rules`. Use context 7.

Analise a estrutura atual do projeto, incluindo arquivos de configuração do Axios, regras em `@.cursor/rules`, e documentação em `@/docs`. Quero criar uma forma segura, reutilizável e consistente de configurar os headers do Axios (ex: `Authorization`, `content-type` etc). Leve em conta boas práticas de segurança, testes, desacoplamento e facilidade de manutenção. Revise o que está implementado e proponha melhorias. Use context 7.

# Desenvolvimento Acelerado na Prática



# Repositório para criação do Boilerplate:

[https://github.com/igorspestana/  
webinar-desenvolvimento-acelerado-ia](https://github.com/igorspestana/webinar-desenvolvimento-acelerado-ia)

# Propostas de Uso



# Uso de agentes no contexto de equipes de desenvolvimento:

## 1. Entendimento de Código Legado

- Explicar **um módulo ou função complexa** em linguagem simples.
- Criar **resumo de como partes do sistema se conectam** (ex.: fluxo de login, pipeline de dados).
- **Mapear dependências** de um arquivo ou pacote.
- **Gerar fluxogramas** de execução a partir do código.
- **Localizar funções chave** relacionadas a um comportamento específico (ex.: “Onde é feita a validação de senha?”).

## 2. Criação e Manutenção de Documentação

- Gerar **docstrings padronizadas** para funções, classes e APIs.
- Criar **documentação técnica** para módulos inteiros, com exemplos de uso.
- Produzir **diagramas UML** ou de arquitetura a partir do código.
- Resumir PRs ou commits para **atualizar changelogs** automaticamente.
- Manter documentação sincronizada quando o código muda.

## 3. Suporte à Leitura e Navegação

- **Responder perguntas sobre o código** como se fosse um “guia turístico” do repositório.
- Gerar **mapa de arquivos importantes** para determinado recurso.
- Criar **índice temático** (por funcionalidade) do repositório.
- Sugerir pontos de entrada para investigar um bug.

## 4. Aprendizado Acelerado de Projetos

- Criar **roteiro de onboarding** técnico para novos devs.
- Explicar **padrões e convenções usadas no projeto**.
- Sugerir **exemplos de uso** para funções utilitárias.

Quanto mais documentação e dados  
menos chance da IA errar. Mas a code  
review é sua e o código é seu.

Obrigado!