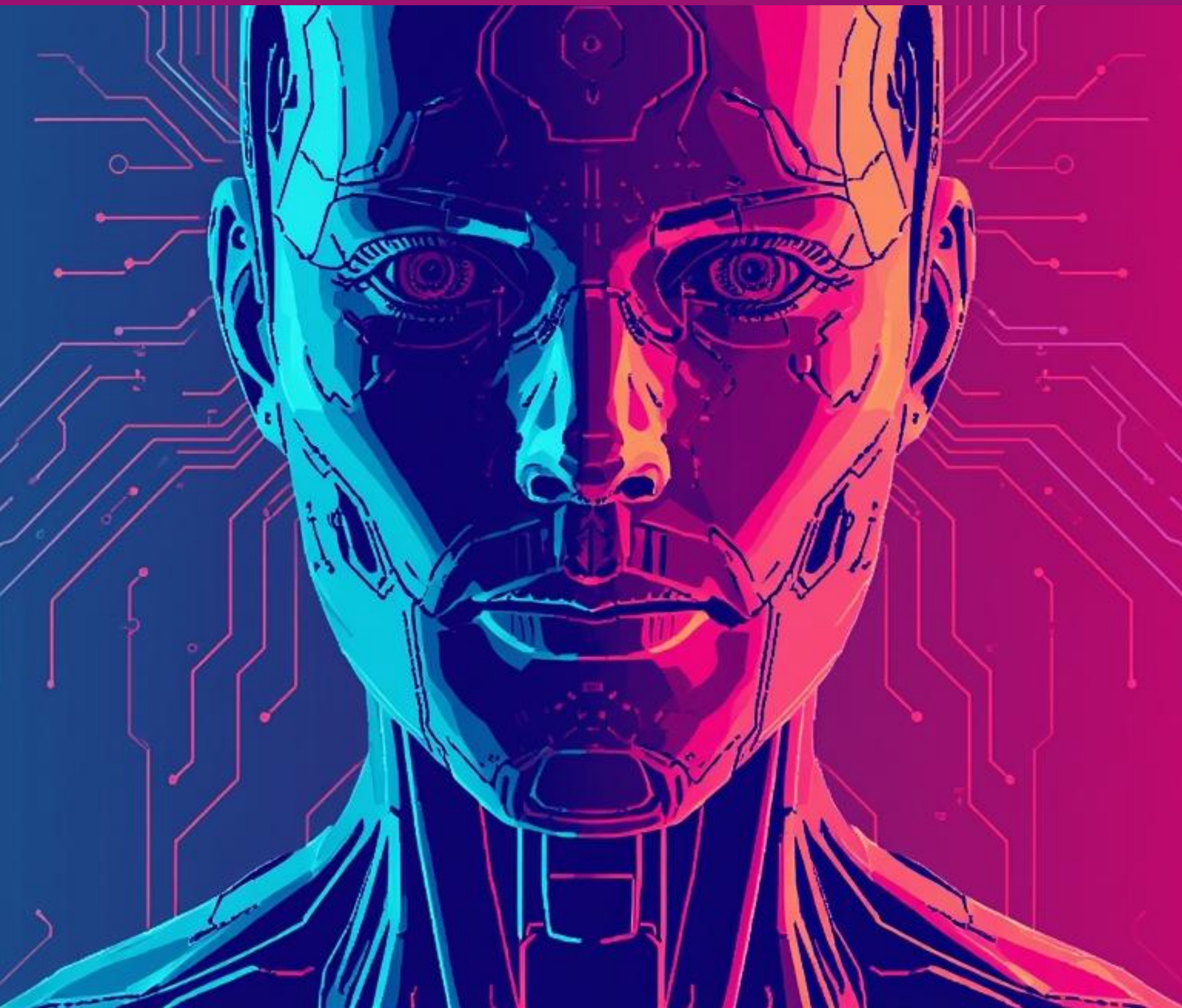


# Matriz de Comando

Prompts para a Conquista da Máquina



**Rafael Perroni**

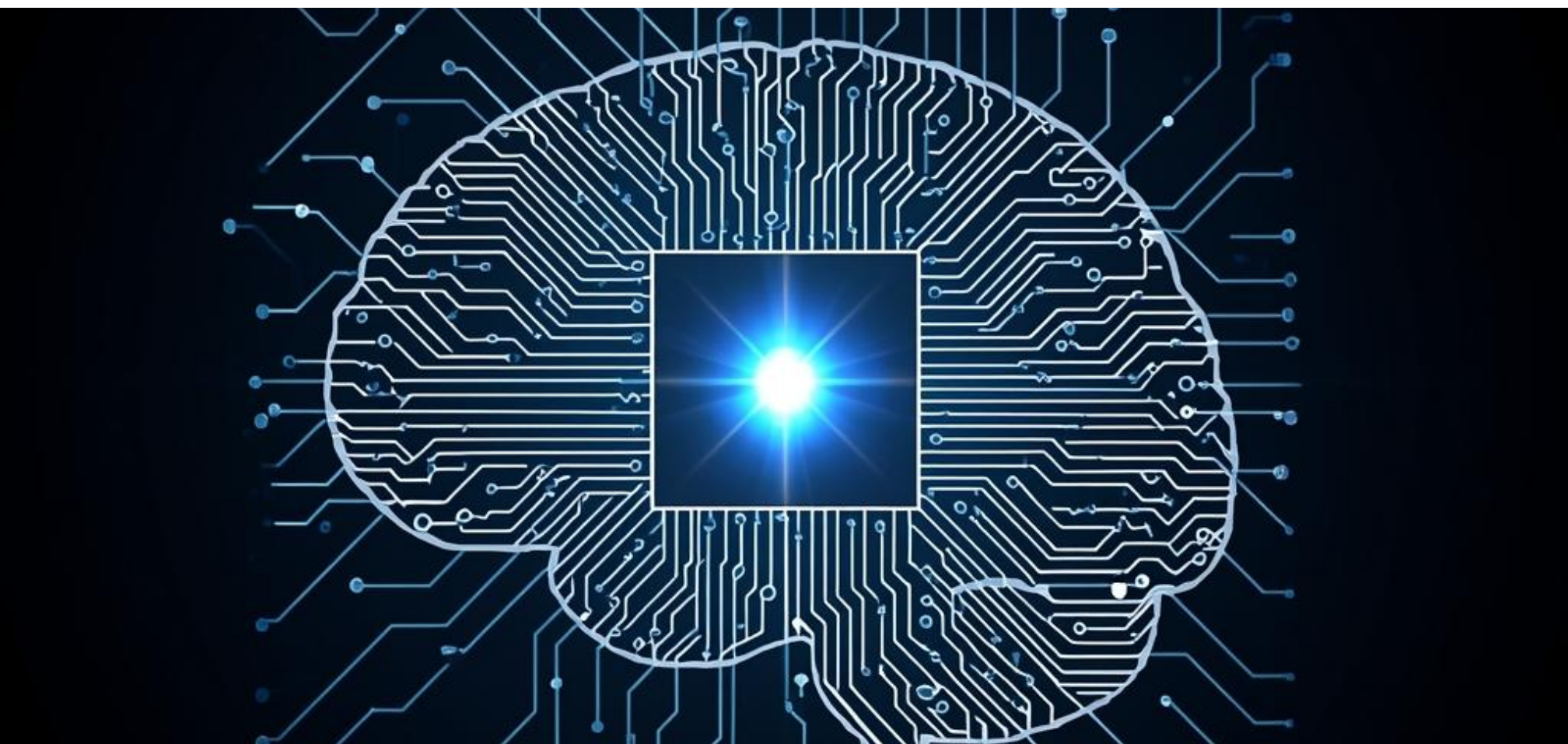


# O Código que Comanda

## Técnicas Essenciais de Prompt Engineering

Seja bem-vindo à nova era da programação. Você não está mais escrevendo código para a máquina seguir, mas sim comandos de alto nível para uma inteligência. Um *prompt* bem escrito é a chave para transformar uma IA de uma ferramenta passiva em um copiloto de programação extraordinário.

A seguir, listamos as técnicas mais poderosas para dominar a arte de escrever *prompts* eficazes.




# 01

## **Seja um Exterminador**

---

**O Poder da  
Especificidade**



Muitos prompts falham porque são vagos. A IA é uma máquina de padrões, e detalhes claros eliminam ambiguidades, levando a resultados mais precisos e úteis. Pense na IA como um estagiário júnior: você precisa dizer exatamente o que quer.

Prompt ruim:

"Me dê código Python para acessar um site."

Prompt bom:

"Gere uma função assíncrona em **Python** usando a biblioteca **requests** para fazer um GET na URL <https://api.exemplo.com/dados> e extrair o campo **user\_id** do JSON, tratando possíveis erros de conexão."

Prompt ruim:

"Resuma este artigo."

Prompt bom:


"Resuma este artigo de 2000 palavras em **três parágrafos curtos**. Use uma linguagem formal e liste os **três pontos principais** em uma seção separada de bullet points."

# 02

## O Protocolo Asimov

---

**Atribuição de Papel  
(Role-Playing)**



Diga à IA quem ela é. Ao atribuir um **papel** ou uma **persona** (ex: "Você é um especialista em segurança cibernética...", "Atue como um programador sênior em Go..."), você ativa um subconjunto de conhecimento especializado, elevando a qualidade e o tom da resposta.

Exemplo de prompt

"Atue como um **desenvolvedor Sênior de Back-end**. Analise o seguinte trecho de código **Java** e me diga qual é o **erro de *thread safety*** e como corrigi-lo usando **synchronized** ou **ReentrantLock**."

Outro exemplo:


"Você é um **Arquiteto de Soluções Cloud**. Projete uma infraestrutura em AWS para um aplicativo que espera 1 milhão de usuários/dia. Sugira os serviços (RDS, EC2, Lambda) e justifique suas escolhas de **escalabilidade**."

# 03

## Construção Passo a Passo

---

O Método da Cadeia de  
Pensamento (CoT)



Em vez de pedir o resultado final de uma só vez, peça à IA para **pensar em voz alta** ou detalhar o raciocínio. Essa técnica, conhecida como *Chain-of-Thought* (CoT), força a IA a processar a lógica, resultando em respostas mais coerentes e menos erros.

### Exemplo de prompt

"Para o seguinte problema de lógica booleana:  $P \wedge (Q \vee \neg P)$ , simplifique a expressão. **Primeiro**, mostre a lei que você está aplicando. **Segundo**, execute a simplificação. **Terceiro**, me dê o resultado final."

### Outro exemplo:

"Eu preciso de um script que automatize o *deployment* de um site estático no Vercel. **1.** Liste os passos do *workflow* no GitHub Actions. **2.** Escreva o código YAML para o *workflow* que executa o `npm install` e `npm run build` e, por fim, o *deploy*."




# 04

## A Sentença T-800

---

Restrições e Limitações



Defina o que a IA **não deve** fazer ou o **limite** que ela deve respeitar. Usar palavras-chave como "**Apenas**," "**Exclua**," "**Máximo de X**" garante que a resposta permaneça dentro dos seus parâmetros definidos.

Exemplo de prompt

"Crie 5 exemplos de nomes de variáveis seguindo o padrão **camelCase** para um projeto de comércio eletrônico. **Apenas** forneça os nomes, **não inclua** nenhuma explicação ou definição."

Outro exemplo:


"Explique o conceito de *API Gateway* usando o **máximo de 150 palavras**. **Exclua** qualquer menção a provedores de nuvem específicos (AWS, Azure, GCP)."

# 05

## Feedback do Sistema

---

A Tática do Refinamento  
Iterativo



Em vez de tratar o *prompt* como uma única interação, use a IA em um **ciclo de *feedback***. O segredo é pedir à IA para **criticar, refinar ou depurar** a sua própria saída (ou a de um código). Isso imita o processo de revisão de código, onde o resultado é gradualmente melhorado.

### Exemplo de prompt

"O código JavaScript a seguir funciona, mas é lento. **Identifique** a seção com a maior complexidade de tempo (*Big O* notation) e **reescreva** a função para ser mais eficiente, mantendo a mesma saída."

### Outro exemplo:

"Você gerou a documentação de API anteriormente. Agora, **revise-a** para garantir que todos os exemplos de código estejam em **cURL** e que o **código de status HTTP** para cada endpoint esteja explicitamente listado."


# 06

## **Data Input Matrix**

---

A Estrutura de Entrada  
de Dados





Quando você tem muitos dados para processar (como listas de variáveis, requisitos ou configurações), não jogue tudo no texto corrido. Use formatos **estruturados** como tabelas, JSON ou XML dentro do *prompt*. Isso ajuda a IA a interpretar os dados de forma mais limpa e a evitar erros de leitura.

### Exemplo de prompt

"Eu tenho a seguinte lista de *endpoints* e seus métodos HTTP em formato JSON. **Gere** a documentação OpenAPI (Swagger) para cada um deles. O campo **required\_auth** indica se o token JWT é necessário."

### Outro exemplo:

"Com base na tabela de requisitos de usuário abaixo, **crie** um *script* de migração para o banco de dados PostgreSQL. Para os status, crie um **ENUM** e para os valores, crie a *query* **INSERT**."

# 07

## Zero e Few-Shot

---

A Arte da Demonstração

Esta técnica define o quanto você precisa "ensinar" a IA no *prompt*:

- **Zero-Shot:** Não há exemplos. Você confia apenas na instrução.
- **Few-Shot:** Você fornece **um ou mais exemplos** de como a entrada deve ser processada para obter a saída desejada. Isso é essencial para tarefas que exigem um estilo ou formato muito específico.

Exemplo de prompt

"Você deve transformar títulos de *commits* em formato de *Conventional Commit*. Use o formato `type(scope) : subject`. Veja o exemplo:

**Exemplo de Entrada:** Otimizei a função de login.

**Saída Desejada:** `feat(auth): otimiza função de login`

**Agora, transforme o seguinte:** Corrigi um bug no *checkout* que estava travando o pagamento."

Outro exemplo:

"Quero converter mensagens de erro genéricas em mensagens amigáveis para o usuário. Use este padrão:

**Erro Bruto:** `NullPointerException: user_id is null`

**Mensagem Amigável:** 'Não foi possível carregar as informações. Tente fazer login novamente.'


**Converta:** `SyntaxError: Unexpected token '}'`"

# 08

## O Paradigma da Colaboração

---

Você é o Controlador da Skynet



Você completou a exploração das principais técnicas para dominar o *Prompt Engineering*. É hora de estabelecer a perspectiva correta sobre o seu papel neste novo ecossistema de desenvolvimento.

Apesar da temática épica, a realidade é muito mais colaborativa do que de confronto. As IAs, por mais avançadas que sejam, **não estão aqui para substituir o programador humano**, mas sim para atuar como uma **ferramenta de potencialização** de conhecimento e otimização de tempo.

## **A IA como Copiloto (e Não o Piloto)**

O verdadeiro poder das IAs no nicho de programação reside em sua capacidade de:

1. **Acelerar a Produção:** Gerar código *boilerplate* ou rascunhos rapidamente.
2. **Otimizar o Tempo:** Permitir que o desenvolvedor se concentre em problemas complexos de arquitetura e lógica de negócio, e não na sintaxe repetitiva.
3. **Aprimorar o Conhecimento:** Sugerir abordagens, documentar APIs e explicar conceitos complexos instantaneamente.



## O Papel Insubstituível do Humano

É fundamental lembrar que a IA opera com base em padrões e dados de treinamento. Ela não possui a capacidade de **raciocínio crítico contextual, validação de requisitos éticos** ou **compreensão total** das implicações de um código em um sistema complexo e único.

Portanto, o ciclo de trabalho com a IA exige a intervenção humana:

- **Verificar a Saída:** O código gerado pela IA pode conter erros sutis, vulnerabilidades de segurança ou ineficiências. Você deve atuar como o **revisor e testador primário**.
- **Corrigir e Refinar:** O *prompt* raramente é perfeito de primeira. É o seu trabalho **refinar o código** para atender aos padrões específicos do seu projeto (formatação, *design patterns*, etc.).
- **Validar a Lógica:** O código está sintaticamente correto, mas **ele realmente resolve o problema de negócio?** A validação final da lógica e dos requisitos é sempre uma responsabilidade humana.

O *Prompt Engineering* transforma você no **Arquiteto da Comunicação**. Ao dominar as técnicas deste Ebook, você se torna o colaborador mais valioso da IA, garantindo que a tecnologia sirva à sua criatividade e eficiência, e não o contrário.

**O futuro da programação é um ciclo de colaboração inteligente, onde a máquina gera e o humano valida, corrige e inova.**

# AGRADECIMENTOS



# OBRIGADO POR LER ATÉ AQUI

Esse Ebook foi gerado por IA, e diagramado por humano.  
O passo a passo se encontra no meu Github

•

Esse conteúdo foi gerado com fins didáticos de construção,  
não foi realizado uma validação cuidadosa humana no  
conteúdo e pode conter erros gerados por uma IA.



<https://github.com/rafaelpoh/prompts-recipe-to-create-a-ebook>