APOSTILA DE DOMÍNIO · MATERIAL DE ESTUDO DO APRESENTADOR

Claude Code para Idosos

O tema a fundo, para você dominar e ensinar com autoridade.

10 capítulos · glossário de **24 termos** · baseado no **FATOS-VERIFICADOS.md** · reconferir preços/modelos na véspera (28/06)

Índice

01 Inteligência Artificial e LLMs: o que está por trás

Antes de você ensinar alguém a usar o Claude Code, precisa conseguir responder, com calma e sem gaguejar, a pergunta que TODO iniciante faz na primeira aula: "mas afinal, o que é isso? É um robô? Ele pensa? Como ele sabe o que escrever?". Se você titubear aqui, perde a autoridade na largada. Se responder bem, o resto da aula flui, porque a pessoa para de ter medo de uma caixa-preta mágica e passa a entender que está lidando com uma ferramenta, poderosa e com limites claros.

Este capítulo te dá o chão. Você vai entender o que é Inteligência Artificial sem virar professor de computação, o que é um LLM (o "motor" por trás do Claude) e como ele consegue escrever português que parece humano. E, principalmente, vai entender o calcanhar de Aquiles dessa tecnologia: a alucinação, o hábito de inventar com cara de quem sabe, para que você ensine seus alunos a confiar na medida certa: usar muito, conferir sempre o que importa. Quem entende a mecânica não cai em armadilha e não vende ilusão.

Inteligência Artificial sem mistério: imitar o resultado, não copiar o cérebro

Inteligência Artificial é o nome que se dá a programas de computador feitos para fazer tarefas que, até pouco tempo, só uma pessoa conseguia fazer: entender uma frase, reconhecer um rosto numa foto, traduzir um texto, sugerir o próximo filme que você vai gostar. Não é um cérebro eletrônico, não é uma criatura consciente, não é o robô dos filmes. É software (instruções rodando numa máquina) só que de um tipo que aprende com exemplos em vez de seguir uma receita fixa escrita à mão.

A diferença entre IA e um programa comum vale a pena fixar, porque é ela que assusta as pessoas. Um programa comum é uma receita de bolo: o programador escreve 'se o cliente clicar aqui, mostre aquilo', passo a passo, e o computador obedece à risca. Já a IA moderna não recebe a receita pronta. Ela recebe milhões de exemplos e descobre sozinha os padrões. É a diferença entre ensinar uma criança a reconhecer um cachorro listando todas as regras ('tem quatro patas, focinho, late...') e simplesmente mostrar duas mil fotos de cachorros até ela pegar o jeito. A IA aprende pelo segundo caminho.

Um detalhe que tranquiliza o aluno iniciante: 'inteligência', aqui, é uma palavra de marketing, não de filosofia. A máquina não 'entende' no sentido humano, não tem intenção, não quer nada. Ela ficou muito boa em produzir o resultado que uma pessoa inteligente produziria, que é uma coisa bem diferente de ser inteligente. Pense num papagaio extremamente treinado que repete frases no momento certo: o resultado impressiona, mas ninguém acha que o papagaio entende português. Guardar essa distinção evita os dois erros opostos: nem achar que é mágica todo-poderosa, nem desprezar como 'só um truquezinho'.

O que é um LLM: o motor de linguagem por trás do Claude

Quando você conversa com o Claude, com o ChatGPT ou com o Gemini, o que está rodando por baixo é um LLM. A sigla vem do inglês Large Language Model, que em português é 'Grande Modelo de Linguagem'. Vale destrinchar palavra por palavra, porque o nome já explica quase tudo.

'Linguagem' (Language): é uma máquina especializada em texto: ler, entender o sentido e produzir texto novo. Não foi feita para fazer conta nem para guardar um banco de dados. Foi feita para lidar com palavras. 'Modelo' (Model): é uma representação, um mapa estatístico de como a língua funciona: quais palavras costumam andar juntas, como uma frase normalmente continua, que tom combina com que assunto. 'Grande' (Large): é a escala absurda do treinamento. O modelo leu uma quantidade gigantesca de texto (livros, sites, artigos, conversas) e tem uma capacidade interna enorme de guardar esses padrões. É o 'grande' que faz a diferença entre um corretor de celular que erra direto e algo que escreve um e-mail inteiro coerente.

A analogia que funciona bem na aula é a do estagiário que leu uma biblioteca inteira. Imagine uma pessoa que passou anos lendo praticamente tudo o que já se escreveu (toda a internet, milhares de livros) mas que não decorou nada palavra por palavra. O que ficou foi a intuição: ela 'sente' como um texto bem escrito soa, sabe responder no tom certo, reconhece o assunto e devolve algo no formato esperado. É exatamente isso que o LLM é. Ele não tem uma gaveta com a resposta certa guardada. Ele tem uma intuição treinada sobre como a linguagem se comporta. Por isso ele escreve bem sobre quase qualquer tema, mas (como você vai ver adiante) também por isso às vezes 'chuta com confiança'.

Como ele escreve: prevendo a próxima palavra, uma de cada vez

Aqui está o segredo que desmistifica tudo, e é o conceito mais importante deste capítulo. No fundo, um LLM faz UMA coisa só, repetida muitas vezes: dado o texto até agora, ele prevê qual é a próxima palavra mais provável. Só isso. Ele escreve uma palavra, depois olha tudo de novo (incluindo a palavra que acabou de escrever) e prevê a próxima, e a próxima, e a próxima, até a resposta ficar pronta.

A melhor analogia é o teclado do celular, aquele que sugere a próxima palavra enquanto você digita. Se você escreve 'bom', ele sugere 'dia'. Se escreve 'muito obrigado pela', ele sugere 'atenção'. O LLM é esse mesmo mecanismo de autocompletar, só que numa potência incomparável: em vez de sugerir uma palavrinha óbvia, ele consegue manter o raciocínio de um parágrafo inteiro, lembrar do que foi dito lá no começo da conversa e ajustar o tom ao que você pediu. Mesma ideia básica, escala completamente diferente. Quando você entende isso, a 'mágica' desaparece e fica a engenharia: é previsão de texto, palavra após palavra, numa qualidade altíssima.

Duas consequências práticas saem direto dessa mecânica, e você deve ensiná-las. Primeira: o LLM constrói a resposta na hora, ele não busca uma resposta pronta numa gaveta como o Google faz com páginas. Por isso a mesma pergunta pode gerar respostas um pouco diferentes em dias diferentes. Ele está reconstruindo o texto, não recuperando um arquivo. Segunda, e mais importante: como ele sempre escolhe a continuação que 'soa mais provável e mais natural', ele tende a produzir texto que parece certo, esteja ele certo ou não. Soar bem e estar correto são duas coisas separadas para uma máquina que só sabe prever o que soa bem. É exatamente daí que nasce o problema da próxima seção.

Onde ele brilha: texto, resumo, ideias e código

Entendido o motor, fica fácil saber onde apontá-lo. O LLM é excelente em tudo que é, na essência, trabalho com linguagem, e isso cobre boa parte do dia de quem trabalha com marketing. Quatro forças se destacam.

Primeira, produzir texto. Escrever um e-mail, três versões de uma legenda, um roteiro, uma descrição de produto, um rascunho de proposta. Ele nunca trava diante da página em branco e entrega rápido várias opções para você escolher e lapidar. Segunda, resumir e reorganizar. Você cola um texto longo (a transcrição de uma reunião, um artigo de cinco páginas, um monte de comentários de clientes) e pede os pontos principais, ou pede para transformar aquilo numa lista, numa tabela, num parágrafo curto. Ele é ótimo em pegar muita informação e enxugar sem perder o essencial. Terceira, gerar ideias. Como ele 'viu' uma enormidade de exemplos, é um parceiro de brainstorm que não cansa: peça vinte nomes para uma campanha, dez ângulos para um anúncio, cinco objeções que um cliente poderia levantar. Quarta, lidar com código de programação. Código é uma linguagem como outra qualquer, com gramática rígida, e o LLM aprendeu essa linguagem junto com o português. É exatamente essa força que o Claude Code explora, e que você vai explorar em marketing (montar uma página em HTML, organizar arquivos, automatizar uma tarefa repetitiva) sem precisar ser programador.

O fio que une essas quatro forças é o mesmo: são tarefas onde existem muitas respostas boas possíveis e o valor está em produzir um bom rascunho com velocidade. O LLM é um gerador de primeiras versões de altíssima qualidade. O trabalho de quem usa não é mais começar do zero. É dirigir, escolher, corrigir e refinar. Essa mudança de papel, do 'fazedor' para o 'diretor', é a mentalidade que você quer plantar no seu aluno logo de cara.

O limite que você nunca pode esconder: a alucinação

Agora a parte que separa quem ensina com responsabilidade de quem vende fumaça. O LLM tem um defeito estrutural, não um bug que vão consertar amanhã, mas uma consequência direta de como ele funciona. Esse defeito tem nome: alucinação. É quando o modelo inventa uma informação que não existe e a apresenta com a mesma segurança e fluência de uma informação verdadeira. Ele não mente no sentido humano (mentir exige saber a verdade e esconder). Ele simplesmente preenche a lacuna com o que 'soa mais provável', e às vezes o que soa provável é falso.

Volte à mecânica da terceira seção e você vê por que é inevitável. A máquina foi feita para escolher a próxima palavra mais natural, não a mais verdadeira. Ela não tem um verificador de fatos embutido. Se você pergunta o nome de um livro sobre determinado assunto e nenhum título real 'vem com força' na previsão, ele monta um título que parece perfeitamente plausível: autor crível, ano crível, capa crível na descrição. Tudo coerente, tudo convincente, e o livro nunca existiu. Os exemplos clássicos para mostrar na aula: pedir referências e ele inventa fontes. Pedir uma estatística e ele cospe um número redondo e confiante. Perguntar sobre um detalhe muito específico de uma lei, um endereço ou uma data e ele entrega algo bem-acabado e errado. O perigo não está no erro em si (todo mundo erra) está na embalagem: o erro vem com a mesma voz firme e bem escrita das respostas certas, sem nenhum aviso de 'olha, aqui eu não tenho certeza'.

A regra de ouro que sai disso, e que deve virar mantra na sua aula, é simples: use o LLM à vontade para criar, rascunhar e pensar junto. Mas todo fato verificável que importa (nome, número, data, preço, citação, dado de cliente, qualquer coisa que você vá publicar ou apresentar como verdade) você confere na fonte antes de usar. Para gerar três versões de uma headline, não precisa conferir nada, é tudo criação. Para afirmar 'pesquisa mostra que 73% dos pacientes...', você confere a pesquisa, ponto. Saber separar o trabalho criativo (onde a alucinação não atrapalha) do trabalho factual (onde ela é veneno) é a habilidade número um do usuário maduro. É isso que você entrega de mais valioso neste capítulo inteiro.

COMO EXPLICAR ISSO PRO ALUNO

- Sobre o que é IA: 'Esquece o robô dos filmes. Inteligência Artificial é só um programa de computador que aprendeu a fazer uma tarefa olhando milhões de exemplos, em vez de seguir uma receita pronta. É a diferença entre ensinar uma criança a reconhecer um cachorro listando regras ou simplesmente mostrando duas mil fotos. A IA aprende pelas fotos.'
- Sobre como o LLM escreve: 'Sabe o teclado do seu celular que sugere a próxima palavra? Você digita bom e ele sugere dia. O Claude é exatamente esse autocompletar, só que num nível absurdo: em vez de uma palavrinha, ele mantém um raciocínio de parágrafos inteiros. No fundo ele faz uma coisa só, adivinhar a próxima palavra mais provável, uma de cada vez, até a resposta ficar pronta.'
- Sobre a força dele: 'Pense nele como um estagiário gênio que leu a internet inteira e nunca trava na página em branco. Você não pede pra ele a resposta final. Você pede o primeiro rascunho. Dez legendas, cinco ângulos de anúncio, um e-mail. Em segundos. Seu trabalho deixa de ser começar do zero e passa a ser escolher e melhorar.'
- Sobre a alucinação: 'A única coisa que você nunca pode esquecer: às vezes ele inventa, e inventa com a maior cara de certeza do mundo. Pede uma estatística e ele te dá um número redondo e bonito que nunca existiu. Por quê? Porque ele foi feito pra escrever o que soa mais natural, não o que é verdade. Então a regra é: usa à vontade pra criar, mas todo número, nome, data e citação que você for publicar, você confere antes. Cria com ele, confirma você.'

PONTOS DE ATENÇÃO

- Não diga que a IA 'pensa', 'entende' ou 'tem consciência'. Ela produz o resultado que uma pessoa inteligente produziria, sem nada disso por dentro. Use sempre verbos como 'prevê', 'gera', 'reconhece padrões'. Isso mantém o aluno com a expectativa calibrada e te blinda da pergunta 'então ela é viva?'.
- Cuidado para não passar a ideia de que o LLM 'busca' respostas como o Google. Ele não consulta uma gaveta com a resposta certa: ele reconstrói o texto na hora, prevendo palavra por palavra. É por isso que a mesma pergunta pode ter respostas um pouco diferentes, e é por isso que ele pode inventar. Confundir as duas coisas leva o aluno a confiar demais.
- Nunca apresente a alucinação como um defeito que 'logo vão corrigir' ou minimize com 'mas é raro'. É consequência direta de como a tecnologia funciona. Está presente em todos os modelos, inclusive nos melhores. Prometer que sumiu é vender ilusão e destruir sua autoridade na primeira vez que o aluno pegar um erro.
- Resista a citar números de treinamento ('foi treinado com X bilhões de palavras', 'tem Y parâmetros'). Esses dados variam, não estão no seu bloco de fatos e não ajudam o iniciante. Fique na ideia relativa ('leu uma quantidade gigantesca de texto') que é verdadeira, suficiente e impossível de te deixar em má posição se alguém checar.

02 A Anthropic e a família de modelos

Quem você convida para a mesa importa tanto quanto o que vai ser feito nela. No capítulo anterior ficou claro o que é um modelo de linguagem e por que ele às vezes inventa com confiança. Agora a pergunta muda de "como essa tecnologia funciona" para "em quem eu estou confiando para colocá-la dentro do meu computador e do meu trabalho". E essa pergunta tem um peso prático: a ferramenta que você vai ensinar não é um site qualquer onde se cola um texto. É um agente que vai ler seus arquivos, mexer em pastas e executar tarefas no seu nome. A empresa por trás dela e a filosofia que ela carrega deixam de ser detalhe institucional e viram parte da decisão.

Este capítulo te dá duas coisas para ensinar com autoridade. Primeira: quem é a Anthropic e por que a aposta declarada dela em "IA segura e confiável" não é marketing vazio. Ela aparece no comportamento da ferramenta, naquele "pedir antes de agir" que o aluno vai ver na tela. Segunda: a família de modelos que existe por baixo do Claude (Haiku, Sonnet, Opus, Fable e Mythos) e o trade-off real entre velocidade, capacidade e custo. O ponto que você vai martelar para os iniciantes é libertador: ninguém precisa escolher modelo na mão para começar. Mas você, que ensina, precisa entender o cardápio inteiro para responder quando alguém perguntar "e esse tal de Opus, é melhor?".

Quem é a Anthropic e por que a palavra 'segurança' aparece duas vezes na proposta dela

A Anthropic é a empresa que cria e mantém o Claude e toda a família de modelos que veremos neste capítulo. O traço que a define, e que você deve destacar ao ensinar, é o foco declarado em inteligência artificial segura e confiável. Não é um slogan jogado no rodapé do site: é a forma como a empresa se posiciona publicamente e, mais importante para o seu aluno, é algo que se reflete no comportamento da ferramenta na prática.

Vale explicar a diferença entre as duas palavras, porque elas não são sinônimos. 'Segura' (safe) tem a ver com a IA não causar dano: não executar uma ação destrutiva por conta própria, não te empurrar para algo perigoso, parar para pedir confirmação antes de mexer em algo importante. 'Confiável' (trustworthy) tem a ver com você poder depender do que ela faz: previsibilidade, honestidade quando não sabe a resposta, transparência sobre o que está prestes a fazer. Quando você unir isso ao Capítulo 6, que trata do 'pedir antes de agir', vai ficar evidente: aquele comportamento de mostrar o que vai mudar e esperar seu 'Aceitar' ou 'Rejeitar' é a filosofia da empresa virando botão na tela.

Por que isso importa para quem usa, e não só para quem estuda IA? Porque o Claude Code não é um chat inofensivo. Ele pode abrir seus arquivos, criar pastas, escrever um site, rodar comandos. A diferença entre uma ferramenta assim ser uma aliada ou um risco está justamente na cultura de quem a construiu. Uma analogia honesta: é como escolher entre um eletricista que desliga a chave geral antes de mexer na fiação e te explica cada passo, e outro que mete a mão no fio com a casa energizada. Os dois trocam a tomada. Você quer o primeiro dentro da sua casa, e, no caso do Claude, dentro do seu computador.

Por que existe uma família de modelos, e não um único 'Claude'

Para o iniciante, 'Claude' é uma coisa só. Para você que ensina, é útil entender que por baixo existe uma família de modelos, versões diferentes do mesmo cérebro, cada uma calibrada para um ponto distinto entre três forças que puxam em direções opostas: velocidade, capacidade e custo.

A razão de existir mais de um modelo é simples e cabe numa analogia de cozinha. Você não usa a mesma faca para descascar um alho e para desossar um peixe. A faca pequena é rápida e ágil para a tarefa simples. A grande dá conta do trabalho pesado, mas seria exagero pegá-la para descascar um dente de alho. Modelos de IA seguem a mesma lógica: um modelo mais 'leve' responde quase instantaneamente e gasta menos recurso, perfeito para tarefas diretas. Um modelo mais 'poderoso' raciocina sobre problemas complicados, mas é mais lento e consome mais.

O trade-off central, que você deve saber explicar de cor, é este: quanto mais capaz o modelo, mais lento e mais caro ele tende a ser. Quanto mais rápido e barato, menos ele aguenta de complexidade. Não existe o modelo que seja ao mesmo tempo o mais rápido, o mais inteligente e o mais barato. Se existisse, os outros não precisariam existir. A graça de ter uma família é justamente poder casar a ferramenta certa com o tamanho do problema. E a boa notícia, que vamos detalhar mais à frente, é que na prática do dia a dia essa escolha costuma ser feita automaticamente, sem o iniciante precisar tocar em nada.

O cardápio por dentro: Haiku, Sonnet e Opus, a linha clássica

Vamos aos nomes, do mais leve ao mais poderoso dentro da chamada linha clássica. Os nomes não são técnicos de propósito. São tipos de texto, do mais curto ao mais elaborado, o que já dá uma pista da ideia.

Haiku 4.5 é o mais rápido e leve da família. O haiku é aquele poema japonês de três versinhos. O nome sugere algo curto e ágil. É o modelo para tarefas diretas em que a velocidade importa mais que o raciocínio profundo. Sonnet 4.6 é o equilíbrio, o do dia a dia. Um soneto é mais longo e estruturado que um haiku, mas ainda enxuto. Esse é o modelo que junta rapidez e inteligência num ponto confortável para a maior parte do que você faz no cotidiano. Opus 4.8 é o mais poderoso da linha clássica. Uma 'opus' é uma obra de grande fôlego. É o modelo para raciocínio complexo e programação pesada, quando o problema exige que a IA pense com mais profundidade, mesmo que isso custe um pouco mais de tempo e de recurso.

Repare na lógica de escalada: Haiku (mais rápido) é o ponto de partida, Sonnet fica no meio como o equilíbrio do dia a dia, e Opus fica no topo da linha clássica como o mais robusto. Um número importante para sua autoridade: esses '4.5', '4.6', '4.8' são versões, e versões mudam. Você não precisa decorar, precisa saber que existe essa ordem de leve para poderoso, e que o número é só o registro de qual geração está no ar naquele momento.

O topo da linha: Fable 5 e Mythos 5, e a diferença que muda a resposta certa

Acima da linha clássica existem dois modelos mais avançados, e aqui mora uma sutileza que separa quem entende de quem decora. Fable 5 é, desde 9 de junho de 2026, o modelo mais capaz disponível ao público. 'Fable' é fábula, uma narrativa mais rica e elaborada. O nome continua a brincadeira de tipos de texto, agora num patamar acima. É o modelo para o raciocínio mais exigente, e qualquer pessoa pode acessá-lo. Mythos 5 vai ainda mais longe em capacidade, mas tem acesso restrito por convite. Não está disponível ao público geral. 'Mythos' é mito, a narrativa máxima. O nome combina com a posição de topo, mas o ponto prático é o acesso fechado.

Por que essa distinção importa tanto para você que ensina? Porque a pergunta 'qual é o modelo mais capaz?' tem duas respostas certas, e dar a errada para o público errado te faz tropeçar. Se alguém pergunta 'qual o mais avançado que EU posso usar?', a resposta é Fable 5, porque é o mais capaz disponível ao público. Se a conversa é sobre a linha clássica do dia a dia, o topo é o Opus 4.8. E se mencionarem o Mythos 5, a honestidade que vira autoridade é dizer: 'esse é ainda mais avançado, mas o acesso é restrito por convite, não dá para qualquer um usar'. Para os iniciantes da aula, não enrole com o Mythos. O destaque prático e útil é Opus 4.8 e Fable 5. Citar um modelo que eles não conseguem acessar só gera frustração.

O alívio do iniciante: você não escolhe modelo na mão para começar

Aqui está a mensagem mais importante deste capítulo para quem está começando, e ela é tranquilizadora: o iniciante não precisa, e nem deve, ficar escolhendo modelo manualmente. Todo o cardápio que acabamos de detalhar é conhecimento de bastidor, útil para você responder perguntas, irrelevante para o aluno dar o primeiro passo.

Na prática, o que ele faz é descrever a tarefa em português, e o sistema cuida de usar o modelo adequado para aquilo. Pense num carro automático: você não troca de marcha a cada subida e descida. Pisa no acelerador e o câmbio escolhe a marcha certa para a situação. Saber que existem marchas (e que um carro de corrida tem mais) é cultura automobilística. Mas para chegar ao trabalho de manhã, você só dirige. Com o Claude é igual: a 'troca de marcha' entre velocidade e capacidade acontece nos bastidores, e o iniciante só precisa pisar no acelerador, ou seja, escrever o pedido.

Então por que você gastou energia aprendendo a família inteira? Por dois motivos. Primeiro, autoridade na frente da câmera: quando um aluno perguntar 'mas e o tal do Opus que vi num vídeo?', você responde com naturalidade em vez de gaguejar. Segundo, porque conforme a pessoa avança, ela pode querer, sim, pedir conscientemente um modelo mais robusto para uma tarefa cabeluda, e aí o conhecimento que parecia decorativo vira ferramenta. Mas isso é um capítulo posterior na vida do usuário. No começo, a regra de ouro é: descreva bem a tarefa e deixe o sistema decidir o resto. Quanto menos botão o iniciante tiver que apertar, mais rápido ele vê valor, e mais fácil fica a sua aula.

COMO EXPLICAR ISSO PRO ALUNO

- 'Quem fez a ferramenta importa porque ela vai mexer nos seus arquivos. A Anthropic, que criou o Claude, é uma empresa cuja bandeira é IA segura e confiável, e isso não é frase bonita: é por isso que, na hora de fazer qualquer coisa no seu computador, o Claude para e te mostra o que vai fazer antes de fazer. É um eletricista que desliga a chave geral antes de mexer na fiação.'
- 'Existe mais de um Claude por baixo do capô, e a ideia é a mesma de uma cozinha: você não usa a faca de desossar para descascar um alho. Tem o modelo rápido e leve para tarefa simples (Haiku), o equilibrado do dia a dia (Sonnet) e o mais poderoso para problema pesado (Opus). Quanto mais inteligente, mais lento e mais caro. Não existe o que seja tudo ao mesmo tempo.'
- 'Se te perguntarem qual é o modelo mais avançado, tem que saber para quem está respondendo. O mais capaz que QUALQUER UM pode usar hoje é o Fable 5. Tem um ainda mais forte, o Mythos 5, mas o acesso é só por convite. Não adianta sonhar com ele porque não está aberto ao público.'
- 'Calma que você não vai ter que escolher modelo nenhum. É como dirigir carro automático: você não troca a marcha, pisa no acelerador e o carro faz o resto. Aqui você descreve a tarefa em português e o sistema usa o modelo certo sozinho. Saber os nomes é só cultura geral. Para começar, é só pedir.'

PONTOS DE ATENÇÃO

- Não trate versão de modelo como verdade eterna. Os números (Haiku 4.5, Sonnet 4.6, Opus 4.8, Fable 5, Mythos 5) e as datas são o lineup verificado em 25/06/2026. Eles mudam rápido. Reconfira a família na véspera da aula (28/06) na página oficial e ensine o conceito de 'leve a poderoso', não os números decorados, porque o conceito não envelhece e o número sim.
- Cuidado com a armadilha do 'modelo mais capaz', que tem resposta dupla. Ao público amplo, o mais capaz disponível é o Fable 5. Dentro da linha clássica, o topo é o Opus 4.8. Dizer só 'o Opus é o melhor' é ambíguo e tecnicamente incompleto desde que o Fable existe. E nunca prometa o Mythos 5 como algo usável. O acesso é restrito por convite, não é público.
- Não invente capacidades, benchmarks, números de parâmetros ou comparações de desempenho entre os modelos. Os fatos confirmados são apenas a ordem (leve → poderoso), o papel de cada um e o status de acesso. Qualquer afirmação do tipo 'o Opus é X% melhor que o Sonnet' ou 'o Fable tem tantos bilhões de parâmetros' é chute: fora do escopo verificado e proibido na aula.
- Não confunda o aluno com excesso de cardápio. O objetivo da seção é tranquilizar, não sobrecarregar. Apresente a família como contexto e deixe explícito que ninguém escolhe modelo na mão para começar. O sistema decide. Despejar cinco nomes técnicos sem essa moldura de 'você não precisa decidir isso' assusta o iniciante e trabalha contra a própria aula.

03 O trio: Claude, Cowork e Claude Code

Quase todo mundo que ouve "Claude" pensa numa só coisa: aquela caixinha de texto onde você digita uma pergunta e ele responde. Está certo, mas é só um terço da história. A Anthropic não vende um produto, ela oferece três formas de trabalhar com a mesma inteligência por trás: o Claude (o chat, que conversa e responde), o Cowork (que entrega o trabalho pronto sem você precisar entender uma linha de código) e o Claude Code (que põe a mão na massa dentro do seu computador). Quem confunde os três acaba subutilizando a ferramenta. Pede pro chat fazer o trabalho do Code, ou tem medo do Code achando que vai precisar virar programador. Não vai.

Este capítulo separa os três com clareza, explica o conceito que muda tudo (o de "agente de IA") e mostra, com exemplos de marketing, quando você abre cada um. No fim, você vai conseguir olhar pra uma tarefa e saber na hora: "isso é conversa de chat" ou "isso é trabalho de agente". É essa distinção que faz a diferença entre usar o Claude como um Google mais esperto e usá-lo como um estagiário que executa.

O Claude: o chat que pensa junto com você

O Claude é o ponto de partida e o que mais gente conhece. Funciona como uma conversa: você escreve, ele responde, você refina, ele ajusta. É um assistente de texto: escreve, resume, analisa, dá ideias, traduz, estuda um assunto com você, organiza pensamento. Roda no site claude.ai, nos aplicativos de celular (iPhone e Android), e dentro do app desktop. No plano gratuito, é exatamente isso que você tem acesso: só o chat.

O ponto que precisa ficar nítido é o limite dele. O Claude no modo chat vive dentro da própria janela de conversa. Ele lê o que você cola ali e devolve texto ali. Ele não abre o seu computador, não mexe nos seus arquivos, não cria uma pasta, não roda um programa. Se você pede 'crie uma página de captura', o chat te devolve o código escrito na tela, e você teria que copiar, colar num arquivo, salvar com o nome certo e abrir no navegador por conta própria. Ele te dá o peixe escrito num papel. Quem cozinha é você.

Para marketing, o chat resolve uma fatia enorme do dia: reescrever um e-mail que ficou seco, gerar dez variações de assunto, criticar uma oferta, resumir um PDF de pesquisa de mercado, transformar um áudio transcrito em roteiro. Tudo isso é 'pergunta e resposta': entra texto, sai texto. Sempre que a sua tarefa termina quando o Claude para de escrever, o chat é a ferramenta certa, e a mais leve.

O que é um 'agente de IA' (e por que isso muda tudo)

Antes de explicar o Claude Code, precisamos do conceito que está embaixo dele: o agente. Essa é a palavra mais importante do capítulo.

Um assistente de chat responde. Um agente age. A diferença não é de tamanho, é de natureza. Quando você dá uma tarefa a um agente, ele não devolve uma resposta única e pronta. Ele quebra a tarefa em vários passos, executa um, olha o resultado, decide o próximo, executa, corrige se deu errado, e segue até a coisa estar feita. Ele planeja e executa, em sequência, sozinho, mantendo o objetivo na cabeça do começo ao fim.

A analogia mais honesta é a de um estagiário. Se você pergunta a um estagiário 'qual a capital da França', ele responde Paris e acabou. Isso é chat. Mas se você diz 'organize a pasta de criativos da campanha de junho, separe por formato e renomeie tudo no padrão', o estagiário não te responde uma frase: ele vai até a pasta, olha o que tem dentro, cria subpastas, move arquivo por arquivo, renomeia cada um, e volta pra te avisar quando terminou. Ele tomou uma porção de pequenas decisões no caminho sem te perguntar cada uma. Isso é um agente.

Na prática, um agente de IA tem três coisas que o chat não tem: ele consegue usar ferramentas (abrir arquivos, rodar comandos, navegar), ele trabalha em ciclo (faz, observa, ajusta, repete) e ele persegue um objetivo de múltiplos passos sem parar a cada passo pra te perguntar. É por isso que um agente consegue construir um site inteiro a partir de um pedido em português, enquanto o chat só te entrega o texto do código. Guarde esta frase: chat responde, agente realiza. Esse poder de agente chega até você em dois formatos, que vamos ver agora um de cada vez: primeiro o Cowork, a porta de entrada para quem não é técnico, e depois o Claude Code, a forma mais completa.

O Cowork: o agente para quem não é técnico

De todos os formatos, o Cowork é o que mais combina com quem trabalha com marketing e não quer ver uma linha de código. Ele é um recurso do app desktop (disponível nos planos pagos) que pega aquele poder de agente que acabamos de explicar e entrega num formato simples e visual.

A ideia central do Cowork é a delegação. Você descreve uma tarefa de vários passos em português comum ('organize esses arquivos', 'monte esse material', 'prepare esse relatório') e o Claude vai trabalhar nela sozinho, executando os passos, enquanto você toca outra coisa. Em vez de você acompanhar comando por comando, você entrega a tarefa e deixa o agente conduzir. É a diferença entre dirigir o carro você mesmo e chamar um motorista, dizer o destino e relaxar no banco de trás.

Para marketing, o Cowork é onde mora boa parte do valor prático. Peça um relatório e receba uma planilha pronta. Peça a organização de uma pasta de criativos e receba tudo separado e renomeado. Peça um calendário de conteúdo e receba o material montado. Você delega e recebe o resultado, sem precisar entender o que rodou nos bastidores. Por isso, para o público iniciante da sua aula, o Cowork costuma ser a porta de entrada mais confortável. Importante: ele exige, no mínimo, o plano Pro. No plano gratuito, ao clicar em Cowork ou Code, o app pede upgrade.

O Claude Code: o agente em sua forma mais completa

O Claude Code é o mesmo motor de agente do Cowork, na forma mais direta e poderosa, com acesso total ao seu computador. É a mesma inteligência do chat, mas agora com mãos: ele lê os seus arquivos, edita arquivos, cria pastas, roda comandos e constrói coisas (sites, programas, planilhas, automações) em vez de só descrever como se faz.

A diferença com o chat fica óbvia num exemplo. Você pede uma página de captura para a campanha de um lançamento. No chat, ele escreve o código na conversa e o trabalho de transformar aquilo num arquivo que abre no navegador é seu. No Claude Code, você diz a mesma coisa e ele cria o arquivo da página no seu computador, salva com o nome correto, e se você pedir abre no navegador pra você ver o resultado. Você saiu de 'aqui está o texto de como fazer' para 'pronto, está feito e salvo na sua pasta'.

Um ponto que tira o medo de muita gente: o Claude Code não exige que você seja técnico nem instale um monte de programa estranho. Ele vem dentro do app desktop. Quando você instala o aplicativo do Claude no Windows ou no Mac, o Claude Code já está lá dentro, é só clicar na aba Code no topo. Não precisa instalar aquele tal de Node.js que se ouve falar, o app cuida disso. A única exigência real no Windows é o Git instalado (o capítulo de instalação cobre o porquê e como), e no Mac ele quase sempre já vem.

Para o marketing, é aqui que o trabalho operacional mais pesado acontece: montar a estrutura de arquivos de uma campanha, gerar várias páginas de uma vez, renomear e organizar um monte de criativo, criar um relatório a partir de uma exportação de dados, escrever e instalar uma skill. Se o Cowork é o jeito mais confortável de delegar, o Claude Code é o que dá mais controle. O chat te ajuda a pensar, o Code faz o serviço braçal pra valer.

Onde cada um roda: terminal, IDE, app desktop e navegador

Oficialmente, o Claude Code roda em quatro lugares, e vale entender cada um para não se perder quando alguém mencionar um nome estranho.

O terminal é aquela tela preta de comandos, a 'linha de comando', o jeito tradicional dos programadores, em que você digita instruções de texto. A IDE é o programa onde desenvolvedores escrevem código (pense num editor de texto turbinado, tipo o VS Code). O Claude Code pode morar dentro dela. O navegador é o próprio Chrome ou Edge, com o Claude operando ali dentro. E o app desktop é a janela do aplicativo do Claude instalado no seu computador, com botões e visual amigável.

Na sua aula, o foco é o app desktop, e a razão é simples: é o único dos quatro feito para quem não é técnico. Ele tem janela, botões, você vê as mudanças acontecerem de forma visual, sem decorar comando nenhum. Os outros três pressupõem alguma intimidade com ferramentas de programação. Para um iniciante total, mandar abrir o terminal seria como entregar a chave de um avião pra quem nunca dirigiu carro.

Um detalhe técnico que vale guardar como prova de que você domina o assunto: existe um erro chamado 'Raw mode is not supported' que assusta quem topa com ele. Esse erro só aparece na versão de linha de comando (o terminal) quando rodada dentro de um programa específico chamado Git Bash, nunca no app desktop. Ou seja, justamente o caminho que você vai ensinar, o app desktop, está blindado contra esse susto. Se um aluno comentar que viu esse erro num tutorial pela internet, você já sabe explicar: ele estava na linha de comando, não no app.

COMO EXPLICAR ISSO PRO ALUNO

- Pense num garçom e num cozinheiro. O chat do Claude é o garçom: você pede, ele anota e te traz a informação. O Claude Code é o cozinheiro: ele entra na cozinha e prepara o prato de verdade. Mesma inteligência, papéis diferentes: um te responde, o outro faz a coisa acontecer.
- O segredo é a palavra 'agente'. Um assistente comum responde uma pergunta e para. Um agente recebe uma missão de vários passos, planeja, faz um passo, confere, faz o próximo, conserta se errou, e só te chama quando terminou, igual a um bom estagiário a quem você delega uma tarefa inteira em vez de ficar mandando comando por comando.
- Não tenha medo da palavra 'Code'. O Cowork foi feito justamente para você que não programa: descreve a tarefa em português, manda, e o Claude trabalha sozinho. É o mesmo poder do Code, só que sem você ver código nenhum. Você é o chefe que delega, não o técnico que digita.
- Na nossa aula a gente usa o app desktop, aquela janela com botões. Por quê? Porque os outros lugares onde o Claude roda (o terminal, aquela tela preta de comandos) são feitos para programador. O app desktop é o único pensado para a gente, que quer ver as coisas acontecerem na tela sem decorar comando.

PONTOS DE ATENÇÃO

- Não diga que 'Claude, Cowork e Claude Code são três IAs diferentes'. É a mesma inteligência da Anthropic por baixo, em três formatos de uso. O que muda é o que cada formato pode fazer (só responder × agir no computador) e o público (técnico × não-técnico), não o cérebro.
- Cuidado para não prometer que o chat gratuito 'cria seu site'. O chat só devolve o texto/código na conversa. Quem cria o arquivo e o salva no computador é o Claude Code/Cowork, que exigem plano pago (Pro ou superior). No Free, clicar em Code pede upgrade. Seja honesto sobre essa fronteira.
- Ao explicar 'onde roda', não trate terminal, IDE e navegador como se fossem o foco da aula. São opções oficiais reais, mas a aula é sobre o app desktop. Mencione os quatro para mostrar domínio, e deixe claro por que o iniciante fica no app desktop.
- O erro 'Raw mode is not supported' não acontece no app desktop. Ele é exclusivo da versão de linha de comando rodada dentro do Git Bash. Não passe a impressão de que é um problema do app, ou você assusta o aluno à toa com algo que ele nunca vai ver no caminho que você está ensinando.

04 Planos, preços e cotas de uso

Quando alguém pergunta "quanto custa o Claude?", a resposta honesta é: depende do quanto você vai usar e do que você quer fazer com ele. A Anthropic não vende o Claude como um produto único. Vende como uma assinatura escalonada, parecida com a lógica de um streaming. Existe um plano de entrada barato e planos cada vez mais robustos para quem precisa de mais. A diferença é que aqui o que escala não é "quantos filmes você assiste", e sim quanto trabalho de IA você consegue espremer por dia. Entender essa estrutura é o que separa quem orienta um aluno com segurança ("comece no Pro, é o suficiente para aprender") de quem chuta e deixa o aluno gastar US$ 200 sem precisar, ou pior, travar no Free e achar que o Claude Code "não funciona".

Este capítulo abre a caixa-preta da assinatura. Você vai entender o que cada plano libera, por que o salto do Free para o Pro é o divisor de águas real (e não os planos caros), e por que a Anthropic fala de "uso" de um jeito proposital e meio vago. Esse jeito vago não é falta de transparência: é uma escolha de engenharia que protege tanto a empresa quanto você. No fim, você terá clareza para responder, na frente da câmera, a pergunta que todo iniciante faz ("preciso pagar pra usar isso?") sem enrolar e sem prometer o que não existe.

O degrau que importa: Free não tem Claude Code (e por que isso muda tudo)

Existe uma confusão clássica que você precisa desarmar logo no começo da aula: 'Claude' e 'Claude Code' não são a mesma coisa, e o plano gratuito só te dá o primeiro. No Free (US$ 0), você acessa o chat, aquela janela onde você conversa, faz perguntas, pede um texto. É útil, é de graça, e muita gente acha que isso é 'usar o Claude por completo'. Não é. O Claude Code e o Cowork (o agente que age no seu computador, que é o coração desta aula) simplesmente não estão disponíveis no Free. Se um aluno no plano gratuito clicar para abrir o Code, o próprio app vai mostrar um pedido de upgrade. Ele não 'quebrou' nada. Ele esbarrou no muro do plano.

Isso reposiciona toda a conversa de preço. Para 90% das aulas de chat de IA por aí, o conselho é 'começa no grátis'. Aqui, o conselho honesto é o oposto: para aprender o que ensinamos neste curso, o Free não serve. Ele serve para conhecer o Claude como assistente de texto, e só. O degrau de verdade (o que destrava o agente que mexe em arquivos, cria skills e conecta ferramentas) é o Pro. Por isso eu chamo o Free e o Pro de 'antes e depois': não é uma questão de 'pagar mais para ter um pouco mais'. É a fronteira entre 'olhar a ferramenta pela vitrine' e 'entrar na loja'.

Na prática didática: deixe claro para o aluno que ele PODE testar o chat de graça hoje para se familiarizar com o jeito de conversar com o Claude. Mas avise, sem rodeios, que no minuto em que ele quiser fazer o Claude trabalhar no computador dele (o pulo do gato do curso) vai precisar assinar o Pro. Isso evita a frustração número um do iniciante: instalar tudo, clicar em Code, ver 'faça upgrade' e concluir que 'isso não funciona pra mim'.

O ponto de entrada certo: por que o Pro é onde o aluno deve começar

O Pro custa US$ 20 por mês na assinatura mensal, ou US$ 17 por mês se a pessoa fecha o plano anual (paga o ano de uma vez e sai mais barato por mês). Esses US$ 3 de diferença não são o ponto. O ponto é o que esse plano destrava: ele libera o Claude Code e o Cowork. Em uma frase: o Pro é o plano mais barato que te dá o agente. Tudo que vem acima do Pro não adiciona poderes novos. Adiciona mais espaço para usar os poderes que o Pro já tem.

Para um iniciante de marketing digital, o Pro é mais do que suficiente para a curva de aprendizado inteira deste curso e para os primeiros meses de uso real. Escrever copy, gerar headlines com uma skill, montar uma página de captura em HTML, organizar arquivos de criativos, conectar o Google Drive. Nada disso exige um plano caro. Exige o Pro. Recomendar Max ou superior para alguém que está começando é como mandar quem acabou de tirar a carteira comprar uma picape de mil cavalos: o problema dela não vai ser a potência, vai ser não saber dirigir ainda.

Vale ancorar uma comparação de valor honesta, sem inventar número: US$ 20 por mês é a faixa de um streaming de vídeo razoável. Quando o aluno pesa isso contra ter um assistente que escreve, programa páginas, organiza arquivos e conecta as ferramentas que ele já usa, a conta de custo-benefício se resolve sozinha, desde que ele entenda que está pagando pelo agente, não por um chat melhorzinho. Esse é o enquadramento que você quer plantar: o Pro não é 'a versão paga do chat', é 'o ingresso para o Claude Code'.

Acima do Pro: Max, Team e Enterprise, mais do mesmo, em escala maior

Acima do Pro, a régua sobe em uso, não em recurso. O Max vem em dois tamanhos: US$ 100 por mês (descrito como 5x) e US$ 200 por mês (descrito como 20x). Esse '5x' e '20x' são relativos ao Pro. Significam, em linguagem simples, 'mais ou menos cinco vezes' e 'vinte vezes' mais espaço de uso do que o Pro entrega. Repare que o Max não te dá uma skill secreta nem um conector exclusivo: ele te dá fôlego. É para quem usa o Claude Code o dia inteiro, em projetos pesados, e bate no limite do Pro com frequência. Um iniciante quase nunca está nesse cenário no começo.

O Team é o plano para grupos: US$ 25 por pessoa no mensal (US$ 20 no anual), com um mínimo de cinco pessoas. A lógica aqui é diferente. Você não está comprando 'mais poder', está comprando licenças para uma equipe, com gestão centralizada. É o plano de uma agência ou de um time de marketing que quer todo mundo usando o Claude sob a mesma conta administrada. Já o Enterprise é 'sob consulta': não tem preço de tabela porque é desenhado caso a caso, para empresas grandes, com necessidades de segurança, controle e volume que fogem do padrão de prateleira.

A mensagem que você passa ao aluno é tranquilizadora: existe um caminho de crescimento. Se um dia ele virar uma agência, vira Team. Se ele virar um operador que vive dentro do Claude Code, vira Max. Mas isso é problema do futuro dele, não da primeira semana. Apresentar esses planos serve para mostrar que a ferramenta cresce junto com o uso, e não para pressionar ninguém a gastar mais do que precisa agora. Um bom professor mostra a escada inteira e depois aponta o primeiro degrau.

Cotas de uso: por que ninguém te diz 'você tem 500 mensagens'

Aqui está o conceito que mais confunde o iniciante, e que você vai precisar traduzir com cuidado. A Anthropic não fala 'seu plano dá X mensagens por dia' nem 'você tem Y tokens'. Ela fala de uso de forma relativa: o Pro te dá uma certa quantidade de uso, o Max 5x te dá cerca de cinco vezes isso, e assim por diante. Por que essa vagueza proposital? Porque uma 'mensagem' no Claude não tem tamanho fixo. Pedir 'resuma este parágrafo' consome muito menos do que pedir 'leia estes dez arquivos, entenda o projeto e reescreva a landing page inteira'. Contar mensagens seria injusto. Uma valeria por cinquenta. Então o que o sistema mede, por baixo dos panos, é trabalho, e trabalho não cabe numa contagem redonda de mensagens.

O termo técnico por trás disso é 'token'. Um token é um pedacinho de texto (mais ou menos uma sílaba ou uma palavra curta) e é a unidade que o modelo de fato processa para ler o que você manda e escrever o que responde. Quanto mais texto entra e sai, mais tokens são gastos. Mas a Anthropic deliberadamente NÃO expõe isso como um contador na sua cara, porque para o usuário comum 'você gastou 3.412 tokens' não significa absolutamente nada e só geraria ansiedade. É a mesma razão pela qual seu plano de celular fala em 'internet de sobra para o dia a dia' em vez de te obrigar a calcular megabytes por foto enviada. A medida existe. Ela só não é jogada na sua frente.

O recado prático para o aluno é: você não precisa ficar fazendo conta. Use o Claude com naturalidade. Se você for um usuário comum no Pro (escrevendo copy, criando uma skill, montando uma página de vez em quando) você dificilmente vai bater no teto. As cotas costumam ser renovadas em janelas (ao longo do dia e do período de cobrança), então mesmo que você um dia use muito e veja um aviso de limite, é uma pausa, não um castigo: depois de um tempo, você volta a ter uso disponível. Travar de fato no limite, no Pro, é sinal de uso intenso, e aí, sim, é hora de pensar em Max.

As cotas dobraram em maio de 2026: por que isso importa para a aula

Tem um fato recente que muda a régua e que você precisa carregar na ponta da língua: em 6 de maio de 2026, a Anthropic dobrou as cotas de uso do Claude Code para os planos pagos. Na prática, quem assina Pro, Max, Team ou Enterprise passou a ter aproximadamente o dobro de espaço de uso do Claude Code do que tinha antes dessa data, pelo mesmo preço. O plano Free ficou de fora dessa melhoria, o que é coerente, já que o Free nem tem Claude Code para começar.

Por que isso importa numa aula gravada para o YouTube? Por dois motivos. Primeiro, reforça o argumento de que o Pro é generoso para quem está aprendendo: o limite que já era confortável virou ainda mais confortável. Você pode dizer ao aluno, com base em fato, que o teto subiu recentemente em favor de quem paga. Ele está entrando num momento bom. Segundo, e mais importante para a sua autoridade: esse fato é um lembrete vivo de que números de IA têm prazo de validade. As cotas mudaram em maio. Podem mudar de novo. Por isso o jeito certo de ensinar não é cravar quantidades, é ensinar o princípio (uso relativo, renovável, que cresce com o plano) e checar os detalhes na fonte quando precisar.

Esse cuidado vira uma vantagem de credibilidade. Quando um aluno comentar 'mas eu vi que o limite é tal coisa', você responde com tranquilidade: 'esses números a Anthropic ajusta de tempos em tempos. Em maio de 2026 eles dobraram as cotas dos planos pagos, por exemplo. O que não muda é a lógica: você paga por mais espaço de uso, não por mais recursos.' Isso te posiciona como alguém que entende a mecânica, e não decorou um folheto que pode estar desatualizado amanhã.

Como falar de preço sem mentir: dólar, real e o que você NÃO deve cravar

Todos os preços oficiais do Claude são em dólar americano (US$), e não existe uma tabela oficial em reais. Isso tem uma consequência direta para a sua aula: você não deve cravar 'o Pro custa R$ tanto'. O valor em real depende do câmbio do dia, de IOF, da bandeira do cartão e da operadora, ou seja, varia de pessoa para pessoa e de mês para mês. Se você disser um número fixo em reais e ele estiver errado na semana seguinte, perde credibilidade. O caminho honesto é apresentar em dólar (US$ 20 no Pro, por exemplo) e orientar o aluno a fazer a conversão pelo câmbio do dia para ter a ideia aproximada do que vai cair na fatura.

A mesma disciplina vale para tudo que é quantidade de uso. Nunca diga 'você tem X mensagens' ou 'Y tokens por dia'. Esses números a Anthropic não publica como promessa fixa, justamente porque o uso é relativo e foi ajustado em maio de 2026. Se você inventar um número para parecer mais concreto, está construindo autoridade em cima de areia. O movimento de mestre é o contrário: explicar POR QUE não existe esse número exato (porque cada tarefa pesa diferente) é mais impressionante e mais útil do que fingir que existe.

Fecha com a regra de ouro que protege você em qualquer pergunta de preço ao vivo: dado real ou nada. Você conhece os fatos sólidos: Free é US$ 0 e não tem Code. Pro é US$ 20 (ou US$ 17 anual) e é o ponto de entrada. Max é US$ 100 (5x) e US$ 200 (20x). Team é US$ 25 por pessoa, mínimo de cinco. Enterprise sob consulta. Cotas dobraram em maio/2026 nos pagos. Com esse punhado de fatos verificados você responde 95% das perguntas. Para o resto, a melhor resposta é 'isso eu confiro na página oficial antes de te garantir', e não um chute. Numa aula sobre uma ferramenta que muda rápido, saber o que você NÃO sabe é parte do domínio.

COMO EXPLICAR ISSO PRO ALUNO

- 'O plano grátis te deixa conversar com o Claude, mas ele não te deixa colocar o Claude pra trabalhar no seu computador. Pra isso (que é o que a gente vai aprender aqui) você precisa do plano Pro, que sai uns 20 dólares por mês. Pensa assim: o Free é a vitrine, o Pro é entrar na loja.'
- 'Por que ninguém te fala quantas mensagens você tem? Porque uma mensagem não tem tamanho fixo. Pedir um resumo curtinho gasta pouco. Mandar o Claude ler dez arquivos e refazer uma página inteira gasta muito. Contar mensagem seria injusto. É igual ao seu plano de celular: ele fala internet de sobra, não te obriga a contar cada megabyte.'
- 'Não precisa ficar com medo de estourar o limite no Pro. Usa numa boa. Se um dia você usar MUITO e ele avisar que chegou no limite, não é castigo. É só uma pausa. Depois de um tempo ele renova e você volta a usar. Só quem vive dentro do Claude Code o dia todo é que precisa pensar em plano maior.'
- 'Vou te dar os preços em dólar porque não existe tabela oficial em real. O valor na sua fatura muda conforme o câmbio e o cartão. E esses limites de uso a Anthropic mexe de vez em quando: em maio de 2026, por exemplo, eles dobraram a cota de quem paga. Então eu te ensino a lógica, não decoro número que pode mudar amanhã.'

PONTOS DE ATENÇÃO

- Nunca cravar preço em reais nem dizer 'custa R$ X'. Todos os valores oficiais são em dólar (Pro US$ 20 / US$ 17 anual, Max US$ 100 e US$ 200, Team US$ 25 por pessoa, Enterprise sob consulta) e não há tabela oficial em real. Apresente em dólar e oriente a conversão pelo câmbio do dia.
- Jamais prometer número fixo de mensagens ou tokens. A Anthropic fala de uso de forma RELATIVA de propósito, porque cada tarefa pesa diferente. Dizer 'você tem X mensagens por dia' é invenção e pode estar errado. Fale sempre em 'mais ou menos espaço de uso', com Max 5x e 20x sendo relativos ao Pro.
- Não confundir o que cada plano ADICIONA. Acima do Pro, Max/Team/Enterprise dão mais USO e gestão, não recursos novos. O Claude Code e o Cowork já estão liberados no Pro. Quem está começando não precisa de Max. Recomendar o plano caro a um iniciante é exagero.
- Lembrar que o Free NÃO tem Claude Code nem Cowork: só o chat. Clicar em Code no Free leva a um pedido de upgrade. E que as cotas dos planos pagos dobraram em 6 de maio de 2026 (Free ficou de fora): números de IA têm prazo de validade, então ensine o princípio e reconfira a fonte oficial antes de garantir um detalhe.

05 Instalação a fundo (Windows e Mac)

Instalar um programa parece o passo mais trivial de qualquer treinamento, e justamente por isso é onde mais gente trava e desiste. Numa aula para iniciantes totais, o momento da instalação é o filtro: se você dominar cada tela, cada aviso e cada erro possível, ninguém abandona a call frustrado dizendo "no meu computador não funcionou". Este capítulo existe para que você, Alexandre, conheça o caminho de instalação melhor do que o próprio aluno conhece o computador dele, e consiga diagnosticar qualquer travamento à distância, só pela descrição da tela.

Há um detalhe que muda tudo e que você precisa fixar antes de qualquer coisa: existem duas formas diferentes de usar o Claude Code, e elas têm experiências de instalação e de erro completamente distintas. A aula foca no aplicativo de desktop, a versão com janela, botões e mouse. Mas existe também a versão de linha de comando (o terminal), que é onde mora a maior parte das dores de cabeça que circulam na internet. Quase todo erro assustador que um aluno pode achar no Google pertence à versão de terminal, não à que você vai ensinar. Saber separar as duas é o que te dá autoridade para dizer com calma: "esse erro não vai acontecer com você, ele é de outra versão".

As duas portas de entrada: app desktop (a janela) e CLI (o terminal)

O Claude Code não é um único programa com uma única cara. Ele tem duas portas de entrada, e a confusão entre elas é a raiz de quase todo problema de instalação que você vai encontrar nos fóruns.

A primeira porta é o aplicativo de desktop: um programa com janela, igual ao Word ou ao Chrome. Você clica num ícone, abre uma janela, digita o que quer num campo de texto e usa o mouse para clicar em botões. É a experiência que qualquer pessoa que já usou um computador reconhece de imediato. Esta é a porta que a aula ensina, e é a porta certa para iniciantes, porque não exige aprender nada novo sobre como operar um computador.

A segunda porta é a CLI, sigla de 'Command-Line Interface', ou em português 'interface de linha de comando'. É aquela tela preta (ou de fundo escuro) onde você não clica em nada: digita comandos com o teclado e aperta Enter. No mundo da CLI, para abrir o Claude Code você digita literalmente a palavra 'claude' e dá Enter. É a ferramenta preferida de programadores porque é rápida e poderosa, mas é também onde nascem as mensagens de erro mais intimidadoras para quem está começando.

A analogia que funciona na aula: o app desktop é como dirigir um carro automático. Você senta, liga e anda. A CLI é como dirigir um carro de câmbio manual. Faz a mesma coisa, mas exige que você saiba operar a embreagem. Ninguém aprende a dirigir num manual de corrida. A aula toda vive no automático, e isso é uma escolha consciente, não uma limitação.

O download oficial e por que não se instala Node.js

O endereço de download é um só, e ele precisa estar cravado na sua cabeça: claude.com/download. Tudo começa ali. A página detecta ou oferece a versão certa para o sistema do aluno.

Os sistemas suportados são quatro: macOS (Mac), Windows, Windows ARM (uma variação do Windows que roda em alguns notebooks mais novos com um tipo diferente de processador) e ChromeOS (o sistema dos Chromebooks). Repare no que falta nessa lista: Linux. Não existe versão do app para Linux. Se algum aluno mais técnico perguntar, a resposta honesta é 'o app de desktop não tem versão para Linux', e isso encerra o assunto sem rodeios.

Agora um ponto que poupa você de muita explicação confusa: o Claude Code vem embutido dentro do app. Você não precisa instalar nada além do aplicativo. Existe uma crença, vinda dos tutoriais antigos voltados para programadores, de que é preciso instalar primeiro uma ferramenta chamada Node.js (um motor que faz certos programas rodarem) para depois instalar o Claude Code por cima. Isso valia para a versão de linha de comando. No app de desktop, esquece. O Code já vem dentro, montado e pronto. Se um aluno chegar dizendo 'li que preciso instalar o Node primeiro', você corrige na hora: 'isso é da versão antiga de terminal. No app você só baixa o aplicativo e pronto'.

Windows: o Git é obrigatório e por quê

No Windows existe um pré-requisito que não aparece no Mac e que é a causa número um de travamento na primeira execução: o Git. Você precisa entender o que é e por que ele é exigido, senão vai apenas decorar uma instrução sem saber explicar.

Git é um programa de bastidor, sem janela própria, que serve para controlar versões de arquivos. Ele registra cada alteração feita num projeto, como um histórico de tudo o que mudou. O Claude Code, quando vai trabalhar com seus arquivos no computador, apoia-se nesse maquinário do Git para enxergar e organizar as mudanças. Sem o Git instalado, falta uma peça da engrenagem, e o Code simplesmente se recusa a funcionar no Windows.

O sintoma é direto: ao tentar usar o Claude Code sem o Git, aparece a mensagem 'Git is required' ('o Git é necessário'). O conserto é instalar o Git pelo endereço oficial git-scm.com/install/windows, e (este detalhe é crucial) fechar e reabrir o app do Claude depois de instalar. O app só percebe que o Git existe na próxima vez que abre. Aluno que instala o Git mas não reinicia o aplicativo continua vendo o mesmo erro e jura que 'não funcionou'. A sequência correta é: instalou o Git, fechou o Claude, abriu de novo.

Deixe claro na aula que isso é específico do Windows. É um passo a mais que o pessoal do Mac não precisa dar, e tudo bem, faz parte de como o Windows é montado.

Mac: por que costuma ser mais simples (e como checar)

No Mac, a história do Git é mais tranquila, e vale você saber o motivo para responder com segurança a quem usa Apple na turma.

O Mac geralmente já vem com o Git, ou o instala automaticamente quando você usa certas ferramentas de desenvolvimento. Na prática, a maioria dos usuários de Mac não precisa fazer nada de especial. Baixa o app por claude.com/download e segue em frente. É por isso que o passo do Git aparece como uma exigência forte só no Windows.

Mas 'geralmente' não é 'sempre', então convém saber a verificação. Para checar se o Git está presente num Mac, abre-se o terminal e digita-se 'git --version'. Se aparecer um número de versão, está tudo certo. Se o Mac não tiver o Git, ele costuma oferecer instalar as ferramentas de linha de comando, ou você roda o comando 'xcode-select --install', que baixa esse pacote de ferramentas (Xcode é o ambiente de desenvolvimento da Apple, e essas ferramentas vêm junto). Para a aula, o resumo prático é: 'no Mac normalmente já está tudo pronto. Se por acaso pedir o Git, é um comando só para resolver'. Não transforme isso num drama, para a esmagadora maioria dos alunos de Mac, não vai aparecer.

SmartScreen e o pedido de upgrade no Free: dois sustos que não são erros

Há duas telas que assustam o iniciante durante a instalação mas que não são defeitos. São comportamentos normais. Antecipá-las na aula evita pânico.

A primeira é o SmartScreen do Windows. Quando você executa um programa recém-baixado, o Windows às vezes exibe uma tela azul dizendo algo como 'o Windows protegeu o seu computador'. Parece um bloqueio, parece que algo deu errado, e o iniciante fecha tudo com medo de ter pego vírus. Não é nada disso: é só o Windows sendo cauteloso com qualquer instalador novo. O caminho é clicar em 'Mais informações' e depois em 'Executar assim mesmo'. Pronto, a instalação segue. Mostrar essa tela na aula e dizer 'vai aparecer isso, é normal, clica em Mais informações e Executar assim mesmo' tira o medo antes que ele apareça.

A segunda tela é o pedido de upgrade para quem está no plano Free. Se o aluno instalou o app com uma conta gratuita e clica na função 'Code', o app pede para fazer upgrade, porque o Claude Code não está liberado no plano Free, só nos planos pagos. Isso não é um erro de instalação. É a regra do plano. O aluno fez tudo certo, o programa está instalado corretamente, ele só esbarrou no limite do plano gratuito. (O detalhe de qual plano libera o quê pertence ao capítulo de planos e preços. Aqui basta você saber reconhecer a tela e dizer: 'isso não é bug, é o plano Free pedindo o upgrade para destravar o Code'.)

O erro 'Raw mode is not supported': o bicho-papão que não morde no app

Esse é o erro que mais aparece quando alguém pesquisa Claude Code no Google, e é exatamente o erro que você precisa dominar para projetar autoridade, porque a maioria dos tutoriais explica mal a causa.

A mensagem completa é parecida com 'Raw mode is not supported'. Em português, 'raw mode' é um modo especial em que o teclado entrega cada tecla pressionada diretamente ao programa, sem esperar você apertar Enter. É o que permite a interface interativa do Claude Code funcionar no terminal. O erro diz, em essência: 'este ambiente não oferece o tipo de terminal que eu preciso para funcionar de forma interativa'.

Agora o ponto que separa quem entende de quem só decorou: esse erro acontece SÓ em uma situação muito específica: ao rodar o comando 'claude' (a versão de linha de comando) dentro de um programa chamado Git Bash no Windows. A causa técnica é que o Claude Code usa uma biblioteca de interface chamada Ink para desenhar sua tela no terminal, e o Ink precisa de um terminal de verdade, do tipo TTY (um terminal completo que suporta interação tecla a tecla). O Git Bash não oferece esse tipo de terminal da forma que o Ink espera, então a interface se recusa a iniciar.

Duas conclusões práticas que você leva para a aula. Primeira: esse erro NÃO acontece no app de desktop, nunca. É exclusivo da versão de linha de comando. Como a aula vive no app, seu aluno jamais vai encontrá-lo no caminho que você ensina. Segunda: se mesmo assim alguém quiser usar a versão de terminal e topar com isso, o conserto é trocar de terminal, usar o PowerShell, o Prompt de Comando (cmd) ou o Windows Terminal em vez do Git Bash. Esses três oferecem o terminal completo que o Ink precisa, e o erro desaparece. Saber dar esse diagnóstico de cabeça ('isso é da CLI no Git Bash, troca para o PowerShell') é o tipo de resposta que faz você parecer quem realmente domina a ferramenta.

COMO EXPLICAR ISSO PRO ALUNO

- 'Existem duas portas de entrada para o Claude Code: o aplicativo com janela, que é igual a abrir o Word, e a versão de terminal, aquela tela preta onde você digita comandos. A gente vai usar só a janela. É como dirigir um carro automático em vez de um manual: faz a mesma coisa, mas você não precisa aprender a embreagem.'
- 'No Windows, antes de tudo, você precisa instalar um programinha chamado Git. Pensa nele como uma peça da engrenagem que o Claude usa para mexer nos seus arquivos com segurança. Sem essa peça, ele mostra a mensagem Git is required. Instala pelo site, FECHA o Claude e ABRE de novo. Esse fechar e abrir é o que a maioria esquece.'
- 'Se aparecer uma tela azul dizendo que o Windows protegeu o seu computador, fica calmo: não é vírus, é só o Windows sendo cauteloso com um programa novo. Clica em Mais informações e depois em Executar assim mesmo. Segue normal.'
- 'Aquele erro Raw mode que você talvez veja no YouTube não vai acontecer com a gente. Ele só aparece na versão de terminal, num programa específico chamado Git Bash. Na janela do aplicativo, que é onde a gente trabalha, ele simplesmente não existe. Pode esquecer.'

PONTOS DE ATENÇÃO

- Nunca misture os erros das duas versões. 'Raw mode is not supported' e a exigência de instalar Node.js são problemas da CLI (linha de comando), não do app desktop. Apresentar esses erros como se fossem do app assusta o aluno sem motivo e te faz parecer mal informado.
- O passo 'fechar e reabrir o app' depois de instalar o Git no Windows é obrigatório, não opcional. Omitir isso gera o falso relato de que 'instalei o Git e mesmo assim não funcionou'. O app só reconhece o Git numa nova abertura.
- Não confunda o pedido de upgrade do plano Free com falha de instalação. Quando o usuário Free clica em Code e o app pede upgrade, a instalação está perfeita. É a regra do plano. Tratar isso como erro técnico confunde o aluno.
- Cuidado com a tentação de simplificar dizendo 'instala o Git no Mac também'. No Mac o Git costuma já vir pronto. A exigência forte é do Windows. Achatar a diferença entre os dois sistemas gera instrução errada para metade da turma. E lembre que não há versão do app para Linux. Se perguntarem, seja direto.

06 Como o Claude age no seu PC com segurança

Aqui mora a parte que mais assusta o iniciante e que, bem explicada, mais constrói autoridade: "deixar uma inteligência artificial mexer no meu computador". A reação instintiva é recuar. Mas a verdade técnica é o contrário do medo: o Claude Code foi desenhado para que você NÃO precise confiar cegamente. Ele não age às escondidas. Cada passo que toca seus arquivos passa por uma porta com cadeado, e a chave é sua. O modelo propõe, mostra exatamente o que pretende fazer, e só executa depois que você clica em "Aceitar". Sem o seu clique, nada acontece de irreversível.

Este capítulo destrincha esse mecanismo de proteção por dentro: o ciclo de trabalho de um agente (entender, planejar, mostrar, esperar aprovação), o sistema de permissões, a tal "caixa isolada" do Cowork, e a fronteira clara entre o que o Claude faz sozinho e o que exige a sua autorização. O objetivo é que você termine capaz de olhar um aluno nos olhos e dizer, com fundamento técnico: "experimentar é seguro, e eu vou te mostrar por quê".

O loop do agente: por que ele para e espera você

No Capítulo 3 você viu que um agente não só responde. Ele PLANEJA e EXECUTA vários passos. O que esse capítulo acrescenta é a mecânica de segurança dentro desse ciclo. Pense no Claude Code como um cozinheiro experiente que você contratou, mas que cozinha na SUA cozinha. Antes de jogar qualquer ingrediente na panela, ele para e diz: 'vou usar estes três ovos e este leite, tudo bem?'. Só depois do seu 'pode ir' é que ele mexe.

Na prática, o ciclo tem quatro estágios. Primeiro, ele ENTENDE o pedido: você escreve em português normal o que quer ('organize os criativos desta pasta por campanha'). Segundo, ele monta um PLANO: frequentemente ele lista, em texto, os passos que pretende dar antes de dar o primeiro. Terceiro, e este é o coração da segurança, ele MOSTRA o que vai mudar antes de mudar. Quarto, ele ESPERA a sua decisão: Aceitar ou Rejeitar. Enquanto você não responde, o trabalho fica congelado naquele ponto.

O detalhe que vale gravar: esse 'parar e perguntar' não é uma cortesia opcional do Claude. É o comportamento padrão do programa para qualquer ação que mexa de verdade no seu computador. Ele foi construído para interromper o próprio fluxo nesses pontos. Você não precisa lembrar de ativar nada. A pergunta vem sozinha.

O diff: a 'prévia' que mostra exatamente o que vai mudar

O termo técnico é diff: abreviação de 'difference', diferença em inglês. É a forma como o Claude te mostra, lado a lado, o ANTES e o DEPOIS de qualquer alteração, antes de ela acontecer. Pense no 'controlar alterações' do Word, aquele modo em que o texto novo aparece destacado e o texto removido aparece riscado. O diff é exatamente isso, só que para qualquer arquivo: você vê em verde o que será adicionado e em vermelho o que será apagado.

Isso muda tudo do ponto de vista de confiança. Você não autoriza no escuro. Suponha que você peça: 'corrija o título desta página de captura de Harmonização Facial para Harmonização Facial em Goiânia'. O Claude não troca o texto e avisa depois. Ele te mostra: a linha antiga (riscada, em vermelho) e a linha nova (destacada, em verde). Você lê com seus próprios olhos, confere se ele entendeu, e só então clica em Aceitar. Se ele tiver entendido errado (trocou a cidade errada, mexeu numa linha que não devia) você clica em Rejeitar e nada foi alterado no arquivo real.

Para o iniciante, a frase que dissolve o medo é esta: o diff é uma maquete, não a obra. Enquanto você está olhando a maquete, a casa de verdade continua intacta. A construção só começa no clique de Aceitar. E mesmo que algo passe, arquivos editados costumam ter histórico recuperável, mas o diff existe justamente para você não chegar nesse ponto: o erro é barrado na porta, antes de entrar.

O sistema de permissões: o cadeado fica do seu lado

Aceitar e Rejeitar são o nível mais visível, mas por baixo existe um sistema de permissões mais granular. A ideia central: o Claude pede autorização proporcional ao risco da ação. Ler um arquivo para entender o que tem dentro é uma ação de baixo risco. É como abrir uma gaveta e olhar. Modificar, apagar ou rodar um comando que muda o estado do computador é alto risco. É mexer no conteúdo da gaveta. Quanto maior o risco, mais explícita a autorização que ele exige.

Na hora em que ele pede para fazer uma ação sensível, você costuma ter mais de uma resposta possível, não apenas sim ou não para aquela vez. Por exemplo: você pode autorizar SÓ AQUELA ação específica, ou autorizar aquele tipo de ação para o resto da conversa (útil quando você já confia e não quer clicar 'sim' trinta vezes para a mesma tarefa repetitiva), ou negar. A lógica é que VOCÊ calibra o nível de liberdade conforme a sua confiança vai crescendo, em vez de o programa decidir isso por você.

Uma analogia útil para ensinar: é como dar a chave de casa a um prestador de serviço. No primeiro dia você abre a porta para ele a cada vez. Depois de ganhar confiança, talvez você deixe a chave para ele entrar sozinho na área específica onde ele trabalha, mas NUNCA no cofre. O Claude funciona assim: você abre as portas no seu ritmo, e as portas que você nunca abriu permanecem fechadas.

A caixa isolada do Cowork: brincar com rede de proteção

O Cowork (que você viu no Capítulo 3 como 'o poder do Code num formato para não-técnicos') roda dentro de uma caixa isolada. O nome técnico dessa caixa é máquina virtual, ou VM. Vale entender o conceito porque ele é o que torna o Cowork tão seguro para um leigo experimentar.

Uma máquina virtual é, na prática, um 'computador de mentira dentro do seu computador', um ambiente separado, com paredes próprias, onde o trabalho acontece sem encostar diretamente no resto da sua máquina. Imagine uma sala de testes envidraçada dentro de uma fábrica: o que acontece lá dentro fica contido lá dentro. Se um experimento der errado naquela sala, a fábrica inteira continua funcionando. É essa separação que dá a sensação (justificada) de que você pode pedir coisas ousadas sem medo de 'quebrar o computador'.

Dentro dessa caixa, o comportamento padrão continua sendo o 'pedir antes de agir': o Cowork não toma decisões de impacto sem te consultar. A combinação é poderosa para iniciante: caixa isolada (o estrago fica contido) somada ao pedido de aprovação (o estrago raramente chega a acontecer). É a diferença entre aprender a dirigir num pátio vazio com o instrutor de pé no freio, e ser jogado direto na avenida. O Cowork é o pátio vazio com instrutor.

A fronteira: o que ele faz sozinho e o que sempre pede

Para ensinar com autoridade, você precisa traçar a linha com clareza, sem exagerar a liberdade do Claude nem assustar à toa. De um lado da linha estão as ações de leitura e raciocínio: ler o conteúdo de um arquivo que você apontou, analisar, pensar, redigir um texto de resposta, montar um plano. Isso ele faz no fluxo normal porque não altera nada do seu lado. É o equivalente a pensar em voz alta.

Do outro lado estão as ações que MUDAM algo: escrever ou sobrescrever um arquivo, apagar, mover, rodar um comando no sistema, ou agir através de um conector para fora do seu computador (mandar um e-mail, criar um evento na agenda). Essas atravessam a porta da permissão. Mesmo quando você já autorizou um tipo de ação para a conversa toda, ações de maior impacto tendem a reabrir a pergunta. O programa erra para o lado da cautela.

O princípio de fundo, que conversa com a aposta da Anthropic em IA segura e confiável (Capítulo 2), é simples: o padrão é a contenção. Quando há dúvida sobre se uma ação pode causar dano, o comportamento projetado é PERGUNTAR, não presumir. Isso significa que o pior caso realista de um iniciante curioso não é 'destruí meus arquivos'. É 'cliquei em Aceitar numa mudança que eu não tinha lido direito'. E é por isso que a única disciplina que você precisa ensinar é: leia o diff antes de aceitar. Faça isso, e a margem de erro vira quase zero.

COMO EXPLICAR ISSO PRO ALUNO

- Pensa no Claude como um pedreiro de altíssima confiança que mostra a maquete antes de levantar a parede. Ele desenha exatamente o que vai construir, você olha, e só quando você fala 'pode tocar' é que ele pega na ferramenta. Enquanto você não aprova, sua casa continua exatamente do jeito que estava.
- Sabe o 'controlar alterações' do Word, que deixa o texto novo destacado e o texto apagado riscado? O Claude faz isso com qualquer arquivo, e (esse é o pulo do gato) ANTES de mudar de verdade. Você vê o verde do que ele quer adicionar e o vermelho do que ele quer tirar, lê com calma, e aí decide. Errou? Você clica em Rejeitar e nada aconteceu.
- Tem dois botões que você vai apertar o tempo todo: Aceitar e Rejeitar. Esse é todo o segredo da segurança. O computador não faz nada irreversível sem o seu dedo no Aceitar. Então não dá para 'estragar sem querer', porque estragar exige a sua autorização ativa, e ela vem sempre depois de ele te mostrar o que pretende fazer.
- O Cowork roda numa espécie de 'computador de mentira dentro do seu computador', uma sala de vidro à prova de bagunça. Se um teste der errado lá dentro, fica contido lá dentro. O resto da sua máquina nem sente. É por isso que dá pra ousar e pedir coisas que você não faria à mão: tem rede de proteção embaixo.

PONTOS DE ATENÇÃO

- Não venda 'totalmente automático e sem riscos'. A segurança vem justamente de ele NÃO ser totalmente automático: ele para e pede aprovação. Se um aluno achar que é mágica que faz tudo sozinho sem supervisão, ele vai clicar em Aceitar no automático sem ler, e aí, sim, pode causar uma mudança indesejada. A disciplina-chave a ensinar é: leia o diff antes de aceitar.
- Cuidado para não prometer 'desfaz qualquer coisa com um botão mágico'. A proteção principal é PREVENTIVA (o diff e a aprovação barram o erro na porta), não corretiva. Edições de arquivo costumam ter histórico recuperável, mas nem toda ação é trivialmente reversível. Por isso o foco é não deixar o erro acontecer, e não confiar num 'desfazer' universal que pode não existir para certos casos.
- Não confunda o sistema de permissões/diff (do Claude Code e do Cowork, agindo no SEU computador, foco deste capítulo) com a autorização de um conector (que dá acesso a um app externo como Gmail ou Drive). São camadas de segurança diferentes: a permissão local e o login do conector. Os detalhes de conectar contas externas e quais permissões você concede ficam no Capítulo 7. Aqui o foco é o que acontece dentro da sua máquina.
- A 'caixa isolada / VM' é uma característica associada ao Cowork. Não generalize dizendo que tudo que o Claude faz roda numa máquina virtual blindada. O que é universal e vale para todos os formatos é o modelo de pedir aprovação antes de agir e mostrar o diff. Apresente a VM como o reforço extra do Cowork, não como uma muralha presente em toda e qualquer interação.

07 MCP e Conectores: como o Claude conversa com os seus outros aplicativos

Até aqui o Claude trabalhou sozinho, com o que você digita na conversa. O problema é que o seu trabalho de verdade não mora dentro de uma janela de chat: ele mora no seu Gmail, no seu Google Drive, no Notion onde está o calendário de conteúdo, no Canva onde estão os criativos, na planilha de campanha. Um assistente que não enxerga nada disso te obriga a virar "carregador de baldes": copiar daqui, colar ali, exportar, anexar. É trabalhoso e, pior, é onde o erro entra. Os conectores existem para derrubar essa parede: eles dão ao Claude uma porta autorizada para ler e agir dentro dos aplicativos que você já usa, sem você ficar no meio do caminho transportando informação na mão.

Este capítulo te dá duas coisas que valem ouro na hora de ensinar e na hora de responder pergunta difícil. A primeira é entender de verdade o que é o MCP (a tecnologia por baixo dos conectores) para você nunca tropeçar na sigla diante de um aluno. A segunda, e mais delicada, é a verdade honesta sobre o que dá e o que não dá para conectar. Aqui é onde a maioria dos "cursos de IA" mente, prometendo "WhatsApp e Instagram em um clique". Você não vai mentir, e é justamente por não mentir que você vira a autoridade do assunto. Honestidade técnica é a sua vantagem competitiva neste tema.

O que é MCP, em português de gente: a tomada universal da IA

MCP é a sigla de Model Context Protocol, em tradução solta, "Protocolo de Contexto do Modelo". A palavra que importa ali é protocolo, e ela significa apenas isto: um conjunto de regras combinadas para que duas coisas conversem sem mal-entendido. O alfabeto Morse é um protocolo. As regras de uma tomada de parede são um protocolo: qualquer aparelho com o pino certo encaixa e funciona, não importa a marca. O MCP é exatamente isso, só que para ligar a inteligência artificial aos seus aplicativos.

Vale guardar a analogia da tomada, porque ela explica por que isso é importante e não só um detalhe técnico. Antes de existir um padrão de tomada, cada eletrodoméstico precisaria de uma ligação elétrica feita sob medida, um eletricista para cada aparelho. Caro, lento, e cada fabricante reinventava a roda. Com a tomada padronizada, a Brastemp e a Electrolux fazem a geladeira. Quem cria a tomada não precisa saber nada de geladeira. Foi isso que o MCP fez para a IA: a Anthropic publicou o "formato da tomada" e o abriu para qualquer empresa usar (por isso se diz que é um padrão aberto). Aí o Google pôde construir o "plugue" do Gmail, o Notion o do Notion, a Stripe o da Stripe, cada um cuidando do próprio aplicativo, todos encaixando no mesmo Claude.

Na prática, para você e para o seu aluno, MCP é a engrenagem invisível embaixo do capô. Ninguém precisa saber a sigla para usar, assim como ninguém precisa entender de eletricidade para plugar a TV. O nome que aparece para o usuário é "conector" (em inglês, connector). Sempre que neste material você ler "conector", saiba que por baixo dele está rodando o MCP. Quando um aluno mais curioso perguntar "o que é esse tal de MCP?", a resposta de uma frase é: "é o padrão técnico que faz o Claude conseguir falar com seus outros programas, do mesmo jeito que a tomada faz qualquer aparelho funcionar na parede".

Como um conector funciona por dentro: leitura, ação e o login que você autoriza

Um conector faz duas categorias de coisa, e separá-las ajuda muito na hora de explicar e na hora de avaliar risco. A primeira é ler: o Claude passa a enxergar o que existe lá dentro do aplicativo, os e-mails da sua caixa, os arquivos da sua pasta no Drive, as páginas do seu Notion. A segunda é agir: o Claude pode criar, alterar ou mover coisas, rascunhar um e-mail, montar uma página nova, atualizar uma tarefa. Nem todo conector faz as duas. O do HubSpot, por exemplo, é só leitura: o Claude consulta o seu CRM (contatos, negócios, empresas) mas não mexe em nada. Já o do Stripe lê e escreve. Saber dessa diferença evita expectativa errada. Você não vai prometer que o Claude "organiza seu HubSpot" quando ele, na verdade, só consegue olhar.

O passo a passo para ligar um conector é curto e é o mesmo para todos. Dentro de uma conversa no Claude, você clica no botão de mais (o "+"), escolhe Connectors, encontra o aplicativo na lista e clica para conectar. Aí abre a tela de login do próprio aplicativo, a tela do Google, da Notion, do Slack, a verdadeira deles, não uma do Claude. Você faz login ali e o aplicativo te mostra exatamente quais permissões está concedendo ("este app poderá ler e enviar e-mails em seu nome", por exemplo). Você aprova, e pronto: a partir dali, naquela conversa, o Claude consegue usar aquele aplicativo. É o mesmo fluxo de quando você entra num site novo clicando em "Continuar com o Google". Você reconhece a tela, sabe que é segura, e é o Google que decide o que liberar, não o site.

Um detalhe que tranquiliza o aluno: você nunca entrega a sua senha ao Claude. O login acontece na casa do aplicativo, e o que volta para o Claude é uma espécie de "crachá de visitante" com permissões limitadas, não a sua chave-mestra. E esse crachá pode ser cancelado a qualquer momento, tanto desconectando dentro do Claude quanto revogando o acesso nas configurações de segurança da sua conta Google, Microsoft, etc. Ensine isso: conectar não é um caminho sem volta.

O panorama real: centenas de conectores oficiais, e o que "oficial" significa

O catálogo oficial fica em claude.com/connectors, e o número certo de dizer é "centenas de conectores". Resista à tentação de cravar um número exato. Isso muda toda semana e te deixa exposto a estar errado na frente da turma. "Centenas" é verdadeiro, é impressionante o suficiente, e não envelhece.

A palavra-chave aqui é oficial, e ela tem um peso concreto. Conector oficial quer dizer que foi a própria empresa dona do aplicativo que construiu e mantém aquele plugue. O conector do Gmail foi feito pelo Google. O do Notion, pela Notion. O do Slack, pela Slack. Isso importa por dois motivos: confiabilidade (quem fez é quem entende do produto e atualiza quando o produto muda) e segurança (você está autorizando o acesso para a empresa dona, não para um terceiro estranho). Os oficiais aparecem com o nome e o logo do aplicativo no catálogo, e ligam com aquele fluxo de um clique e login que descrevemos.

Para você ter um repertório de exemplos na ponta da língua, vale agrupar por uso de marketing. Google Workspace é o carro-chefe: o conector do Google Drive lê e salva seus arquivos, e é por dentro dele que Google Docs e Planilhas funcionam (não existe um botão separado de "Google Docs", isso confunde muita gente). Somam-se o Gmail e o Google Agenda. Para conteúdo e design: Canva (cria, preenche e exporta designs por instrução) e Notion (páginas, tarefas, bases de conhecimento). Para operação e time: Slack, Asana, Linear. Para quem mexe com vendas e dinheiro: HubSpot (consulta de CRM, só leitura), Stripe (clientes e pagamentos), além de GitHub, Microsoft 365, Airtable e muitos outros. Todos esses ligam em qualquer plano. Uma observação de plano que cai em prova: no Free, você pode ter no máximo um conector customizado seu (os do catálogo não contam nesse limite). E em contas de Team e Enterprise, é o administrador que precisa liberar os conectores antes. Então, se um aluno de empresa disser que não aparece nada, o caminho é falar com o admin, não é defeito.

A verdade sobre Meta Ads, Instagram, Facebook e WhatsApp: o ponto onde os outros cursos mentem

Este é o trecho mais importante do capítulo para a sua autoridade, porque é exatamente onde "curso de IA de internet" promete o que não existe. Para um público de marketing digital, a primeira pergunta vai ser: "dá para postar no Instagram e mandar no WhatsApp direto pelo Claude?". A resposta honesta vende mais do que a fantasia. Vamos por partes, porque cada plataforma tem um status diferente.

Meta Ads tem, sim, um conector oficial da própria Meta, em mcp.facebook.com/ads, lançado em 29 de abril de 2026 e ainda em fase de testes aberta (o que o pessoal de tecnologia chama de beta: versão liberada ao público mas sob aperfeiçoamento). E aqui está a pegadinha que separa quem domina de quem repete: esse conector serve só para anúncios pagos. Ele cria, gerencia e analisa campanhas (útil de verdade para tráfego) mas não posta nada orgânico. Então quando você falar de Meta no Claude, seja cirúrgico: é para a parte de mídia paga, não para a página da marca. Duas ressalvas práticas, porque é beta: está gratuito durante o beta, mas a Meta não anunciou preço futuro. E vale confirmar se já está liberado no Brasil antes de prometer numa demonstração ao vivo.

Postar conteúdo orgânico no Instagram e no Facebook (aquele post de feed, o Reels, o carrossel) não tem conector oficial. Ponto. Se alguém quiser automatizar isso com o Claude, o único caminho é via ferramentas de terceiros (Make, Zapier, n8n) ou conectores feitos pela comunidade, que não são oficiais, não são um clique e exigem montagem técnica. WhatsApp é o caso mais delicado de todos: não existe conector oficial nenhum. O endereço claude.com/connectors/whatsapp dá erro 404, ou seja, a página simplesmente não existe. Qualquer integração de WhatsApp com IA hoje passa por terceiros usando API não oficial, e isso carrega risco real de bloqueio da conta. Jamais venda "WhatsApp no Claude em um clique". A frase de autoridade, que você pode repetir tal qual: "WhatsApp não tem conector oficial. Só dá via terceiros, que exigem configuração e têm risco de banir o seu número". Dizer isso te posiciona como o profissional sério da sala, não como mais um vendedor de milagre.

Segurança ao conectar: o que você está autorizando, de verdade

Conectar é dar uma chave da sua casa. Não é motivo para pânico. É motivo para consciência. A regra mental que vale ensinar é a de qualquer permissão digital: antes de clicar em autorizar, leia o que a tela está pedindo. Quando o login do aplicativo abre, ele lista os poderes que vai conceder: "ler seus e-mails", "enviar e-mails em seu nome", "acessar todos os arquivos do seu Drive". Essa tela não é burocracia para passar batido. É o contrato. Se um conector de leitura de e-mail pedir permissão para apagar e-mails, isso é estranho e merece um pé atrás. Para os conectores oficiais das grandes (Google, Microsoft, Slack), você pode confiar no fluxo, mas o hábito de ler antes de aprovar é o que separa quem usa com segurança de quem se complica.

Vale também entender o alcance da permissão. Um conector ligado vale para o seu Claude. Ele passa a poder usar aquele aplicativo nas suas conversas. Por isso a recomendação é conectar só o que você realmente vai usar, e desconectar o que parou de usar. Menos portas abertas, menos superfície de risco. E lembre que o controle não fica preso dentro do Claude: a sua conta Google ou Microsoft tem uma área de "aplicativos conectados" ou "acesso de terceiros", e dali você pode revogar a permissão do Claude a qualquer momento, mesmo sem abrir o Claude. É o equivalente a trocar a fechadura: o crachá que você deu para de funcionar na hora.

Um ponto que diferencia conector de skill (que você vê no próximo capítulo) e que ajuda a montar o quadro mental: o conector liga o Claude a um serviço externo com a permissão e o login daquele serviço. Quem guarda os dados e impõe os limites é o Google, o Notion, a Stripe. Em ambiente de empresa isso fica ainda mais controlado, porque o administrador decide quais conectores existem e dados de contas de time não são usados para treinar os modelos por padrão. Para o seu aluno autônomo, a mensagem final é simples e honesta: conector é seguro de usar desde que você ligue os oficiais, leia o que autoriza e desligue o que não usa mais.

COMO EXPLICAR ISSO PRO ALUNO

- "Sabe a tomada da parede? Qualquer aparelho com o pino certo encaixa e funciona, não importa a marca. O MCP é a tomada da inteligência artificial: é o padrão que faz o Claude encaixar nos seus programas. Você não precisa saber o nome dele, igual você não precisa entender de eletricidade para plugar a TV. O que aparece para você na tela chama-se conector, e por baixo de todo conector está rodando esse tal de MCP."
- "Ligar um conector é o mesmo gesto de quando você entra num site novo clicando em 'Continuar com o Google'. Dentro da conversa você clica no '+', escolhe Connectors, acha o app e faz login na tela verdadeira dele. Você nunca dá a sua senha pro Claude. O login acontece na casa do Google, e o que o Claude recebe é um crachá de visitante com permissão limitada, que você pode cancelar quando quiser."
- "Vou ser honesto com vocês, porque é aqui que muito curso de IA mente: NÃO dá para postar no Instagram nem mandar no WhatsApp em um clique pelo Claude. Não existe conector oficial pra isso. O único oficial da Meta é o de Meta Ads, e ele serve só pra anúncio pago, cria e analisa campanha, mas não posta nada. WhatsApp então: zero conector oficial, só dá via terceiros e com risco de banir o número. Quem promete isso fácil está te enrolando."
- "Antes de clicar em 'autorizar', leia o que a tela está pedindo. Ela é o contrato. Se um conector que só deveria LER seus e-mails pedir permissão pra APAGAR e-mails, desconfie. E grave: conectar não é caminho sem volta. Na sua conta do Google, em 'aplicativos conectados', você corta o acesso do Claude na hora que quiser, igual trocar a fechadura."

PONTOS DE ATENÇÃO

- Nunca cravar um número exato de conectores. O termo verificado e seguro é "centenas de conectores oficiais". O catálogo muda toda semana. Um número específico te deixa exposto a estar errado na frente da turma e a envelhecer o material.
- Meta Ads é oficial MAS só para anúncios pagos, e está em beta. Não confunda com postar orgânico (que não tem conector oficial). Detalhes a manter: lançado em 29/04/2026, em mcp.facebook.com/ads, gratuito no beta com preço futuro não anunciado, e convém confirmar disponibilidade no Brasil antes de qualquer demonstração ao vivo. Não prometer que "o Meta no Claude posta na página".
- WhatsApp não tem conector oficial nenhum: claude.com/connectors/whatsapp retorna erro 404 (página inexistente). Só existe via terceiros com API não oficial, e isso carrega risco real de bloqueio de conta. Jamais vender "WhatsApp em um clique". A integridade nesse ponto é o que sustenta a autoridade do apresentador.
- Não há um conector separado de Google Docs ou Planilhas. Eles funcionam por dentro do conector do Google Drive. Errar isso gera confusão prática (o aluno procura um botão que não existe). E em contas Team/Enterprise, é o administrador quem libera os conectores: se não aparecem, o caminho é o admin, não é defeito do app.

08 Skills por dentro

Se um conector liga o Claude ao mundo lá fora (capítulo 7), uma skill faz o contrário: ensina o Claude a trabalhar do SEU jeito, sem você precisar repetir a mesma explicação toda vez. Pense numa skill como uma receita escrita que você entrega ao Claude uma única vez. A partir daí, sempre que você pedir aquele prato, ele já sabe os ingredientes, a ordem dos passos e o ponto certo, sem que você dite tudo de novo. O nome técnico do recurso é "skill" (habilidade), e o ponto que mais surpreende quem vem do mundo da programação é que uma skill não tem nenhuma linha de código. É um arquivo de texto, escrito em português, com instruções. Quem sabe escrever um bom briefing já sabe, no fundo, escrever uma skill.

Para você, Alexandre, isso muda o jogo na aula. O aluno iniciante não vai "programar" nada. Vai escrever, que é exatamente o que ele já faz o dia inteiro. Este capítulo abre o capô: o que é o arquivo SKILL.md por dentro, como o Claude descobre e dispara uma skill quando você digita uma barra, como criar uma do zero e instalar, qual é o cuidado de segurança que você nunca pode pular, e (no fim) uma dissecação completa da skill "Gerador de Headlines" que vocês vão usar ao vivo, para você explicar por que ela entrega 10 ângulos e por que funciona tão bem.

A anatomia de uma skill: um arquivo SKILL.md, duas partes

Uma skill é um único arquivo chamado SKILL.md (o ".md" indica Markdown, um jeito simples de escrever texto com títulos e listas usando só sinais como # e *). Esse arquivo tem duas partes bem separadas, e entender essa divisão é a chave de tudo.

A primeira parte é o frontmatter (cabeçalho), escrito num formato chamado YAML. É um bloco no topo, delimitado por três tracinhos (---) em cima e embaixo, com apenas dois campos obrigatórios: name (o nome da skill) e description (a descrição do que ela faz e quando deve ser usada). É a etiqueta do produto na prateleira: curta, padronizada, feita para ser lida rápido.

A segunda parte é o corpo: tudo que vem depois do cabeçalho. São as instruções de verdade, escritas em português corrido: quem é o especialista, o que ele faz, em que ordem, em que formato entrega, o que é proibido. É a receita completa, com os passos e os truques do chef.

A analogia mais honesta é a de um funcionário com um manual de procedimento. O cabeçalho é o crachá e o cargo ("Copywriter: chamo quando o assunto é título de anúncio"). O corpo é o treinamento que você deu a ele ("primeiro colete estas 4 informações. Depois escreva nesta ordem. Nunca use clichê"). Você escreve esse manual uma vez. O Claude o segue todas as vezes. E como é texto puro, dá para abrir no Bloco de Notas e ler de cabo a rabo. Guarde isso, porque é o coração da parte de segurança mais adiante.

Como o Claude acha e dispara a skill quando você digita /nome

Aqui está a parte que parece mágica mas tem uma mecânica simples por trás. Existem dois caminhos para acionar uma skill, e vale ensinar os dois.

O caminho explícito é o da barra. Você digita uma barra seguida do nome da skill (por exemplo /gerador-headlines) e o Claude carrega aquele manual e passa a seguir aquelas instruções na conversa. O nome que vem depois da barra é exatamente o campo name do cabeçalho. É como chamar o funcionário pelo nome do crachá: direto, sem ambiguidade.

O caminho automático é mais sutil e é onde o campo description ganha importância. O Claude lê a descrição de todas as skills disponíveis e, quando você faz um pedido em linguagem natural que combina com aquela descrição, ele pode acionar a skill sozinho. Por isso a descrição não diz só o que a skill faz. Diz quando usá-la, muitas vezes listando os "gatilhos", as frases típicas. Se a descrição inclui "use quando o usuário pedir headlines, títulos de anúncio, chamadas", então quando você escrever "me dá uns títulos pro meu criativo" o Claude reconhece o pedido e chama a skill por conta própria, mesmo sem a barra.

Resumo da mecânica para o aluno: a barra é o atalho que VOCÊ controla. A descrição é o que deixa o CLAUDE controlar. Uma descrição preguiçosa ("gera headlines") quase só funciona pela barra. Uma descrição caprichada, recheada de gatilhos, faz a skill aparecer na hora certa sem você lembrar dela. Esse é o motivo de a description ser, na prática, o campo mais estratégico do arquivo inteiro.

Criar uma skill do zero: você só precisa saber escrever

Criar uma skill é escrever um arquivo de texto e salvá-lo com o nome SKILL.md. Não há compilação, não há instalação de programa, não há código. Vou dar a receita mínima que você pode mostrar ao vivo.

Passo 1: o cabeçalho. Abra com os três tracinhos, escreva o name (use letras minúsculas e hífens, sem espaço nem acento, por exemplo roteiro-reels) e a description, e feche com outros três tracinhos. Capriche na descrição: diga o que faz E quando usar, com frases-gatilho. Exemplo de cabeçalho:

---
name: roteiro-reels
description: Cria roteiro de Reels de 30 a 45 segundos com gancho, desenvolvimento e CTA, a partir do tema e do público. Use quando o usuário pedir roteiro de Reels, vídeo curto, script pra Instagram ou TikTok.

Passo 2: o corpo. Logo abaixo, escreva as instruções como se estivesse treinando um redator novo. Diga quem ele é ("você é um roteirista de vídeos curtos"), o que coletar antes de começar, os passos na ordem, o formato da entrega e as proibições ("sem clichê, sem gancho genérico"). Quanto mais específico, melhor a saída. Vaguidão no manual vira vaguidão no resultado.

A regra de ouro que vale repetir na aula: uma skill é só a sua melhor explicação, escrita uma vez. Se você consegue ensinar um estagiário a fazer a tarefa, você consegue escrever a skill. A diferença é que o estagiário esquece e o arquivo não.

Instalar a skill: dois caminhos, e o que muda entre eles

Depois que o arquivo existe, ele precisa entrar no Claude. Há dois jeitos oficiais, e funciona em todos os planos, Free incluído.

Caminho A: pela tela (mais amigável para iniciante). Vá em Customize, depois Skills, e faça upload de um arquivo .zip. O .zip (arquivo compactado, aquela "pastinha zipada" que todo mundo já recebeu por e-mail) deve conter o SKILL.md. É o caminho que não exige mexer em pasta nenhuma do computador: clica, sobe, pronto. Para a aula, este é o caminho que eu mostraria primeiro, porque é visual e não assusta.

Caminho B: pela pasta (mais direto para quem não tem medo de pasta). Existe uma pasta no seu computador, ~/.claude/skills/ (o ~ significa a sua pasta de usuário). Dentro dela você cria uma subpasta com o nome da skill e coloca o SKILL.md ali: ~/.claude/skills/roteiro-reels/SKILL.md. O Claude varre essa pasta e encontra a skill sozinho. É o jeito que a maioria dos profissionais acaba usando, porque editar o arquivo direto na pasta é mais rápido do que zipar e subir de novo a cada ajuste.

A diferença prática: o caminho A é "sobe um pacote pronto". O caminho B é "deixa o arquivo morando numa pasta que o Claude vigia". Para distribuir uma skill para outra pessoa, o .zip é o formato natural. Foi assim que a skill da aula foi empacotada. Para você mesmo iterar e melhorar, a pasta ganha. Mostre o A no palco pela simplicidade. Mencione o B para quem quiser ir além.

Segurança: trate uma skill como anexo de e-mail

Esta é a frase que eu colocaria num slide sozinho: skill é como anexo de e-mail. Você não abre um anexo de um remetente que não conhece, e não instala uma skill de fonte que você não confia. O motivo é direto: uma skill dá instruções que o Claude vai seguir, e um agente que segue instruções pode executar tarefas no seu computador (lembre do capítulo 6, em que o Claude age com permissões). Uma skill mal-intencionada é, no fundo, um conjunto de ordens que alguém escreveu para o seu Claude obedecer.

A boa notícia, e o ponto que tranquiliza o aluno, é que uma skill não tem nada escondido. É texto puro. Você pode abrir o SKILL.md no Bloco de Notas antes de ativar e ler exatamente o que ele manda o Claude fazer. Se o manual diz "gere 10 headlines num ângulo diferente cada", é só isso que ele faz. Se um arquivo dissesse algo estranho como "apague arquivos" ou "copie dados e envie para tal lugar", você veria em português, na sua frente, antes de qualquer coisa acontecer. Anexo de e-mail que você consegue ler inteiro antes de abrir já é muito mais seguro do que a média.

A regra prática de três passos para a aula: (1) Só instale skill de fonte confiável, sua, de alguém que você conhece, ou de catálogo oficial. (2) Antes de ativar uma skill de terceiros, abra e leia o SKILL.md. (3) Na dúvida, não instale. Como você usa as próprias skills e as da aula, o risco no dia a dia é baixo, mas o aluno precisa sair com o hábito de olhar o anexo antes de abrir.

Dissecando a "Gerador de Headlines": por que 10 ângulos e por que funciona

Agora a estrela da aula, aberta por dentro. O cabeçalho dela tem name: gerador-headlines e uma descrição que lista os gatilhos, "gera headlines", "títulos pro meu anúncio", "copy pro criativo". É por isso que ela aparece tanto pela barra /gerador-headlines quanto sozinha, quando você só pede títulos para um anúncio.

O corpo faz quatro coisas que valem destacar no palco. Primeira: ele define um papel, "você é um copywriter de resposta direta especialista em anúncios que param o scroll". Dar um papel ao Claude muda o nível da saída, porque ele passa a escrever como aquele especialista. Segunda: antes de produzir, a skill manda coletar 4 informações (produto, público, dor ou desejo, e oferta ou diferencial) numa única pergunta curta, e se faltar algo ela autoriza "suposições explícitas" marcadas, para nunca travar a entrega. Isso resolve o maior erro do iniciante, que é pedir headline sem dar contexto.

Terceira, e o coração da skill: ela exige 10 conjuntos, cada um por um ângulo de copy nomeado e nesta ordem: Benefício direto, Curiosidade, Dor, Prova, Urgência, Pergunta, Mecanismo único, Transformação, Quebra de objeção e Específico/numérico. Aqui mora o porquê de funcionar tão bem: não são "10 títulos parecidos", são 10 portas psicológicas diferentes para o mesmo produto. Um anúncio pode falhar porque o público não responde a "benefício" mas reage forte a "medo de perder" (urgência) ou a uma "pergunta". Entregando os 10 ângulos, a skill te dá um cardápio para testar, não um chute único. Quarta: ela embute padrão de qualidade, pt-BR natural, proibição de clichê de IA ("Descubra o poder de…", "Eleve seu negócio a outro nível"), foco no resultado do público e, importante para o seu nicho, proibição de promessa de cura ou ganho garantido em saúde, estética e finanças, respeitando as políticas do Meta.

O exemplo que a própria skill carrega (um curso de inglês para viagem, público 40-60 anos com medo de passar vergonha) mostra o ângulo "Dor" virando "Cansado de travar na hora de pedir um café lá fora?". Para a demo ao vivo, use o caso da clínica de estética: produto "harmonização facial", público "mulheres 35+ que querem rejuvenescer sem exagero". A skill devolve as 10 versões e, no fim, uma linha de dica dizendo qual ângulo testar primeiro. É a tradução perfeita do conceito: você escreveu o manual uma vez, e agora colhe 10 ângulos profissionais em segundos.

COMO EXPLICAR ISSO PRO ALUNO

- Pensa numa skill como uma receita que você entrega pro Claude uma vez só. Depois disso, toda vez que você pedir aquele prato, ele já sabe os ingredientes e a ordem dos passos. Você não precisa repetir nada. E o melhor: a receita é escrita em português, sem uma linha de código. Quem sabe escrever um bom briefing já sabe criar uma skill.
- O arquivo tem duas partes. Em cima fica o crachá: o nome da skill e uma descrição de quando usá-la. Embaixo fica o treinamento: as instruções do que fazer, em que ordem, e o que é proibido. Você digita uma barra e o nome (/gerador-headlines) e o Claude carrega esse manual e passa a trabalhar exatamente do jeito que você ensinou.
- Skill é como anexo de e-mail: só abre o de quem você confia. Mas tem uma vantagem enorme. Você consegue ler o anexo inteiro antes de abrir. É texto puro em português. Dá pra abrir no Bloco de Notas e ver exatamente o que ele manda o Claude fazer. Nada fica escondido.
- Por que a skill de headlines entrega 10 e não 1? Porque cada uma usa uma porta diferente da cabeça do cliente: uma fala do benefício, outra do medo de perder, outra faz uma pergunta. Você não recebe um chute, recebe um cardápio pra testar. É como pescar com dez iscas em vez de uma.

PONTOS DE ATENÇÃO

- O nome (name) não pode ter espaço nem acento. Use letras minúsculas e hífens, como gerador-headlines. Acionar é digitar barra + esse nome exato. Se errar o nome, a barra não encontra a skill.
- Não prometa que a skill "aprende sozinha" ou "fica mais inteligente com o uso". Ela não muda. É um arquivo de texto fixo. O que muda a qualidade da saída é você melhorar o texto do SKILL.md, não o uso repetido. Diga isso com clareza para não criar expectativa falsa.
- O campo description é o que faz o Claude acionar a skill sozinho a partir de um pedido em linguagem natural. Descrição vaga faz a skill quase só funcionar pela barra. Não venda "ele adivinha quando usar" como mágica. Funciona porque a descrição lista os gatilhos certos.
- A skill funciona em todos os planos, Free incluído. Esse é um fato verificado e bom de citar. Mas cuidado com o limite que pertence a outro assunto: no Free, conector customizado próprio é limitado a 1. Isso é conector, não skill. Não misture os dois ao responder, para não passar informação errada.

09 Marketing na prática com o Claude

Até aqui você entendeu o que o Claude é, como ele age no seu computador com segurança, como conectar ferramentas e como criar skills. Este capítulo é onde tudo isso vira faturamento. A pergunta que o aluno iniciante faz na call é sempre a mesma: "tá, mas o que EU faço com isso amanhã de manhã?". A resposta honesta é que o Claude não substitui a sua cabeça de marketing. Ele substitui as duas horas que você perde escrevendo a quarta versão de uma headline, organizando 200 criativos numa pasta bagunçada, ou montando do zero um relatório que o cliente pediu para ontem. Ele é o estagiário sênior que nunca cansa, nunca reclama e entrega um primeiro rascunho em segundos, contanto que você saiba pedir direito.

E "pedir direito" é a habilidade central deste capítulo. A diferença entre um profissional que tira ouro do Claude e um que tira lixo genérico não está no modelo. É o mesmo modelo para os dois. Está no prompt. Um leigo escreve "faz uma copy de estética". Um estrategista escreve um prompt com contexto, exemplo e formato definido, e por isso recebe material que ele só precisa revisar, não reescrever. Vou ancorar o capítulo inteiro num caso concreto (uma clínica de estética que vende harmonização facial) para você sair daqui com prompts-modelo prontos para copiar, adaptar e usar ao vivo na frente dos seus alunos.

A anatomia de um bom prompt: contexto, exemplo, formato (CEF)

Antes de qualquer prompt específico, fixe um modelo mental que serve para TUDO o que você vai pedir ao Claude: Contexto, Exemplo, Formato. Eu chamo de CEF. É o esqueleto invisível por trás de cada prompt deste capítulo.

Contexto é você dizer ao Claude quem é o cliente, quem é o público e qual é o objetivo. Sem contexto, ele preenche as lacunas com o lugar-comum da internet, e o resultado sai com cara de IA porque é, literalmente, a média da internet sobre o assunto. Compare: 'escreve uma headline de harmonização facial' versus 'escreve uma headline para uma clínica de estética premium em Belo Horizonte, cujo público são mulheres de 35 a 50 anos que querem rejuvenescer sem parecer artificial e têm medo de ficar com cara de plástico'. A segunda versão já elimina 80% das respostas ruins, porque você cortou o medo do público (o 'cara de plástico') que vira o ângulo de copy.

Exemplo é você mostrar, não só descrever, o que considera bom. Se você tem uma headline antiga que converteu, cole. Se você gosta do tom de uma concorrente, descreva. O Claude aprende o seu gosto por imitação muito mais rápido do que por adjetivos. Dizer 'tom sofisticado' é vago. Colar uma frase que você acha sofisticada é preciso.

Formato é você definir a forma da entrega: quantas opções, em que estrutura, com ou sem emoji, tamanho de cada peça, se quer em tabela ou em lista. 'Me devolve 5 opções de headline, cada uma com no máximo 12 palavras, numeradas, e abaixo de cada uma escreva em uma linha qual gatilho mental ela usa.' Quando você não define o formato, o Claude escolhe um, e quase sempre não é o que você queria, e aí você perde uma rodada pedindo de novo. Definir o formato na primeira mensagem economiza idas e voltas.

Guarde o CEF. Todo prompt-modelo a seguir é uma aplicação dele.

Copy que converte: headlines, e-mail e anúncios

Comecemos pelo pão com manteiga do marketing: a copy. Aqui o Claude brilha porque escrever variações é cansativo para humanos e trivial para um LLM. Ele não fica sem ideia na décima opção.

Headlines. O jeito de pedir importa muito. Um prompt-modelo bom para a clínica de estética seria: 'Você é copywriter de uma clínica de estética premium. Produto: harmonização facial. Público: mulheres de 35 a 50 anos que querem rejuvenescer com naturalidade e têm medo de exagero. Me dê 10 conjuntos de headline + sub-headline, cada um num ângulo de copy diferente (benefício, dor, prova social, autoridade, antes-e-depois, urgência, identidade, objeção quebrada). Máximo 12 palavras na headline. Numere e, abaixo de cada conjunto, escreva em uma linha qual ângulo usou.' Note que esse prompt é exatamente o que a skill 'Gerador de Headlines' da aula faz de forma empacotada. A skill é esse prompt salvo e refinado, para você não ter que digitá-lo toda vez (o Capítulo 8 mostra a skill por dentro. Aqui o ponto é o raciocínio do prompt manual por trás dela).

E-mail. Aqui o segredo é pedir a sequência inteira de uma vez e pedir o assunto separado do corpo. Modelo: 'Escreva um e-mail de venda para a lista da clínica oferecendo uma avaliação gratuita de harmonização facial. Tom acolhedor e consultivo, não agressivo. Estruture assim: 3 opções de linha de assunto (curtas, sem clickbait), depois o corpo com abertura que cria identificação, um parágrafo de benefício, um de quebra de objeção (medo de ficar artificial) e uma chamada para agendar pelo WhatsApp. Máximo 180 palavras no corpo.' Repare que o medo voltou, porque você o pôs no contexto, ele aparece na peça.

Anúncios. Para Meta Ads, peça o texto pensando na estrutura do Gerenciador: primária, título e descrição, separados. Modelo: 'Escreva 3 variações de anúncio para Instagram/Facebook da clínica. Para cada uma, me dê: texto primário (até 125 caracteres para não cortar), título (até 40 caracteres) e descrição (até 30 caracteres). Ângulo: naturalidade do resultado. Sem promessa de cura, sem antes-e-depois explícito por causa das políticas do Meta sobre saúde.' Aqui entra um detalhe de profissional: avisar o Claude das restrições da plataforma faz ele já escrever dentro da regra, evitando reprovação do anúncio.

Roteiro de Reels e carrossel: do gancho ao CTA

Conteúdo orgânico é onde o iniciante mais trava, porque exige ritmo e gancho, duas coisas que o Claude entrega rápido se você estruturar o pedido por blocos.

Para um Reels, peça o roteiro dividido em gancho, desenvolvimento e chamada, e peça também a indicação visual de cada cena, porque o roteiro serve para você gravar, não só para ler. Modelo: 'Escreva um roteiro de Reels de 30 segundos para a clínica de estética sobre harmonização facial. Público: mulher de 40 anos com medo de ficar artificial. Divida em: GANCHO (primeiros 3 segundos, uma frase que faça ela parar de rolar), DESENVOLVIMENTO (o que falar, em 3 ou 4 falas curtas) e CTA. Para cada bloco, sugira o que mostrar na tela. Tom de conversa, não de propaganda.' O gancho é o ativo mais valioso. Peça inclusive 3 opções só de gancho se quiser escolher.

Para um carrossel, o pulo do gato é pedir uma ideia por slide, com a frase do slide e uma nota do que vai na arte. Modelo: 'Crie um carrossel de 7 slides para o Instagram da clínica desmistificando 5 mitos sobre harmonização facial. Slide 1 é a capa com um gancho forte. Slides 2 a 6, um mito por slide com a verdade ao lado. Slide 7 é o CTA para agendar avaliação. Para cada slide, me dê o texto que vai aparecer (curto, porque é arte) e uma orientação de design em uma linha.' Esse formato 'mito x verdade' funciona porque ataca objeções, que é o que prende quem está na dúvida.

Detalhe que separa o amador do profissional: peça o roteiro a partir de uma dor real, não de um tema genérico. 'Faz um Reels sobre estética' produz lugar-comum. 'Faz um Reels respondendo a mulher que tem medo de ficar com cara de plástico' produz conteúdo que fala com alguém. A dor entra no contexto. O resto o Claude desenrola.

Página de captura em HTML pronta para subir

Aqui é onde o leigo descobre que o Claude faz coisa que ele achava que precisava de programador. Pedir uma página de captura em HTML é um dos momentos de maior impacto numa demo ao vivo, porque o resultado é visível e imediato.

O Claude entrega o código HTML completo de uma landing page de captura (com campo de nome, e-mail e WhatsApp, copy de oferta e botão) num único arquivo que você abre no navegador para ver. Modelo de prompt: 'Crie uma página de captura em HTML, em um único arquivo, para a clínica de estética. Oferta: e-book gratuito "5 sinais de que a harmonização facial é para você". Quero: um título forte, um subtítulo, 3 bullets de benefício, um formulário com nome, e-mail e WhatsApp, e um botão chamando para baixar. Use uma paleta sóbria e elegante (tons de bege e dourado), fonte serifada no título. Deixe responsivo para celular. Use só HTML e CSS, sem depender de internet.'

Dois pontos que você precisa dominar para ensinar isso com honestidade. Primeiro: o Claude entrega a página, o visual e a copy, mas a página ainda não 'funciona' sozinha. O formulário precisa ser ligado a algum lugar que receba os leads (uma ferramenta de e-mail, uma planilha, um webhook). Você pode pedir ao Claude para preparar o formulário para enviar os dados a um link de automação que você já tenha, mas a criação dessa automação é outro passo. Não venda 'página funcionando 100%' se você só gerou o HTML. Venda 'página pronta para conectar'. Segundo: como é HTML em um arquivo só, você consegue abrir e ver o resultado na hora, pedir 'deixa o botão maior', 'troca o dourado por um rosé', 'reescreve o título mais curto', e ele refaz na hora. Essa edição conversacional, ao vivo, é o que encanta na demo. Mostre você pedindo uma mudança e a página mudando. Vale mais que dez slides explicando.

Organizar criativos e arquivos sem dor de cabeça

Esse uso é menos glamoroso que copy, mas é o que faz o aluno perceber que o Claude age no computador, não só responde no chat. É também onde o Capítulo 6 (segurança e aprovação) deixa de ser teoria: você vai ver o Claude propor mudanças nos seus arquivos e esperar seu 'aceitar'.

O cenário típico do marketeiro é a pasta de criativos virada num caos: 'imagem1.png', 'final_FINAL_v2.jpg', 'sem título.png', tudo misturado, de três campanhas diferentes. Você pode pedir ao Claude, no app desktop, para organizar isso. Modelo: 'Na minha pasta de criativos da clínica de estética, renomeie os arquivos de imagem seguindo o padrão CAMPANHA-FORMATO-VERSAO (exemplo: harmonizacao-reels-v1) e separe em subpastas por campanha. Antes de mexer em qualquer coisa, me mostre a lista do que você pretende renomear e mover, e espere eu aprovar.'

Dois aprendizados importantes aqui. Um: como o Claude age no seu PC, ele mostra o plano e pede aprovação antes de renomear ou mover. A mecânica de permissão do Capítulo 6 aparece na prática. Você revisa a lista e só então autoriza. Se ele entendeu errado, você rejeita e corrige sem nada ter sido alterado. Dois: peça sempre que ele te mostre o plano antes ('me mostre antes de mexer'). Isso te dá controle e evita que ele faça uma reorganização que não era bem o que você queria. Para arquivos importantes, vale ter um backup, como você teria antes de qualquer faxina grande. A organização também serve para texto: 'leia todos os textos de anúncio nessa pasta e monte um índice numa tabela com nome do arquivo, primeira linha e tema' transforma uma pilha de arquivos soltos num catálogo navegável em segundos.

Relatório de campanha a partir de um export

O relatório é o entregável que mais consome tempo do gestor de tráfego e o que mais impressiona o cliente quando bem feito. Aqui o Claude pega o trabalho braçal (ler a planilha, somar, comparar, achar padrão) e devolve análise em linguagem de gente.

O fluxo é simples: você exporta os dados da campanha do Gerenciador de Anúncios (um arquivo CSV ou planilha com gastos, cliques, leads, custo por lead) e entrega esse arquivo ao Claude no app desktop. Modelo: 'Anexei o export da campanha de harmonização facial da clínica deste mês. Faça um relatório para o cliente, que não é técnico. Estruture em: resumo em 3 frases do que aconteceu, tabela com os 3 melhores e os 3 piores anúncios por custo por lead, uma análise do que provavelmente explica a diferença, e 3 recomendações práticas para o próximo mês. Linguagem simples, sem jargão de mídia.'

O que faz esse relatório ser bom é o contexto que você dá sobre o leitor: 'para o cliente, que não é técnico' muda completamente a saída. O Claude troca 'CTR' por 'taxa de cliques' e explica o número em vez de só citar. E aqui vem o ponto de profissional, que se conecta com o Capítulo 1: o Claude lê e organiza muito bem, mas você confere os números antes de mandar para o cliente. Ele pode somar errado ou interpretar uma coluna trocada se o export estiver bagunçado. O relatório que ele te dá é 90% do trabalho pronto. Os 10% finais (bater os totais e validar a análise) são seus, e são o que justifica o seu cachê. Encadeie isso com o resto do fluxo: depois do relatório, no mesmo chat, peça 'agora escreve um e-mail curto para o cliente apresentando esse relatório e propondo uma call', e você fechou o ciclo, da planilha bruta ao e-mail de entrega, numa conversa só.

Encadear conector + skill: o fluxo de trabalho completo

Os dois capítulos anteriores apresentaram as peças: conectores (Capítulo 7) ligam o Claude às suas ferramentas. Skills (Capítulo 8) empacotam um pedido bem-feito num comando. A mágica para o marketeiro acontece quando você encadeia os dois numa conversa só, e é isso que transforma o Claude de 'chat esperto' em 'assistente de operação'.

Vou montar o fluxo completo da clínica, na ordem em que aconteceria numa conversa real. Passo um, dados via conector: 'No Google Drive, abra a planilha "Pesquisa de clientes - clínica estética" e me diga, em 5 bullets, quais são os principais medos e desejos que aparecem nas respostas.' O conector do Drive (que você autorizou conforme o Capítulo 7) traz dado real do seu cliente para dentro do raciocínio. Você não está mais escrevendo copy no escuro. Passo dois, skill em cima do dado: 'Agora use a skill /headlines com esses medos e desejos que você acabou de levantar, para o produto harmonização facial.' A skill recebe contexto de verdade, extraído da pesquisa, e não um avatar inventado. Passo três, peça em massa: 'Pegue as 3 melhores headlines e crie, para cada uma, um e-mail curto e um anúncio de Meta Ads.' Passo quatro, devolva para a ferramenta: 'Salve tudo num documento no Google Drive chamado "Copy clínica - junho".'

Esse encadeamento (Drive traz o dado, a skill processa, o Claude expande em peças, o Drive guarda o resultado) é um fluxo de trabalho inteiro rodando numa única conversa, com você só conduzindo. Cada peça já existia nos capítulos anteriores. O que este capítulo ensina é a coreografia. E o princípio geral para você levar: o conector é para entrada e saída de dados (de onde vem a informação, para onde vai o resultado), a skill é para repetir um pedido complexo sem redigitar, e o Claude no meio é quem cola as duas pontas. Quando o aluno entende essa divisão de papéis, ele para de ver comandos soltos e passa a desenhar operações.

COMO EXPLICAR ISSO PRO ALUNO

- Pense no Claude como um estagiário brilhante que entrou hoje na sua agência. Ele é capaz de tudo, mas não conhece o cliente. Se você só diz 'escreve uma copy', ele te dá a copy genérica que qualquer um daria. Se você passa o briefing (quem é o cliente, quem é o público, do que ele tem medo, qual o tom) ele te entrega material que parece que você escreveu. A qualidade do que sai depende da qualidade do briefing que entra.
- Existe uma fórmula simples para qualquer pedido: contexto, exemplo e formato. Contexto é dizer quem é o cliente e o público. Exemplo é mostrar uma peça que você acha boa, em vez de só descrever. Formato é dizer como você quer receber: quantas opções, em tabela ou lista, com que tamanho. Quem segue essas três coisas acerta de primeira. Quem pula direto pro pedido fica brigando com o resultado.
- Deixa eu te mostrar uma coisa que parece bruxaria. Vou pedir pro Claude criar uma página de captura inteira pra essa clínica de estética. [gera] Pronto: título, formulário, botão, tudo num arquivo que eu abro no navegador agora. E olha o melhor: se eu não gostei do botão, eu falo 'deixa o botão maior e dourado', e ele muda na hora. Não precisei chamar programador, não precisei mexer em código. Eu conversei com a página até ela ficar do jeito que eu quero.
- O Claude faz 90% do relatório de campanha pra você: ele lê a planilha de gastos e leads, organiza, acha os melhores e piores anúncios e escreve a análise em português de gente. Mas os 10% finais são seus: você confere se os números batem antes de mandar pro cliente. Esse conferir é justamente o que você cobra. A IA fez o trabalho chato. Você botou o nome e a responsabilidade.

PONTOS DE ATENÇÃO

- Não prometa 'página funcionando 100%' quando você só gerou o HTML. O Claude entrega o visual, a copy e o código da página, mas o formulário ainda precisa ser ligado a algo que receba os leads (e-mail, planilha, webhook de automação). Venda 'página pronta para conectar', não 'página no ar capturando leads'. Senão você cria uma expectativa que vira reclamação.
- Sempre confira os números de qualquer relatório antes de entregar ao cliente. O Claude lê e organiza muito bem, mas pode somar errado ou trocar uma coluna se o export estiver bagunçado. É o mesmo risco de alucinação do Capítulo 1, agora com dinheiro do cliente em jogo. Bater os totais leva dois minutos e protege a sua reputação.
- Ao pedir para o Claude organizar, renomear ou mover arquivos, peça sempre que ele mostre o plano antes de executar ('me mostre o que vai mudar e espere eu aprovar'). Ele já trabalha assim por design (Capítulo 6), mas reforçar no pedido te dá controle. Para arquivos importantes, tenha um backup, como em qualquer faxina grande. O ganho de tempo não vale o risco de perder um criativo sem cópia.
- Cuidado com as políticas das plataformas, principalmente em saúde e estética. Avise o Claude das restrições no próprio prompt ('sem promessa de cura', 'sem antes-e-depois explícito por causa das regras do Meta'). Ele não conhece a política específica do seu nicho a fundo e pode escrever uma promessa que reprova o anúncio. A regra entra no contexto. A responsabilidade final de estar dentro das normas é sua.

10 Segurança, privacidade, mitos e boas práticas

Até aqui você aprendeu a instalar, conectar e mandar o Claude trabalhar. Este capítulo é o que separa quem usa a ferramenta de quem usa a ferramenta com responsabilidade, e, na frente de uma turma, isso é o que constrói sua autoridade. Iniciante tem medo, e o medo vem quase sempre de três perguntas não ditas: "ele vai me substituir?", "ele é confiável?" e "ele rouba meus dados?". Quem ensina precisa responder essas três com calma e com fato, sem vender promessa nem espalhar pânico. A honestidade aqui não é só ética. É estratégia de ensino. O aluno que entende o limite confia mais na ferramenta, não menos.

O fio condutor deste capítulo é simples: o Claude é um colega muito capaz, mas é um colega que você revisa. Ele tem dois riscos reais que valem ouro saber explicar: o de inventar com confiança (a alucinação, que você viu no Capítulo 1) e o de você dar a ele acesso a coisas que não precisava. Os dois se resolvem com a mesma postura adulta: pedir bem, revisar sempre, e dar só o acesso necessário. Vamos destrinchar privacidade de dados de forma concreta, listar o que de fato não fazer, derrubar os mitos com a verdade no lugar, e fechar com o método de revisão crítica que transforma uma saída boa numa saída usável.

O que acontece com seus arquivos e seus dados

Quando você manda o Claude resumir um PDF, escrever uma copy a partir de um briefing ou organizar uma pasta no Cowork, esse conteúdo precisa ser processado nos servidores da Anthropic para a inteligência funcionar. É o mesmo princípio de qualquer serviço em nuvem, como subir um documento no Google Drive. O ponto que pesa para um profissional é diferente de "meus dados saem do meu computador?" (saem, é assim que a IA funciona). O ponto é: "esse conteúdo vira material de treino do modelo, que outras pessoas poderiam ver reflexo disso depois?". E aqui mora uma distinção que você precisa dominar para responder com segurança.

A regra prática que vale citar em aula: nos planos corporativos (Team e Enterprise) a Anthropic não usa o conteúdo dos clientes para treinar seus modelos por padrão. É um compromisso de produto pensado justamente para empresas que lidam com dados sensíveis e não podem correr o risco de um contrato ou base de clientes virar combustível de treino. Para os planos individuais (Free, Pro, Max), o tratamento de dados é regido pelos termos e pelas configurações de privacidade da conta, e a recomendação honesta para o aluno é: leia a tela de privacidade da sua conta e ajuste conforme seu conforto, em vez de assumir um padrão de cabeça. Não invente porcentagens nem prazos que você não viu. Aponte para a configuração real.

O exemplo concreto que ancora isso: se você é o estrategista de uma clínica de estética e vai trabalhar a base de pacientes de verdade dentro do Claude, o lugar disso é um plano de empresa (Team/Enterprise), não a conta Pro pessoal que você usa para rascunhar headlines. Separar as duas coisas (brincadeira/aprendizado numa conta, dado real de cliente noutra com a proteção contratual certa) é a higiene básica que um profissional sério adota. Não é paranoia. É o mesmo bom senso de não guardar o contrato do cliente na mesma gaveta bagunçada dos rascunhos.

Princípio do menor acesso: dê só o que a tarefa precisa

Esta é a ideia de segurança mais valiosa do capítulo e a mais fácil de ensinar, porque tem analogia perfeita no mundo físico. Quando um encanador vai consertar a pia, você abre a porta da cozinha. Não entrega a chave mestra do prédio inteiro. Com o Claude é igual: quando você conecta uma ferramenta (um conector, no vocabulário do Capítulo 7) ou aprova uma ação, está abrindo uma porta. A pergunta certa antes de cada "Conectar" ou "Aceitar" é sempre a mesma: essa tarefa precisa mesmo desse acesso?

Na prática, isso aparece em três momentos. Primeiro, ao autorizar um conector: a tela de login mostra quais permissões você está concedendo, por exemplo, um conector de e-mail pode pedir só para ler, ou ler e enviar. Se você só quer que o Claude resuma e-mails, não há motivo para conceder envio. Vale lembrar o caso real que você viu no Capítulo 7: o conector do HubSpot é só de leitura, um exemplo concreto de "acesso mínimo" embutido no próprio produto, e um bom argumento de que dar menos acesso muitas vezes basta. Segundo, ao aprovar ações no Code/Cowork: como você viu no Capítulo 6, o Claude mostra o que vai mudar e espera seu Aceitar. Leia o diff, não clique no automático. Terceiro, ao escolher o modo de trabalho do Cowork: existe o modo "pedir antes de agir" e o modo "agir sozinho". Para tarefa nova, com dados que importam, ou enquanto você ainda está aprendendo, fique no "pedir antes". O "agir sozinho" é conveniência que você libera depois de confiar na rotina, não no primeiro dia.

Um exemplo que cola na cabeça do aluno: você pede para o Claude "limpar e renomear as fotos dessa pasta de criativos". No modo "pedir antes", ele te mostra a lista de renomeações ANTES de mexer. Você bate o olho, vê que faz sentido, aprova. Se estivesse no "agir sozinho" e o pedido fosse ambíguo, ele poderia renomear de um jeito que você não queria, e aí dá trabalho desfazer. O menor acesso e o "pedir antes" existem exatamente para que experimentar seja barato e errar seja reversível.

O que NÃO fazer: a lista que evita dor de cabeça

Três erros concentram quase todo o risco real para um usuário de marketing, e os três são totalmente evitáveis. Vale ensinar como regra de ouro, não como susto.

Não suba dado sensível sem necessidade. Antes de colar algo no Claude, pergunte: "essa informação precisa estar aqui para a tarefa sair?". Para escrever uma copy de harmonização facial, o Claude precisa do perfil do público e da oferta. Não precisa do CPF, do telefone e do prontuário dos pacientes. Se o dado não é necessário para o resultado, ele não deveria estar ali. Quando o dado real é indispensável (uma base para análise de campanha, por exemplo), aí entra a regra do plano certo da primeira seção: faça isso no ambiente com a proteção contratual adequada.

Não instale skill de fonte duvidosa. Como você viu no Capítulo 8, uma skill é um arquivo de instruções que o Claude executa quando você a aciona, ou seja, ela manda o Claude fazer coisas. A analogia oficial é tratar skill como anexo de e-mail: você não abre um .exe que chegou de um remetente desconhecido, e pela mesma razão não instala uma skill de origem que você não confia. A boa notícia é que skill é texto legível. Dá para abrir o SKILL.md e ler o que ela faz antes de ativar. Se você não entende o que está escrito lá ou não sabe de onde veio, não instale. Fonte confiável: você mesmo, um colega conhecido, ou um catálogo oficial.

Não confie cegamente na saída. Este é o erro mais comum e o mais traiçoeiro, porque a saída do Claude é bem escrita e parece certa mesmo quando não está. Nunca publique, envie ao cliente ou tome decisão com base numa entrega que você não leu com olhar crítico. A última seção deste capítulo é justamente o método de revisão, porque "revisar" não pode ser uma intenção vaga, tem que ser um hábito com passos.

Os três mitos, e a verdade no lugar de cada um

Você vai ouvir essas três objeções em toda turma de iniciante. Tê-las respondidas com calma e fato é metade da sua autoridade no palco.

Mito 1: "vai me substituir". A verdade honesta: o Claude não tem o seu repertório de cliente, o seu faro de oferta, nem a responsabilidade pela decisão. Ele acelera a parte mecânica: gerar dez ângulos de headline em segundos, transformar um briefing em rascunho de página, organizar um monte de arquivo. Quem escolhe qual ângulo presta, qual ajustar ao tom da clínica e o que vai pro ar é você. A leitura certa não é "a IA contra o profissional", é "o profissional com IA contra o profissional sem IA". Quem aprende a dirigir a ferramenta produz mais e melhor. Quem a ignora fica para trás. Isso vende a aula sem mentir.

Mito 2: "é infalível / a resposta sempre está certa". A verdade: não. Como você viu no Capítulo 1, o ponto fraco estrutural de um modelo de linguagem é a alucinação. Ele pode afirmar com total confiança um dado, um nome ou um número que simplesmente não existe, porque ele prevê texto plausível, não consulta uma verdade. Por isso a regra: tudo que for fato verificável (preço, prazo, estatística, nome próprio, citação) você confere antes de usar. A confiança do tom não é prova de exatidão. É só estilo. Ensinar isso não enfraquece a ferramenta, fortalece o aluno.

Mito 3: "rouba meus dados". A verdade, já destrinchada na primeira seção: seus dados são processados para a IA funcionar (igual a qualquer serviço de nuvem), mas "processar" não é "roubar" nem "vazar". Nos planos de empresa, o conteúdo não vira treino por padrão. Nos planos individuais, você ajusta isso na configuração de privacidade da conta. A postura madura não é nem "ignoro o tema" nem "tenho pavor". É ler a tela de privacidade, escolher o plano certo para o tipo de dado, e aplicar o menor acesso. Medo informado vira procedimento. Medo desinformado vira paralisia.

Como revisar criticamente o que o Claude entrega

Revisar não é desconfiar de tudo. É um hábito com passos, rápido depois que vira rotina. Quatro perguntas dão conta da maioria das entregas de marketing.

Primeira: os FATOS estão certos? Marque mentalmente tudo que é verificável na saída (número, preço, prazo, nome de pessoa, dado de mercado, link) e confira na fonte real. Se o Claude diz "a harmonização facial dura em média X meses", isso é uma afirmação factual: você valida antes de colocar num anúncio, porque promessa errada em saúde é problema sério. Texto criativo (um ângulo de copy, uma ideia de gancho) você julga por gosto e estratégia. Afirmação factual você confere por fonte.

Segunda: faz sentido para o MEU contexto? O Claude não conhece a sua clínica, o seu tom, as suas restrições. Uma headline pode ser tecnicamente boa e ainda assim agressiva demais para o posicionamento do cliente, ou prometer algo que o serviço não entrega. Você é o filtro de adequação.

Terceira: tem promessa ou claim que me expõe? Em nichos sensíveis (saúde, estética, finanças) existem limites do que se pode afirmar. Leia a saída caçando exagero ("resultado garantido", "sem riscos", "melhor da cidade") e corte ou suavize o que não se sustenta. A IA não conhece o seu código de conduta nem a regulação do seu setor. Esse julgamento é seu.

Quarta: eu assinaria isso? O teste final e mais honesto. Se você colocaria seu nome embaixo daquela entrega e a mandaria para o cliente sem vergonha, está pronta. Se hesitou, volte e ajuste. E quando algo está quase bom, não recomece do zero. Diga ao Claude exatamente o que mudar ("deixe mais formal", "tire a promessa de resultado", "foque no público acima de 40 anos"). É assim que pedir bem e revisar bem se fecham num ciclo: você dirige, ele executa, você revisa, ele ajusta, e a entrega final é sua, com a sua assinatura e o seu critério.

COMO EXPLICAR ISSO PRO ALUNO

- O Claude é tipo um estagiário genial e supereficiente: faz o trabalho pesado numa velocidade absurda, mas você não manda o que ele escreveu direto pro cliente sem ler. Você revisa, ajusta o tom, confere os números, e ENTÃO assina embaixo. A responsabilidade continua sendo sua, e é por isso que ele não te substitui, ele te turbina.
- Conectar uma ferramenta ao Claude é como dar a chave da sua casa. Antes de entregar a chave, pergunte: ele precisa da chave da casa toda, ou só do portão? Se você só quer que ele leia seus e-mails, não dê permissão pra ele ENVIAR e-mails. Dê sempre o mínimo que a tarefa precisa. Dá pra liberar mais depois, com calma.
- Ele escreve tão bem e com tanta confiança que parece que está sempre certo, e essa é exatamente a pegadinha. A IA pode inventar um número ou um nome com a maior cara de séria. Então a regra é simples: tudo que for FATO (preço, prazo, estatística, nome) você confere na fonte antes de usar. Ideia criativa você julga pelo gosto. Fato você confere sempre.
- Pensa numa skill como um anexo de e-mail: você nunca abre um arquivo que veio de um desconhecido. Skill é a mesma coisa. Ela manda o Claude fazer coisas, então só instale de quem você confia. E a parte boa: skill é texto que dá pra ler antes. Na dúvida, abre, lê o que está escrito, e só ativa se entendeu e confia.

PONTOS DE ATENÇÃO

- Não crave números sobre privacidade que você não viu na tela. A afirmação segura e verificada é: Team e Enterprise não treinam com o conteúdo dos clientes por padrão. Para Free/Pro/Max, oriente o aluno a ler a configuração de privacidade da própria conta. Não invente porcentagem, prazo de retenção ou política específica de cada plano individual.
- Cuidado com o falso conforto de dizer 'os dados não saem do seu PC'. Eles saem, sim. É assim que a IA funciona, igual a qualquer serviço de nuvem. O argumento correto não é negar o processamento, é explicar a diferença entre processar (normal e necessário) e virar treino (que nos planos de empresa não acontece por padrão). Honestidade aqui é o que constrói confiança.
- O modo 'pedir antes de agir' versus 'agir sozinho' é uma configuração do Cowork (visto no Capítulo 6). Não apresente isso como se fosse a salvaguarda principal do Claude inteiro. É uma das camadas. Posicione corretamente: para iniciante e para tarefa com dado que importa, recomende 'pedir antes'. O 'agir sozinho' é conveniência liberada depois que a rotina já é confiável.
- Não transforme o capítulo num discurso de medo. O enquadramento certo é 'segurança por design': as permissões, o diff de aprovação e o menor acesso existem justamente para que experimentar seja barato e errar seja reversível. O objetivo é o aluno usar com confiança e critério, não sair da aula com receio de clicar em qualquer coisa.

A-Z Glossário sem jargão

Todo termo da aula explicado em uma frase, do jeito que você explicaria para um leigo.

Agente de IA

Uma IA que não só responde, mas AGE: ela entende seu pedido, monta um plano com vários passos e executa cada um (abrir arquivos, escrever, organizar pastas) até concluir. Pense num assistente que, em vez de só te dar a receita, vai até a cozinha e cozinha o prato. O Claude Code é um agente.

Alucinação

Quando a IA inventa uma informação e a apresenta com total confiança, como se fosse verdade: um número, uma data, um nome de produto, uma citação. Acontece porque o modelo prevê a próxima palavra mais provável, não 'sabe' se aquilo é real. Por isso fato importante sempre se confere.

Anthropic

A empresa americana que cria o Claude. A aposta declarada dela é em IA segura e confiável, ou seja, construir o assistente de forma cautelosa e previsível. Para quem usa, isso se traduz em um sistema que pede aprovação antes de agir, em vez de fazer tudo sozinho.

App desktop

O Claude instalado como um programa de janela no seu computador (igual a abrir o Word ou o Chrome), com botões e telas. É o formato que esta aula usa. Diferente da linha de comando (CLI), aqui você não digita comandos numa tela preta: é tudo clicável.

Caixa isolada (sandbox / VM)

Um espaço fechado e separado onde a IA pode trabalhar sem mexer no resto do seu computador, como uma bancada de testes à parte. O Cowork usa esse tipo de ambiente para experimentar com segurança: se algo der errado ali, não afeta seus arquivos pessoais.

Claude

O assistente de IA da Anthropic na sua forma de chat: você escreve uma pergunta ou pedido e ele responde em texto. É o ponto de partida, o 'conversar com a IA'. Os outros formatos (Claude Code e Cowork) fazem o Claude sair do chat e agir no computador.

Claude Code

A versão do Claude que vira um AGENTE: além de responder, ele age no seu computador, cria e edita arquivos, organiza pastas e executa tarefas em vários passos. Roda no terminal, na IDE, no app desktop e no navegador. Nesta aula, o foco é usá-lo dentro do app desktop.

CLI (linha de comando / terminal)

A forma de usar o computador digitando comandos em texto numa tela (normalmente preta), em vez de clicar em botões. O termo CLI vem de 'Command Line Interface'. O Claude tem uma versão que roda ali, mais técnica. A aula evita isso e fica no app desktop, que é visual.

Conector

A ponte que liga o Claude a um aplicativo ou conta sua (Google Drive, Gmail, Notion, Slack, Canva...) para ele ler e usar esses dados. Você o ativa dentro de um chat pelo botão + > Connectors, escolhe o serviço e faz login. O catálogo oficial fica em claude.com/connectors.

Contexto

Tudo o que você já deu para o Claude considerar naquela conversa: o pedido, os exemplos, os arquivos anexados, o histórico do chat. Quanto mais contexto relevante você fornece, melhor a resposta, porque ele tem mais base. É a memória de trabalho daquela conversa.

Cowork

O poder do Claude Code num formato pensado para quem não é técnico: ele faz o trabalho de agente (planejar, executar vários passos, mexer em arquivos) sem te obrigar a entender de programação, e roda dentro de uma caixa isolada para experimentar com segurança.

Diff

O 'antes e depois' que o Claude te mostra de cada mudança que pretende fazer num arquivo, destacando o que sai e o que entra. Você lê, e só então aprova ou rejeita. É como revisar uma alteração de contrato com as partes alteradas marcadas antes de assinar.

Frontmatter

O cabeçalho de configuração no topo de um arquivo de skill (o SKILL.md), escrito num formato chamado YAML. Ele traz pelo menos o nome (name) e a descrição (description) da skill. É a 'etiqueta' que diz ao Claude como aquela skill se chama e para que serve.

Git

Um programa de bastidor que o Claude Code usa no Windows para funcionar. No Windows ele é obrigatório: sem ele instalado aparece o aviso 'Git is required'. Baixa-se em git-scm.com, instala e reabre o app. No Mac, em geral, já vem instalado.

Headline

A frase de maior destaque de um anúncio, página ou e-mail, a primeira coisa que o leitor lê e que decide se ele continua. Em português, o título-chamada. A skill da aula gera 10 headlines, cada uma com um ângulo diferente, para você escolher a mais forte.

Inteligência Artificial (IA)

Programas de computador que realizam tarefas que costumavam exigir um humano: entender uma frase, escrever um texto, resumir, sugerir ideias. Não é mágica nem consciência. É um sistema treinado em muitos exemplos que aprendeu a reconhecer e reproduzir padrões.

LLM (modelo de linguagem grande)

O tipo de IA por trás do Claude. A sigla vem de 'Large Language Model', modelo de linguagem grande. Ele foi treinado em uma quantidade enorme de textos e funciona prevendo, palavra por palavra, qual a próxima mais provável, e assim escreve frases que fazem sentido.

MCP (Model Context Protocol)

O padrão aberto que permite ligar o Claude a ferramentas e dados de fora dele (seus apps, suas planilhas, seus serviços). A sigla quer dizer 'Model Context Protocol'. Pense numa tomada universal: é a regra comum que faz o Claude e outros aplicativos 'encaixarem' e conversarem. Os conectores funcionam por cima do MCP.

Modelo

O 'cérebro' de IA específico que processa seu pedido. A família do Claude tem vários, com forças diferentes: Haiku (mais rápido e leve), Sonnet (equilíbrio para o dia a dia) e Opus (o mais poderoso da linha clássica). O iniciante não precisa escolher na mão.

Prompt

O pedido que você escreve para a IA, a instrução ou pergunta. Quanto mais claro e completo o prompt (com contexto, exemplos e o formato que você quer), melhor a resposta. É como um briefing: pedido vago gera entrega vaga. Pedido bem feito gera entrega boa.

Skill

Uma 'habilidade' pronta que você instala no Claude e aciona digitando /nome. Por baixo, é um arquivo de instruções (o SKILL.md) que ensina o Claude a fazer uma tarefa específica do seu jeito, sempre. A skill da aula, por exemplo, gera headlines sob comando.

SKILL.md

O arquivo que contém uma skill por dentro: um cabeçalho (frontmatter) com nome e descrição, mais as instruções em português do que o Claude deve fazer. Como é texto que roda tarefas, trate como anexo de e-mail: instale só de fonte confiável e leia antes de ativar.

Sub-headline

A frase de apoio logo abaixo da headline, que complementa, explica ou reforça a promessa do título principal. Headline chama a atenção. Sub-headline dá o próximo empurrão. Por isso a skill da aula entrega os dois juntos em cada um dos 10 conjuntos.

Token

O pedacinho de texto que a IA usa para ler e gerar conteúdo, em geral um pedaço de palavra (a palavra 'marketing' pode virar dois ou três tokens). O modelo trabalha contando tokens, não letras. Por isso o uso costuma ser falado de forma relativa, não em números exatos.
