

LIVRO GRATUITO

Currículo e LinkedIn de Dev

o checklist que os recrutadores usam

Como a triagem funciona, como escrever cada linha, projetos para quem está começando, três currículos reais de antes e depois, LinkedIn, GitHub e um checklist para imprimir



VALANDROUI

Currículo e LinkedIn de Dev

o checklist que os recrutadores usam

Baixe a versão mais recente em jeffev.github.io/jeffersonvalandro

Jefferson Valandro

1ª edição · outubro de 2026 · distribuição gratuita

© 2026 Jefferson Valandro. Este livro é gratuito: você pode baixar, imprimir e compartilhar à vontade, desde que sem alterações, com o nome do autor e sem cobrar por ele. É proibida a venda.

Ana, Bruno, Carla, a LogiRota e as demais pessoas, empresas, vagas, datas e números dos exemplos são fictícios. As referências a leis são descrições gerais e não substituem orientação jurídica. Recursos de plataformas descritos conforme outubro de 2026.

LinkedIn, GitHub, Hotmart e as demais marcas citadas pertencem aos seus respectivos donos. Este guia não tem vínculo com nenhuma delas.

Sumário

Antes de começar

- 1 Como a triagem funciona de verdade**
- 2 Estrutura, tamanho e formato**
- 3 Tópicos que mostram o que você fez**
- 4 Sem experiência na área: projetos e transição**
- 5 Três currículos, antes e depois**
- 6 LinkedIn: um perfil que se encontra e se lê**
- 7 GitHub: a vitrine do código**
- 8 Erros que eliminam na triagem**
- 9 O checklist final**
- 10 Modelos para copiar**

E agora, a entrevista

INTRODUÇÃO

Antes de começar

Você estudou, montou projetos, talvez até mudou de carreira. Agora precisa que alguém do outro lado leia o seu currículo e pense: "vale a pena conversar com essa pessoa". É aí que muita gente trava. O currículo parece uma formalidade, o LinkedIn parece uma rede social qualquer, e os conselhos que circulam se contradizem.

Este livro resolve esse pedaço. Ele vai do zero até um currículo pronto para enviar e um perfil on-line coerente com ele. No caminho, mostra o que quem faz a triagem costuma procurar, o que atrapalha e o que é mito.

Para quem é

Para quem está procurando a primeira vaga em desenvolvimento, como júnior, e para quem está migrando de outra área para tecnologia. Também serve para quem já trabalha como dev e nunca parou para revisar o currículo. Os exemplos cobrem os três casos.

Os exemplos e as recomendações valem para vagas no **Brasil**, em **outubro de 2026**. Plataformas mudam de tela e de recurso com frequência. Quando o livro falar de um botão do LinkedIn ou do GitHub, confira a versão atual.

Três pessoas que vão nos acompanhar

Os exemplos giram em torno de três personagens **fictícios**. Eles aparecem também no livro sobre entrevistas, do mesmo autor.

- **Ana** trabalhou seis anos como assistente contábil. Estudou Python, fez um curso livre de backend e construiu dois projetos. Quer a primeira vaga como desenvolvedora júnior.
- **Bruno** é desenvolvedor backend há cerca de quatro anos, com Python, FastAPI e PostgreSQL, numa plataforma de gestão para varejo. Quer uma vaga de pleno na LogiRota, uma empresa de logística (também fictícia).
- **Carla** tem mais de dez anos de frontend, com React e TypeScript. Criou o design system usado por três times da empresa onde trabalha.

No capítulo 5, você vai ver o currículo de cada um antes e depois da revisão. É o centro do livro. Os outros capítulos explicam as regras que levam de uma versão à outra.

Uma regra acima das outras

Tudo o que está no seu currículo pode virar pergunta numa entrevista. Por isso, a regra do livro é simples: **escreva só o que é verdade e o que você sabe explicar**. Nada de ferramenta que você só viu num vídeo, número que você não mediu ou cargo que você não teve.

Isso não é só ética. É estratégia. Um currículo honesto e específico gera conversas que você consegue sustentar. Um currículo inflado gera perguntas que derrubam a candidatura inteira.

Como o livro está organizado

- **Capítulos 1 a 4:** como a triagem funciona, a estrutura do currículo, como escrever cada linha e o que fazer quando você ainda não tem experiência na área.
- **Capítulo 5:** três currículos completos, antes e depois.
- **Capítulos 6 e 7:** LinkedIn e GitHub.
- **Capítulos 8 a 10:** os erros que mais eliminam, um checklist para imprimir e modelos para copiar.

Você pode ler na ordem ou ir direto ao capítulo que resolve o seu problema de hoje. Se tiver pouco tempo, leia o capítulo 5 e use o checklist do capítulo 9.

As caixas ao longo do texto têm funções fixas: **Antes** e **Depois** mostram reescritas, **Erro comum** aponta armadilhas, **Dica** traz atalhos práticos e **Lei** resume o que a legislação brasileira diz sobre um tema.

DICA

Leia com o seu currículo aberto ao lado. Cada capítulo termina com algo que você pode aplicar na hora.

E depois do currículo?

O currículo serve para conseguir a conversa. O que acontece na conversa (entrevista comportamental, teste técnico, system design, proposta e negociação) é assunto do livro *Como Passar em Entrevistas para Desenvolvedor*, que fica indicado na última página para quem quiser continuar. Este livro aqui é completo sem ele.

CAPÍTULO 1

Como a triagem funciona de verdade

Antes de mexer no currículo, vale entender o caminho que ele percorre. A maior parte dos conselhos ruins nasce de uma imagem errada desse caminho. Uns imaginam um "robô" que lê tudo e decide sozinho; outros, alguém que lê cada linha com calma. Na prática, a triagem tem várias etapas, e cada uma procura coisas diferentes.

Quem lê o seu currículo

Numa empresa média, o currículo costuma passar por mais de uma pessoa.

1. **A pessoa recrutadora.** Pode ser do RH da empresa ou de uma consultoria contratada. Ela confere se o perfil bate com a vaga: os requisitos obrigatórios, o nível, a localização ou o modelo de trabalho, às vezes o idioma. Nem sempre é técnica. Ela procura **correspondência clara** entre o que a vaga pede e o que você escreveu.
2. **A pessoa técnica ou gestora.** Recebe os currículos que passaram pela primeira leitura. Procura **profundidade**: o que você fez de verdade, com que tecnologia, em que contexto, com que resultado. É ela quem percebe se "trabalhei com Docker" significa rodar um container no curso ou manter a infraestrutura de um time.

Em empresas pequenas, uma única pessoa pode fazer as duas leituras. Em empresas grandes, pode haver mais etapas. Isso **depende da empresa**.

O que vale para todas: a primeira leitura costuma ser **rápida e seletiva**. Quem lê está procurando sinais, não lendo um romance.

ERRO COMUM

"O recrutador lê seu currículo em 6 segundos." Esse tipo de número circula há anos. Ele vem de pesquisas pequenas, feitas por empresas do próprio mercado de recrutamento, com métodos que não permitem generalizar. Não existe um dado confiável e geral sobre quanto tempo cada pessoa gasta. O que se sustenta é a ideia, não o número: a primeira passagem é rápida, então o importante precisa estar fácil de achar.

O que é um ATS

ATS vem do inglês *applicant tracking system*, "sistema de acompanhamento de candidaturas". É o software que a empresa usa para publicar a vaga, receber as candidaturas, guardar os currículos e organizar as etapas do processo. Quando você se candidata pela página de carreiras de uma empresa e preenche um formulário, quase sempre está usando um.

O que esses sistemas costumam fazer, em linhas gerais:

- **Guardar e organizar.** Cada candidatura vira um registro, com etapa, anotações e histórico.
- **Extrair texto do currículo.** Muitos tentam ler o arquivo e preencher campos (nome, e-mail, experiências). Por isso o arquivo precisa ter texto de verdade, e não uma imagem.

- **Permitir busca e filtro.** A pessoa recrutadora pode buscar por termos ("PostgreSQL", "React") ou filtrar por respostas do formulário.
- **Aplicar perguntas eliminatórias.** Se a vaga pergunta "Você tem disponibilidade para trabalho híbrido em Curitiba?" e você responde "não", a candidatura pode sair do processo ali mesmo, de forma automática.
- **Ordenar ou sugerir candidatos.** Alguns sistemas, cada vez mais com recursos de IA, sugerem quem combina mais com a vaga. Como isso é configurado e quanto pesa na decisão **varia muito** e, em geral, não é público.

ERRO COMUM

"O ATS descarta 75% dos currículos antes de um humano ver." Esse número é repetido há mais de uma década, e não há estudo publicado que o sustente. Os sistemas variam, as configurações variam, e muitas empresas usam o ATS só como arquivo organizado. Desconfie de qualquer porcentagem sobre "o robô" que venha sem fonte e sem método.

O que de fato ajuda, com ou sem ATS

Como você não sabe qual sistema a empresa usa nem como ele está configurado, o caminho seguro é fazer o que ajuda **a máquina e a pessoa ao mesmo tempo**.

- **Texto real e selecionável.** Se você consegue selecionar e copiar o texto do seu PDF, um sistema provavelmente também consegue.
- **Estrutura simples.** Títulos de seção comuns (Resumo, Experiência, Projetos, Formação, Competências), uma coluna, datas no mesmo formato. Informação importante não fica escondida em caixas de texto, cabeçalhos do editor ou imagens.
- **O vocabulário da vaga, quando ele é verdade.** Se a vaga diz "PostgreSQL" e você usou PostgreSQL, escreva "PostgreSQL", e não só "banco de dados relacional". Isso ajuda um filtro por palavra e ajuda quem lê com pressa. Se você usou RabbitMQ e a vaga pede Kafka, escreva RabbitMQ.
- **Perguntas do formulário respondidas com atenção.** Leia cada pergunta eliminatória duas vezes. Um "não" apressado encerra a candidatura antes de qualquer leitura.

ERRO COMUM

Texto branco escondido com palavras-chave. É um truque que circula com frequência: colar a descrição da vaga em letra branca no rodapé. Quando o sistema extrai o texto, ou quando alguém seleciona o arquivo, as palavras aparecem. A candidatura passa a ser lembrada pelo motivo errado. E, mesmo que algum filtro fosse enganado, a etapa seguinte é humana.

O que a triagem procura

Juntando as duas leituras, a triagem tenta responder a poucas perguntas:

Pergunta de quem lê	Onde a resposta deve estar
Que tipo de profissional é esta pessoa?	Título logo abaixo do nome e primeira frase do resumo
Ela tem os obrigatórios da vaga?	Tópicos de experiência ou de projetos, com as tecnologias citadas
Em que nível ela está?	O tipo de trabalho descrito: executar, decidir, conduzir
Existe evidência, ou só afirmação?	Projetos com link, resultados concretos, escopo claro
Algo impede a contratação?	Localização, modelo de trabalho, idioma, disponibilidade

Repare que nenhuma dessas perguntas é "a pessoa é proativa?" ou "ela é apaixonada por tecnologia?". Adjetivos sobre você mesmo quase não pesam, porque qualquer um pode escrevê-los. O que pesa é **evidência**.

O que está sob o seu controle

Você não controla quantas pessoas se candidatam, qual sistema a empresa usa nem o humor de quem lê. Controla quatro coisas:

1. **Clareza.** Quem lê entende em poucos segundos que tipo de profissional você é e para que vaga se candidatou.
2. **Evidência.** Cada afirmação importante vem com algo concreto: um projeto, um resultado, um escopo.
3. **Aderência.** O que a vaga mais valoriza aparece primeiro.
4. **Coerência.** Currículo, LinkedIn e GitHub contam a mesma história, com as mesmas datas.

Os próximos capítulos tratam de cada uma dessas quatro coisas.

Em resumo

- O currículo costuma passar por uma leitura rápida, de quem verifica a aderência, e depois por uma leitura técnica, de quem verifica a profundidade.
- ATS é o sistema que organiza candidaturas. Ele pode extrair texto, filtrar e aplicar perguntas eliminatórias, mas a configuração varia e não é pública.
- Números como "6 segundos" e "75% descartados pelo robô" circulam sem método confiável. Ignore os números e fique com a ideia: facilite a leitura.
- Texto selecionável, estrutura simples e o vocabulário honesto da vaga ajudam a máquina e a pessoa.

CAPÍTULO 2

Estrutura, tamanho e formato

Um bom currículo de dev é, antes de tudo, fácil de ler. Isso não depende de design sofisticado. Depende de uma ordem lógica, de seções com nomes que todo mundo reconhece e de um arquivo que abre direito em qualquer lugar. Este capítulo monta o esqueleto. O conteúdo de cada linha fica para o capítulo 3.

Uma página ou duas?

Como **recomendação geral** para vagas no Brasil:

- **Estágio e júnior:** uma página. Com pouca experiência, uma segunda página quase sempre é enchimento.
- **Pleno:** uma página, ou duas se a experiência relevante não couber sem cortar o que importa.
- **Sênior:** até duas páginas. Mais do que isso tende a esconder o que importa.

O critério não é o número de páginas, é a densidade. Se a segunda página só tem cursos de 4 horas e uma lista de ferramentas, ela atrapalha. Se tem a experiência mais relevante para a vaga, ela é necessária. Para vagas no exterior, pesquise o costume do país, porque ele muda.

A ordem das seções

A regra é colocar primeiro a seção com as **evidências mais fortes para a vaga**. A tabela abaixo é um ponto de partida, não uma lei.

Seção	Júnior com estudo	Júnior em transição	Pleno e sênior
Cabeçalho	1	1	1
Resumo	2, curto	2, recomendado	2
Projetos	3	3	Só se forem relevantes
Experiência	4, se houver	5, resumida	3
Formação e cursos	3 ou 4	4	Final, uma ou duas linhas
Competências técnicas	Final	Final	Final, curta
Idiomas	Final	Final	Final

Para **estágio**, a formação sobe para logo depois do cabeçalho, porque a modalidade exige matrícula e frequência num curso. Informe curso, instituição, semestre atual, previsão de conclusão e turno.

O QUE DIZ A LEI

A Lei do Estágio (Lei nº 11.788/2008) exige que o estagiário esteja matriculado e frequentando curso de educação superior, profissional, de ensino médio ou outros previstos na lei. Quem já se formou e não está matriculado não pode ser contratado como estagiário. Por isso a Ana, que concluiu a graduação em 2018, procura vagas de júnior, e não de estágio.

O cabeçalho

O cabeçalho ocupa três ou quatro linhas e responde: quem é, que tipo de profissional, onde está e como falar com você.

ANA SOUZA
Desenvolvedora Backend Júnior (Python)
Belo Horizonte, MG · remoto · ana.souza@email.exemplo · (31) 90000-0000
github.com/anasouza-exemplo · linkedin.com/in/anasouza-exemplo

- **Nome** como você é chamado profissionalmente. Nome social é o seu nome.
- **Título-alvo** logo abaixo: a função que você busca, sem inventar cargo. "Desenvolvedora Backend Júnior (Python)" é como a Ana se apresenta, não um emprego que ela teve.
- **Cidade e estado**, e o modelo que você aceita (remoto, híbrido, presencial). O endereço completo não é necessário.
- **E-mail** com o seu nome. Evite apelidos antigos.
- **Telefone** com DDD.
- **Links** que funcionam e levam a algo que você quer mostrar: LinkedIn com URL personalizada, GitHub, portfólio.

Dados pessoais: só o necessário

O currículo circula por e-mails, sistemas e pessoas que você não controla. Cada dado a mais é um dado exposto. E a maioria deles não diz nada sobre o seu trabalho.

Não coloque:

- **CPF, RG, CNH ou número de documentos.** A empresa vai pedir na contratação. Antes disso, só facilitam fraudes.
- **Endereço completo.** Cidade e estado bastam.
- **Foto.** Não é exigida em currículo no Brasil e, como recomendação geral, pode ficar de fora. No LinkedIn é diferente (capítulo 6).
- **Data de nascimento ou idade, estado civil, filhos.** Não dizem nada sobre o trabalho e abrem espaço para viés.
- **Religião, opinião política, filiação a sindicato, saúde, orientação sexual.** São dados que a LGPD trata como sensíveis.

- **Pretensão salarial**, a não ser que a vaga peça. Se pedir, coloque no campo do formulário ou na mensagem.

O QUE DIZ A LEI

A **LGPD** (Lei nº 13.709/2018) adota o **princípio da necessidade**: o tratamento de dados pessoais deve se limitar ao mínimo necessário para a finalidade. A lei também define **dados sensíveis**, como origem racial ou étnica, convicção religiosa, opinião política, filiação a sindicato, dados de saúde e de vida sexual, que têm proteção reforçada. Já a **Lei nº 9.029/1995** proíbe práticas discriminatórias no acesso ao emprego por motivo de sexo, origem, raça, cor, estado civil, situação familiar, deficiência, idade, entre outros. Você não é obrigado a oferecer essas informações no currículo. Isto é uma descrição geral, não aconselhamento jurídico.

Sobre **deficiência**, a decisão de informar é sua. Se você se candidata a vagas reservadas para pessoas com deficiência, siga as instruções do processo, que costumam pedir a informação e o laudo numa etapa própria. Se precisar de adaptação para participar das entrevistas, você pode pedir.

Formato do arquivo

Estas são **recomendações gerais**. Se a vaga pedir algo diferente (um formulário, um formato, um idioma), siga a vaga.

- **PDF com texto selecionável**, exportado de um editor de texto ou de documentos. Nunca uma foto ou um escaneado.
- **Uma coluna**. Layouts em duas colunas, com barras laterais e ícones, podem ficar bonitos, mas às vezes se embaralham quando um sistema extrai o texto. E ficam ruins na tela do celular, onde muita gente lê.
- **Títulos de seção comuns**: Resumo, Experiência, Projetos, Formação, Competências técnicas, Idiomas. Nomes criativos ("Minha jornada") confundem pessoas e sistemas.
- **Fonte comum e legível**, em tamanho confortável. Margens normais. Negrito com moderação.
- **Datas em formato único**, como "fev/2019 – jun/2025". Mês e ano bastam.
- **Nada de barras, estrelas ou porcentagens de habilidade**. "Python ●●●○" não quer dizer nada que alguém possa verificar.
- **Nome do arquivo** com o seu nome e o foco: `ana-souza-backend-python.pdf`. Nunca `curriculo-final-v3-agora-vai.pdf`.

DICA

Teste o seu PDF: abra, selecione tudo (Ctrl+A), copie e cole num bloco de notas. Se o texto sair na ordem certa e completo, o arquivo está bom para pessoas e para sistemas. Se sair embaralhado ou vazio, refaça o layout.

Idioma

Se a vaga está em português, o currículo vai em português. Se a vaga está em inglês e o trabalho será em inglês, envie em inglês, com o **mesmo conteúdo** da versão em português. Não traduza com exagero: se o seu inglês é de leitura técnica, diga isso na seção de idiomas, porque a entrevista vai

mostrar a verdade.

Um currículo base e versões por vaga

Mantenha uma **versão base** completa, com tudo o que é verdade e relevante. Para cada vaga importante, faça uma cópia e **reordene**: o que a vaga mais valoriza sobe, o que não interessa desce ou sai. Os fatos não mudam entre versões. Datas, cargos e empresas são sempre os mesmos.

Uma forma simples de fazer isso é marcar, na descrição da vaga, os requisitos obrigatórios e os desejáveis. Depois, para cada um, aponte onde está a evidência no seu currículo. Requisito sem evidência não se resolve inventando; resolve-se com um projeto, um estudo declarado ("Estudando: Docker") ou, às vezes, com a decisão de não se candidatar àquela vaga.

Em resumo

- Uma página para júnior; até duas para quem tem mais experiência, se o conteúdo justificar.
- A ordem das seções segue a força das evidências. Em transição, projetos vêm antes da experiência anterior.
- Cabeçalho com nome, título-alvo, cidade, contato e links. Sem CPF, foto, idade, estado civil ou endereço completo.
- PDF de texto selecionável, uma coluna, títulos de seção comuns e nome de arquivo que identifica você.
- Fatos iguais em todas as versões; só a ordem e a ênfase mudam por vaga.

CAPÍTULO 3

Tópicos que mostram o que você fez

A parte que mais muda a qualidade de um currículo é a menos visível: a escrita de cada linha. Dois currículos com a mesma experiência podem passar impressões opostas. Um lista tarefas, como se fosse a descrição do cargo. O outro mostra o que a pessoa fez, em que contexto e o que mudou por causa disso. Este capítulo ensina a escrever o segundo tipo.

A fórmula: verbo + o quê + resultado

Cada tópico de experiência ou de projeto pode seguir uma estrutura simples:

1. **Um verbo de ação**, na primeira pessoa implícita: implementei, criei, reescrevi, automatizei, investiguei, conduzi, documentei, migrei, testei.
2. **O quê e em que contexto**, com o detalhe técnico que alguém da área precisa para entender a dificuldade: qual sistema, com que tecnologia, para quem.
3. **O resultado**: o que mudou. Pode ser um número que você mediu, uma consequência observável ("os chamados sobre isso pararam") ou uma adoção ("passou a ser usado por outros dois times").

Nem todo tópico precisa das três partes com o mesmo peso. Mas, se um tópico não tem verbo de ação nem resultado, ele provavelmente descreve um cargo, e não você.

DICA

Para cada tópico, faça o teste da entrevista: se alguém apontar para essa linha e disser "me conte mais", você tem dois minutos de conteúdo? Se não tiver, a linha está inflada ou é irrelevante. Se tiver dez minutos, talvez ela mereça subir.

Trabalho em equipe: delimite a sua parte

Quase todo software é feito em equipe. "Participei do desenvolvimento do novo checkout" pode significar que você desenhou a arquitetura ou que corrigiu dois textos. Na dúvida, quem lê imagina a versão menor.

A solução é dizer **o que foi coletivo e o que foi seu**: "com mais dois colegas", "num time de cinco pessoas, fiquei responsável por...". Também vale o contrário: não escreva "liderei" se você não liderou. Diga o que fez, e o leitor entende o nível.

Números: só os que você mediu

Números chamam atenção, e por isso são perigosos. "Reduzi a latência em 80%" convida a pergunta "80% de quê, medido como?". Se você não sabe responder, o número prejudica em vez de ajudar.

Regras práticas:

- Use só números que você **mediu** ou conhece de **fonte confiável**, e saiba explicar como foram obtidos.

- Se não mediu, descreva a consequência concreta: "o relatório deixou de estourar o tempo limite no fim do mês".
- Não use números confidenciais do empregador, como faturamento ou base de clientes, sem ter certeza de que são públicos.
- Em projeto pessoal, seja literal: "na minha máquina, com 10 mil linhas de dados de exemplo".
- **Nunca invente um número para enfeitar.** Um resultado verdadeiro e sem número vale mais do que um percentual inventado.

Oito reescritas, da mais fraca à mais forte

Os pares abaixo mostram transformações comuns. O objetivo é entender o que muda, não copiar as frases.

1. Responsabilidade em vez de realização

ANTES

Responsável pelo desenvolvimento e manutenção de APIs.

DEPOIS

Desenhei e mantenho a API de pedidos consumida por parceiros de integração (Python, FastAPI, PostgreSQL), com versionamento e documentação OpenAPI.

"Responsável por" descreve o cargo e poderia estar no anúncio da vaga. A versão melhor diz qual API, quem usa, com o que foi feita e um detalhe (versionamento) que mostra cuidado com quem consome. É um tópico do Bruno.

2. "Participei" escondendo a contribuição

ANTES

Participei da migração do sistema de filas.

DEPOIS

Conduzi, com mais dois colegas, a troca da fila baseada em tabela do PostgreSQL por RabbitMQ: levantei os problemas do modelo antigo, comparei duas alternativas, escrevi a proposta e fiz a migração gradual, sem parada do sistema.

A versão melhor separa o que foi coletivo (a migração) do que foi do Bruno (levantamento, comparação, proposta). Ele não diz "liderei a equipe". Diz o que fez.

3. Número sem base

ANTES

Melhorei a performance do sistema em 90%.

DEPOIS

Reescrevi as consultas do relatório de conciliação de pagamentos e criei índices compostos; o tempo de geração caiu de alguns minutos para poucos segundos, medido em produção antes e depois.

"O sistema" é vago e "90%" não tem referência. A melhor diz o que foi otimizado, como, e qual foi o efeito, com uma medida que o Bruno fez e sabe explicar.

4. Lista de tarefas de outra área**ANTES**

Lançamentos contábeis, conciliações bancárias, apuração de impostos, atendimento a clientes e rotinas administrativas.

DEPOIS

Conciliação bancária e fechamento mensal para cerca de 30 clientes do escritório. Automatizei a conferência de extratos com macros e, depois, scripts em Python, que a equipe passou a usar.

A versão fraca é uma descrição de cargo, sem nada que interesse a uma vaga de tecnologia. A melhor seleciona o que é **transferível**: domínio de conciliação, volume e iniciativa de automação. E não chama scripts internos de "sistema". O "cerca de 30" só entra porque a Ana sabe que é verdade.

5. Projeto pessoal com promessa grande demais**ANTES**

Sistema completo de gestão financeira com Python.

DEPOIS

Concilia (projeto pessoal): ferramenta de linha de comando em Python que compara extrato bancário e lançamentos em CSV; valores em centavos para evitar erro de arredondamento e testes com pytest para duplicidades e arquivos vazios.

"Sistema completo de gestão" promete mais do que o projeto entrega. A melhor é específica, diz que é pessoal e mostra duas decisões técnicas que a Ana sabe defender.

6. Lista de tecnologias no lugar de trabalho**ANTES**

Desenvolvimento frontend com React, TypeScript, Redux, Jest, Storybook, Webpack, Git e metodologias ágeis.

DEPOIS

Criei o design system da empresa (React, TypeScript, Storybook), hoje usado por três times de produto; defini com o time de design o processo de proposta e revisão de novos componentes.

A versão fraca diz quais ferramentas existiam no projeto, não o que a Carla fez com elas. A melhor mostra **escopo** (três times) e um processo que ela criou. As ferramentas aparecem entre parênteses, no lugar certo.

7. Mentoria sem conteúdo

ANTES

Mentoria de desenvolvedores e liderança técnica.

DEPOIS

Mentoro duas pessoas desenvolvedoras júnior, com encontros quinzenais e revisão de código comentada; uma delas passou a manter sozinha os componentes de formulário do design system.

"Liderança técnica" aparece em quase todo currículo sênior e, por isso, não diferencia. A melhor diz o formato da mentoria e um efeito observável. O mérito do crescimento é da pessoa mentorada; a Carla relata o que fez e o que viu.

8. Projeto de curso feito em grupo

ANTES

Desenvolvi um e-commerce completo no bootcamp.

DEPOIS

Projeto final do curso, em equipe de quatro pessoas: implementei os endpoints de relatório da API e os testes com pytest.

Muitos colegas de turma terão o mesmo projeto. O que diferencia você é dizer **o que foi seu**. E "completo" sai, porque quase nenhum projeto de curso é.

Verbos que ajudam (e expressões que atrapalham)

Prefira	Evite
implementei, criei, escrevi	responsável por
reescrevi, otimizei, migrei	participei de, atuei em
investiguei, corriji, diagnostiquei	auxiliei, dei suporte
automatizei, documentei, testei	trabalhei com
conduzi, propus, mentorei	liderei (se não liderou)

"Auxiliei" e "dei suporte" não são proibidos. Às vezes é exatamente o que aconteceu, e honestidade vem antes de estilo. Só não use essas expressões quando você fez mais do que elas dizem.

Competências técnicas: menos e com contexto

A seção de competências costuma virar um depósito de tudo o que você já tocou. Uma lista longa e sem hierarquia esconde o que você domina e coloca na mesa assuntos que podem aparecer na entrevista sem você estar preparado.

FRACA

Python, Java, JavaScript, TypeScript, C, PHP, Django, FastAPI, Flask, Spring, React, Vue, Angular, Node, PostgreSQL, MySQL, MongoDB, Redis, Docker, Kubernetes, AWS, GCP, Azure, Terraform, Kafka, RabbitMQ, Git, Linux, Scrum

MELHOR

Uso diário: Python (FastAPI, Django), PostgreSQL, RabbitMQ, pytest, Docker, Git

Uso frequente: Kubernetes como usuário (manifestos e diagnóstico), Prometheus, Grafana

Já usei: Redis (cache de sessões), Node.js (scripts internos)

A versão melhor é do Bruno e cabe em três linhas. Agrupar por frequência de uso é honesto e útil: quem lê sabe onde perguntar fundo. Kafka não aparece, porque ele só viu num curso. Se você está estudando algo que a vaga pede, diga isso numa linha separada: "Estudando: Docker (uso local nos projetos pessoais)".

ERRO COMUM

Colocar "Office", "Windows", "internet" ou "trabalho em equipe" na lista de competências técnicas de dev. Ocupa espaço e passa a impressão de que faltava o que escrever.

Em resumo

- Cada tópico: verbo de ação, o quê e em que contexto, e o resultado.
- Em trabalho de equipe, diga o que foi coletivo e o que foi seu.
- Use só números que você mediu e sabe explicar. Um resultado concreto sem número é melhor que um percentual inventado.
- Competências técnicas curtas, agrupadas por profundidade e coerentes com os tópicos acima delas.

CAPÍTULO 4

Sem experiência na área: projetos e transição

"Pedem experiência para dar experiência." Quem procura a primeira vaga ouve isso o tempo todo. A saída não é inventar experiência. É mostrar **evidência** de outra forma: projetos que alguém pode abrir, estudos com escopo claro e uma trajetória anterior apresentada pelo que ela tem de útil. Este capítulo cobre as três coisas e o resumo profissional, que amarra tudo.

A seção de projetos

Para quem não tem experiência como dev, a seção de projetos é a mais importante do currículo. Ela vem logo depois do resumo, antes da formação e da experiência anterior.

Escolha **dois ou três** projetos. Mais do que isso dilui. Prefira os que:

- resolvem um **problema pequeno e real**, de preferência um que você viveu;
- têm **README** que explica o que é, como rodar e por que você tomou as decisões que tomou;
- têm **testes**, mesmo poucos;
- usam as tecnologias da vaga.

Um clone de rede social feito seguindo um tutorial passo a passo diz pouco, porque milhares de pessoas têm o mesmo. Um projeto pequeno, com uma decisão técnica sua e uma limitação reconhecida, diz muito mais.

Como descrever um projeto

Cada projeto ganha um título com o rótulo honesto, o período e dois ou três tópicos:

```
Concilia – conciliação de extrato bancário (projeto pessoal, 2025-2026)
- Ferramenta de linha de comando em Python que compara extrato bancário e
  lançamentos em CSV e lista o que casou e o que sobrou de cada lado.
- Valores em centavos (inteiros) para evitar erro de arredondamento; tolerância
  de datas configurável para compensações que atrasam.
- Testes com pytest para duplicidades, lançamentos ausentes e arquivos vazios;
  README com decisões técnicas e limitações conhecidas.
```

Repare em três coisas. O rótulo "projeto pessoal" evita qualquer ambiguidade. Cada tópico traz uma decisão que a Ana sabe explicar. E não há promessa de impacto que não existiu.

Sem fingir usuários ou impacto

Como currículos de pessoas experientes falam em "reduziu o tempo de resposta" ou "atendeu milhares de usuários", quem está começando sente que precisa de números parecidos. Daí nascem frases como "plataforma usada por centenas de pessoas" para um projeto que só o autor usou.

Isso é inventar experiência. Uma pergunta simples ("como vocês mediam esses usuários?") desmonta a frase, e tudo o que vier depois será lido com desconfiança. Em vez de impacto comercial, mostre:

- **O problema e por que ele importa para você.** "Eu fazia essa conferência à mão toda semana."
- **O que você decidiu ou aprendeu.** "Troquei float por centavos inteiros depois que um teste falhou por arredondamento."
- **O que é verificável.** Testes, casos de borda tratados, desempenho medido na sua máquina, dito assim.
- **Uso real, no tamanho real.** "Uso toda semana nas minhas finanças pessoais" ou "duas ex-colegas testaram e me deram retorno."

Contribuição em código aberto

Uma contribuição pequena aceita num projeto de código aberto, como corrigir documentação ou um bug simples, mostra que você consegue ler código alheio e seguir o processo de revisão de outra equipe. Vale uma linha, com o que você fez:

Contribuição em código aberto (2026)

- Corrigi a leitura de arquivos CSV com separador de ponto e vírgula numa biblioteca de leitura de CSV; pull request revisado e incorporado.

Projeto com ajuda de IA

Se você usou assistentes de IA para escrever parte do código, não esconda e, principalmente, **entenda cada linha** que está no repositório. Em entrevista, "não sei por que esse trecho está aqui" pesa muito mais do que "usei um assistente para gerar o esqueleto e reescrevi a validação". A política sobre uso de IA varia por empresa.

Cursos, bootcamp e estudo por conta própria

Curso livre, bootcamp e estudo autodidata entram na seção de formação, com o nome certo de cada coisa. Curso livre não é graduação, e não precisa fingir ser. O que valoriza a linha é o **conteúdo**:

FORMAÇÃO

Curso livre de desenvolvimento backend com Python, 2025 (projeto final em equipe de quatro: implementei os endpoints de relatório e os testes da API)
Bacharelado em Ciências Contábeis, 2018

Uma lista de quinze certificados de cursos curtos não ajuda. Mantenha os que sustentam um requisito da vaga e deixe o resto para o LinkedIn, se quiser.

ERRO COMUM

Listar como "experiência profissional" os projetos de curso ou os estudos por conta própria. Projeto de curso é projeto; vai na seção de projetos, com o rótulo certo. Rotular errado é a forma mais rápida de perder a confiança de quem lê.

Transição de carreira: o passado como ativo

Quem vem de outra área tende a cometer um de dois erros opostos: esconder a carreira anterior, como se fosse vergonha, ou vendê-la como se fosse experiência em desenvolvimento. Nenhum dos dois funciona.

O que a experiência anterior não é: experiência de desenvolvimento. "Oito anos de experiência profissional, sendo dois em desenvolvimento" é honesto. "Oito anos de experiência" num currículo de dev não é.

O que ela pode ser:

- **Conhecimento de domínio.** Numa empresa de software financeiro, entender conciliação ou fechamento mensal é útil desde o primeiro dia. Quem veio da saúde entende prontuário; quem veio do varejo entende estoque.
- **Hábitos profissionais.** Prazos, atendimento, documentação, cuidado com dados sensíveis, lidar com erro que custa dinheiro.
- **Automação feita no trabalho.** Macros, planilhas avançadas, scripts. Conte pelo que é ("automatizei a conferência de extratos com macros e scripts em Python"), sem chamar de engenharia de software.

No currículo, a experiência anterior vem **depois** dos projetos e da formação, resumida em dois tópicos que destacam o que é transferível. Se a vaga é numa área próxima da sua anterior, isso vira vantagem, e vale dizer no resumo.

O resumo profissional

O resumo é o parágrafo curto no topo do currículo. Para quem tem pouca experiência, ele dá contexto: explica quem você é antes que a pessoa estranhe a falta de cargos de dev. Um bom resumo responde a três perguntas em duas a quatro frases:

1. **Que tipo de profissional você é** (área, nível, foco).
2. **Qual evidência sustenta isso** (uma ou duas realizações concretas).
3. **O que você busca**, ou onde contribui melhor.

ANTES

Profissional proativa, dedicada e apaixonada por tecnologia, em busca de uma oportunidade para crescer e aprender na área de desenvolvimento. Facilidade de aprendizado, trabalho em equipe e boa comunicação.

DEPOIS

Desenvolvedora júnior de backend com Python, em transição depois de seis anos como assistente contábil. Automatizei a conferência de extratos no escritório e, desde então, construí o Concilia, ferramenta de conciliação bancária com testes, e uma pequena API em FastAPI. Busco uma vaga em que o conhecimento de finanças ajude a entender o produto.

O resumo fraco tem dois problemas. Os adjetivos não podem ser verificados e aparecem em quase todo currículo. E "em busca de oportunidade para crescer" fala do que a Ana quer receber, não do que ela oferece. O melhor afirma o nível sem se diminuir, apresenta a transição como ativo e traz evidência que ela pode mostrar.

DICA

Teste o seu resumo trocando o seu nome pelo de outra pessoa da sua turma. Se o texto continuar verdadeiro para ela, está genérico demais. Um bom resumo só serve para você.

Lacunas no currículo

Um período sem trabalho formal não precisa de justificativa longa no currículo. Se ele foi usado para estudar, os projetos e o curso, com datas, já contam a história. Se foi por outro motivo (cuidar de alguém, saúde, maternidade ou paternidade, um negócio próprio que não deu certo), uma linha neutra basta, ou nenhuma. Você não precisa expor detalhes pessoais.

O que não vale é apagar ou esticar datas para esconder a lacuna. Datas que não batem com o LinkedIn chamam mais atenção do que a lacuna em si.

Em resumo

- Sem experiência como dev, a evidência vem de dois ou três projetos com README, testes e decisões que você sabe explicar.
- Rotule tudo com honestidade: projeto pessoal, curso livre, experiência anterior.
- Em transição, a carreira anterior entra como domínio e hábitos, nunca somada aos anos de desenvolvimento.
- O resumo diz quem você é, qual evidência sustenta isso e o que busca, sem adjetivos genéricos.

CAPÍTULO 5

Três currículos, antes e depois

Este é o capítulo em que as regras viram documento. Você vai ver o currículo da Ana, do Bruno e da Carla em duas versões: a que eles mandavam antes e a revisada para uma vaga específica. Depois de cada par, uma lista curta explica o que mudou e por quê.

Os três são **fictícios**: pessoas, empresas, datas e números foram criados para o livro. Os currículos aparecem em texto simples para facilitar a leitura. No arquivo real, você usaria a formatação do capítulo 2: uma coluna, títulos de seção em negrito, PDF com texto selecionável.

Ana: júnior em transição

A vaga. Pessoa Desenvolvedora Júnior Python numa empresa de software financeiro para pequenas e médias empresas, remoto. Obrigatórios: Python, SQL, Git, Django ou FastAPI, noções de testes. Desejáveis: experiência com produtos financeiros ou contábeis, Docker, inglês para leitura.

Antes

ANTES

[FOTO 3x4]
CURRÍCULO
Ana Paula Souza
Brasileira, casada, 32 anos · CPF 000.000.000-00
Rua Exemplo, 123, apto 45, Bairro Centro, Belo Horizonte, MG
anapaulinha_bh@email.exemplo · (31) 90000-0000

OBJETIVO
Busco uma oportunidade na área de tecnologia para crescer profissionalmente e colocar em prática meus conhecimentos. Sou proativa, dedicada e apaixonada por programação.

EXPERIÊNCIA PROFISSIONAL
Assistente Contábil – Escritório de contabilidade (2019 – 2025)
- Lançamentos contábeis, conciliações bancárias, apuração de impostos, atendimento a clientes e rotinas administrativas.

FORMAÇÃO
Bacharelado em Ciências Contábeis, 2018
Curso de Python (2025) · Curso de Git (2025) · Curso de HTML (2024) · Excel Avançado (2021) · Oratória (2020) · Primeiros Socorros (2017)

PROJETOS
Sistema completo de gestão financeira em Python.
API de despesas.

HABILIDADES
Python ●●●●● SQL ●●●●● Java ●●●●● React ●●●●● Docker ●●●●●
Pacote Office, Windows, trabalho em equipe, comunicação

REFERÊNCIAS
Disponíveis sob consulta.

Depois

DEPOIS

ANA SOUZA

Desenvolvedora Backend Júnior (Python)

Belo Horizonte, MG · remoto · ana.souza@email.exemplo · (31) 90000-0000

github.com/anasouza-exemplo · linkedin.com/in/anasouza-exemplo

RESUMO

Desenvolvedora júnior de backend com Python, em transição depois de seis anos como assistente contábil. Automatizei a conferência de extratos no escritório e, desde então, construí o Concilia, ferramenta de conciliação bancária com testes, e uma pequena API em FastAPI. Busco uma vaga em que o conhecimento de finanças ajude a entender o produto.

PROJETOS

Concilia – conciliação de extrato bancário (projeto pessoal, 2025-2026)

- Ferramenta de linha de comando em Python que compara extrato bancário e lançamentos em CSV e lista o que casou e o que sobrou de cada lado.
- Valores em centavos (inteiros) para evitar erro de arredondamento; tolerância de datas configurável para compensações que atrasam.
- Testes com pytest para duplicidades, lançamentos ausentes e arquivos vazios; README com decisões técnicas e limitações conhecidas.

API de despesas (projeto pessoal, 2026)

- API REST em FastAPI com SQLite para registrar e categorizar despesas; validação de entrada com Pydantic e documentação automática da API.
- Ainda sem autenticação; limitação registrada no README.

Contribuição em código aberto (2026)

- Corrigi a leitura de arquivos CSV com separador de ponto e vírgula numa biblioteca de leitura de CSV; pull request revisado e incorporado.

FORMAÇÃO

Curso livre de desenvolvimento backend com Python, 2025 (projeto final em equipe de quatro: implementei os endpoints de relatório e os testes da API)

Bacharelado em Ciências Contábeis, 2018

EXPERIÊNCIA ANTERIOR

Assistente Contábil – Escritório de contabilidade (fev/2019 – jun/2025)

- Conciliação bancária e fechamento mensal para cerca de 30 clientes.
- Automatizei a conferência de extratos com macros e, depois, scripts em Python, usados internamente pela equipe.

COMPETÊNCIAS TÉCNICAS

Python, FastAPI, pytest, SQL (SQLite e consultas em sistema contábil), Git.

Estudando: Docker (uso local nos projetos pessoais).

IDIOMAS

Inglês: leitura técnica.

O que mudou

- **Dados pessoais enxutos.** Saíram foto, CPF, estado civil, idade e endereço completo. Entrou o que importa: cidade, modelo de trabalho, contato e links. O e-mail passou a ser o nome dela.
- **Título-alvo no lugar de "Currículo".** Quem lê sabe em um segundo que tipo de profissional ela é.
- **Resumo no lugar do objetivo.** Saíram os adjetivos; entrou evidência e a ligação com a vaga (software financeiro).

- **Ordem pela força da evidência.** Projetos subiram para o topo, porque provam os obrigatórios (Python, FastAPI, testes, Git). A contabilidade desceu e virou "experiência anterior".
- **Projetos com conteúdo.** "Sistema completo de gestão" virou uma descrição específica e verificável, com rótulo de projeto pessoal e uma limitação declarada.
- **Cursos filtrados.** Ficou o curso que sustenta a vaga, com a parte dela no projeto em grupo. Oratória e primeiros socorros saíram.
- **Competências sem barras.** Java e React saíram (ela não sabe explicar). Docker virou "estudando", porque é desejável e ela só usou localmente.
- **"Referências sob consulta" saiu.** Se a empresa quiser, pede.
- **Cabe numa página.**

Bruno: pleno de backend para a LogiRota

A vaga. Desenvolvedor(a) Backend Pleno na LogiRota, software de logística, híbrido em Curitiba.

Obrigatórios: Python com FastAPI ou Django, APIs REST, PostgreSQL com modelagem e otimização de consultas, testes automatizados, Git e revisão de código. Desejáveis: mensageria (Kafka ou RabbitMQ), Docker e Kubernetes, observabilidade, experiência com alto volume de transações. Responsabilidades: APIs de rastreamento e de cálculo de frete, plantão em produção.

Antes

ANTES

BRUNO LIMA — DESENVOLVEDOR FULL STACK

Nascimento: 14/03/1997 · Solteiro

Curitiba, PR · brunolima.dev@email.exemplo · (41) 90000-0000

RESUMO

Desenvolvedor full stack com experiência em Python, Django, FastAPI, Node, React, Angular, Java, Docker, Kubernetes, AWS, PostgreSQL, MongoDB, Redis, Kafka e RabbitMQ. Focado em resultados e em entregar valor.

EXPERIÊNCIA

Plataforma de gestão para varejo (2022 - atual)

Desenvolvedor

- Responsável pelo desenvolvimento e manutenção de APIs.
- Participei da migração do sistema de filas.
- Melhorei a performance do sistema em 90%.
- Trabalhei com monitoramento.
- Atuei com testes automatizados.
- Code review.

Software house (2022)

Estagiário

- Desenvolvimento de sistemas web.

FORMAÇÃO

Bacharelado em Sistemas de Informação, 2022

COMPETÊNCIAS

Python, Java, JavaScript, TypeScript, PHP, Django, FastAPI, Flask, Spring, React, Angular, Node, PostgreSQL, MySQL, MongoDB, Redis, Docker, Kubernetes, AWS, GCP, Terraform, Kafka, RabbitMQ, Git, Linux, Scrum, Jira

Depois

DEPOIS

BRUNO LIMA

Desenvolvedor Backend Pleno (Python)

Curitiba, PR · híbrido ou remoto · bruno.lima@email.exemplo · (41) 90000-0000

github.com/brunolima-exemplo · linkedin.com/in/brunolima-exemplo

RESUMO

Desenvolvedor backend com cerca de quatro anos em Python e PostgreSQL, numa plataforma de gestão para varejo, mantendo APIs e integrações com meios de pagamento e marketplaces, com picos de volume em datas promocionais. Desenho APIs REST, otimizei consultas, conduzi a migração do nosso sistema de filas para RabbitMQ e cuido de métricas e alertas dos serviços do time.

EXPERIÊNCIA

Plataforma de gestão para varejo – produto próprio

Desenvolvedor Backend Pleno (jan/2024 – atual)

Desenvolvedor Backend Júnior (ago/2022 – dez/2023)

- Reescrevi as consultas do relatório de conciliação de pagamentos e criei índices compostos no PostgreSQL; o tempo de geração caiu de alguns minutos para poucos segundos, medido em produção antes e depois.
- Conduzi, com mais dois colegas, a troca da fila baseada em tabela do PostgreSQL por RabbitMQ: levantei os problemas, comparei duas alternativas, escrevi a proposta e fiz a migração gradual, sem parada do sistema.
- Criei as métricas de latência e de tamanho de fila do serviço de integrações (Prometheus e Grafana) e os alertas que o time usa no plantão.
- Desenhei e mantenho a API de pedidos consumida por parceiros de integração (FastAPI), com versionamento e documentação OpenAPI.
- Escrevi a maior parte dos testes de integração do serviço com pytest e um PostgreSQL real em container, o que permitiu atualizar dependências com menos receio de regressões.
- Reviso a maioria dos pull requests do time e acompanho um desenvolvedor júnior desde 2025.

Estagiário de Desenvolvimento – Software house (jan/2022 – jul/2022)

- Manutenção de sistemas web em Django para clientes da empresa.

FORMAÇÃO

Bacharelado em Sistemas de Informação, 2022

COMPETÊNCIAS TÉCNICAS

Uso diário: Python (FastAPI, Django), PostgreSQL, RabbitMQ, pytest, Docker, Git

Uso frequente: Kubernetes como usuário (manifestos e diagnóstico), Prometheus, Grafana

Já usei: Redis (cache de sessões), Node.js (scripts internos)

IDIOMAS

Inglês: leitura e escrita técnica; conversação intermediária.

O que mudou

- **Foco no lugar de "full stack".** O Bruno não quer vagas full stack. O título e o resumo agora dizem backend Python, que é o que a LogiRota procura.
- **O resumo fala de trabalho, não de ferramentas.** Desenhar APIs, otimizar consultas, cuidar da observabilidade. E liga o domínio dele ao da vaga pelo que têm em comum: volume e integrações. Ele não finge que conhece logística.
- **Tópicos reescritos com a fórmula do capítulo 3.** Cada um tem verbo, contexto e resultado. Os "90%" sem base viraram uma medida que ele sabe explicar.

- **Ordem pela vaga.** A vaga destaca PostgreSQL e operação em produção, então otimização de consultas, fila e métricas sobem. Nada foi inventado; só reordenado.
- **Progressão visível.** Júnior e pleno aparecem como duas linhas na mesma empresa, com datas.
- **Competências honestas.** Kafka, Terraform, Java, Angular e AWS saíram, porque ele não sustenta uma conversa sobre eles. Kubernetes ficou com a ressalva "como usuário".
- **Data de nascimento e estado civil saíram.**

Carla: sênior de frontend

A vaga. Pessoa Desenvolvedora Frontend Sênior (referência técnica) num SaaS B2B de gestão financeira, remoto. Obrigatórios: React e TypeScript em produção, design system ou biblioteca de componentes, acessibilidade na prática, desempenho web (Core Web Vitals), mentoria e revisão de código, colaboração com design e produto.

Antes

O currículo antigo da Carla tinha três páginas. Abaixo, o começo dele; o resto seguia igual, com cada emprego desde 2015 descrito no mesmo nível de detalhe.

ANTES

CARLA MENDES

Desenvolvedora Frontend Sênior

São Paulo, SP · carla.mendes@email.exemplo · (11) 90000-0000

PERFIL

Desenvolvedora frontend sênior com mais de 9 anos de experiência, especialista em React, com foco em resultados e qualidade. Já trabalhei em diversos projetos de grande porte.

EXPERIÊNCIA

Plataforma de e-commerce B2B (mar/2021 - atual)

Desenvolvedora Frontend Sênior

- Desenvolvimento frontend com React, TypeScript, Redux, Jest, Storybook, Webpack, Git e metodologias ágeis.
- Responsável pelo design system.
- Melhorias de acessibilidade, deixando o sistema 100% acessível.
- Otimização de performance.
- Mentoria de desenvolvedores e liderança técnica.
- Participação em entrevistas.

Empresa de tecnologia educacional (fev/2018 - fev/2021)

Desenvolvedora Frontend Pleno

- Desenvolvimento de telas em React.
- Manutenção de código legado em jQuery.
- Correção de bugs.
- Participação em cerimônias ágeis.

[... mais uma página e meia: agência, cursos, eventos, certificados ...]

Depois

DEPOIS

CARLA MENDES

Desenvolvedora Frontend Sênior · Design system e acessibilidade

São Paulo, SP · remoto · carla.mendes@email.exemplo · (11) 90000-0000

github.com/carlamendes-exemplo · linkedin.com/in/carlamendes-exemplo

RESUMO

Desenvolvedora frontend sênior com mais de dez anos de carreira, a maior parte com React e TypeScript. Nos últimos três anos, criei e mantenho o design system usado por três times de produto, com foco em acessibilidade e consistência. Conduzi a migração de uma aplicação legada para rotas carregadas sob demanda e mentoro duas pessoas desenvolvedoras. Busco um papel em que a influência técnica alcance mais de um time.

EXPERIÊNCIA

Plataforma de e-commerce B2B – produto próprio

Desenvolvedora Frontend Sênior (mar/2021 – atual)

- Criei o design system da empresa (React, TypeScript, Storybook), hoje usado por três times de produto; defini com o time de design o processo de proposta e revisão de novos componentes.
- Conduzi a revisão de acessibilidade dos componentes com base nas WCAG 2.2: navegação por teclado, foco visível, rótulos e contraste; os componentes de formulário passaram a ser testados com leitor de tela antes de cada versão.
- Conduzi a migração do painel de pedidos para rotas carregadas sob demanda; o LCP no percentil 75 da tela de pedidos caiu de cerca de 4 s para cerca de 2,3 s, medido com dados de campo nas quatro semanas antes e depois.
- Propus e implantei um orçamento de tamanho de pacote no processo de integração contínua, que bloqueia mudanças que ultrapassem o limite combinado.
- Mentoro duas pessoas desenvolvedoras júnior, com encontros quinzenais e revisão de código comentada; uma delas passou a manter sozinha os componentes de formulário.
- Participo como entrevistadora técnica nos processos seletivos de frontend.

Empresa de tecnologia educacional – produto próprio

Desenvolvedora Frontend Pleno (fev/2018 – fev/2021)

- Reescrevi o editor de exercícios em React com TypeScript, substituindo uma implementação em jQuery; escrevi os testes de componentes que passaram a ser exigidos no time.

Experiência anterior

Desenvolvedora Web Júnior – Agência digital (2015 – 2018): sites institucionais e primeiras aplicações em React.

FORMAÇÃO

Bacharelado em Ciência da Computação, 2015

COMPETÊNCIAS TÉCNICAS

React, TypeScript, Storybook, Testing Library, Playwright, acessibilidade web (WCAG 2.2, testes com leitor de tela), desempenho web (Core Web Vitals, análise de pacote), monorepo com pnpm workspaces. Node.js para BFF, uso ocasional.

IDIOMAS

Inglês: fluente.

O que mudou

- **De três páginas para duas.** O começo de carreira virou uma linha. As datas continuam lá.
- **O título carrega o foco da vaga:** design system e acessibilidade.

- **Escopo e efeito além do código.** Três times usam o design system, existe um processo com design, um orçamento de desempenho na integração contínua, mentoria com formato definido. Isso é o que separa sênior de pleno no papel.
- **Números com origem declarada.** "LCP no percentil 75", "dados de campo", "quatro semanas antes e depois". Quem entende de desempenho reconhece a métrica e sabe que ela pode explicar como mediu.
- **"100% acessível" saiu.** Ninguém deveria afirmar isso sem auditoria formal. Ela diz o que revisou, com base em quê, e o que passou a ser testado.
- **"Liderança técnica" virou fatos.** Ela não teve pessoas sob gestão formal e não sugere isso. Usa os verbos exatos: conduzi, propus, mentoro.
- **"Mais de 9 anos" virou "mais de dez anos",** porque o currículo antigo não tinha sido atualizado. Parece detalhe, mas é o tipo de coisa que uma pessoa atenta nota.

DICA

Peça a alguém da área que leia o seu currículo ao lado da vaga e responda em uma frase: "que tipo de profissional é esta pessoa e por que ela se candidatou?". Se a resposta não for a que você queria, reescreva o título, o resumo e os três primeiros tópicos.

Em resumo

- As três revisões seguiram as mesmas regras: dados pessoais enxutos, título-alvo, resumo com evidência, tópicos com a fórmula, competências honestas.
- A ordem mudou conforme a pessoa: projetos primeiro para a Ana, tópicos que a vaga destaca primeiro para o Bruno, escopo e efeito para a Carla.
- Em nenhum caso algo foi inventado. As versões "depois" dizem menos coisas, e cada coisa dita é verificável.

CAPÍTULO 6

LinkedIn: um perfil que se encontra e se lê

O currículo você envia. O LinkedIn é onde as pessoas **procuram** você, e onde vão conferir o que o currículo diz. Em outubro de 2026, é comum que pessoas recrutadoras façam as duas coisas: buscar perfis para uma vaga e abrir o perfil de quem se candidatou. Quanto isso pesa varia de empresa para empresa, mas uma regra não varia: **perfil e currículo não podem se contradizer**.

Os nomes de seções e botões abaixo são os da interface em português em outubro de 2026. A plataforma muda com frequência; se algo estiver em outro lugar, procure pelo nome da seção.

Currículo e perfil: o que muda

O currículo muda a cada vaga. O perfil é **um só** para todas. Por isso, ele tende a ser mais amplo:

	Currículo	LinkedIn
Alcance	Uma vaga	O seu alvo principal
Tamanho	Uma ou duas páginas	Pode ser mais completo
Experiências	Recorte do que importa	Todas as relevantes, com descrição
Tom	Terceira pessoa implícita, seco	Primeira pessoa, um pouco mais próximo
Foto	Não	Sim, se você quiser

O que **nunca** muda entre os dois: cargos, empresas e datas. Se o currículo diz que o Bruno entrou em agosto de 2022 e o LinkedIn diz 2021, alguém vai perguntar, e a conversa começa com uma explicação desnecessária.

Foto e imagem de capa

No currículo, a foto fica de fora. No LinkedIn, ela é comum e ajuda as pessoas a reconhecerem você numa conversa por vídeo. É opcional; a decisão é sua.

Se usar, prefira uma foto recente, com o rosto bem visível, fundo simples e boa luz. Não precisa de estúdio nem de terno. A **imagem de capa** (o retângulo atrás da foto) pode ficar com o padrão ou mostrar algo simples relacionado ao seu trabalho. Não é onde a decisão acontece.

URL personalizada

Por padrão, o endereço do perfil vem com números e letras aleatórias. Você pode trocar por algo com o seu nome, na opção de editar o perfil público e a URL. `linkedin.com/in/anasouza-exemplo` cabe no cabeçalho do currículo e é fácil de digitar.

Título

O título é a linha logo abaixo do seu nome. Ele aparece em buscas, em comentários e em convites. É o campo mais lido do perfil depois do nome. O limite é de cerca de 220 caracteres (confira na plataforma), mas um título bom costuma ser bem mais curto.

Por padrão, o LinkedIn preenche o título com o seu cargo atual. Para quem está em transição, isso vira "Assistente Contábil", que é exatamente o que a Ana **não** quer que um recrutador de tecnologia leia primeiro.

Uma fórmula que funciona: **função-alvo + especialidade ou tecnologia principal + um diferencial verdadeiro**.

ANTES

Ana Souza — Assistente Contábil | Estudante de programação | Apaixonada por tecnologia | Em busca de oportunidade

DEPOIS

Ana Souza — Desenvolvedora Backend Júnior · Python, FastAPI e SQL · vinda da contabilidade, com projetos de conciliação financeira

Outros exemplos:

Bruno: Desenvolvedor Backend Pleno · Python, FastAPI e PostgreSQL · APIs, mensageria e observabilidade em produção

Carla: Desenvolvedora Frontend Sênior · React e TypeScript · design system e acessibilidade

Repare que nenhum título usa "apaixonado", "ninja" ou "em busca de oportunidade". Um título diz o que você faz, não como você se sente.

DICA

Posso usar "Desenvolvedora" no título sem ter tido esse cargo? Sim, se você estuda, tem projetos e busca essa vaga. O título é a forma como você se apresenta, e o resto do perfil deixa claro que a experiência é de projetos. O que não pode é criar uma **experiência** com cargo de desenvolvedora numa empresa onde você não teve esse cargo.

Sobre

A seção "Sobre" é o seu resumo em versão um pouco mais longa, em primeira pessoa. O limite é de cerca de 2.600 caracteres, mas quase ninguém lê tudo. As **primeiras linhas** aparecem antes do "ver mais", então elas precisam dizer o essencial.

Uma estrutura simples, em três ou quatro parágrafos curtos:

1. Quem você é e o que faz (a frase mais importante).
2. Uma ou duas evidências concretas, com nomes de projetos ou resultados.

3. O que você busca e onde contribui melhor.
4. Tecnologias principais e como falar com você.

SOBRE – Ana

Sou desenvolvedora backend júnior com Python. Antes disso, trabalhei seis anos como assistente contábil, fazendo conciliação bancária e fechamento mensal para os clientes de um escritório.

Comecei a programar para automatizar a conferência de extratos, primeiro com macros e depois com scripts em Python que a equipe passou a usar. Gostei mais de construir a ferramenta do que de usar o resultado, e decidi mudar de área.

Desde então, fiz um curso livre de backend e construí o Concilia, uma ferramenta de linha de comando que compara extratos e lançamentos (valores em centavos, testes com pytest), e uma API de despesas em FastAPI. Os dois estão no meu GitHub, com README explicando as decisões e as limitações.

Procuro minha primeira vaga como desenvolvedora backend. Se for num produto financeiro, melhor ainda: conheço o domínio por dentro.

Python · FastAPI · SQL · pytest · Git

Experiência

No LinkedIn, liste **todas** as experiências relevantes, inclusive as de outra área. Para quem está em transição, apagar a carreira anterior cria uma lacuna estranha e esconde um ativo.

- **Cargo:** o nome real. Se o nome interno é pouco claro ("Analista III"), mantenha e acrescente a função entre parênteses: "Analista de Sistemas III (desenvolvimento backend)".
- **Datas:** as mesmas do currículo.
- **Descrição:** os tópicos do currículo base, com a fórmula do capítulo 3. Aqui você pode incluir os que não couberam na versão enviada.
- **Promoções:** o LinkedIn permite agrupar vários cargos na mesma empresa. Use isso para mostrar progressão, como o Bruno faz de júnior para pleno.

Projetos pessoais **não** são experiência. Eles vão na seção de **Projetos** (se disponível no seu perfil) e na seção **Em destaque**.

Em destaque

A seção "Em destaque" fica no alto do perfil e aceita links, publicações e arquivos. Para quem tem pouca experiência, ela é uma vitrine valiosa: coloque ali os dois ou três projetos principais, com link para o repositório ou para uma demonstração. Um texto que você escreveu explicando uma decisão técnica também cabe.

Competências

Você pode adicionar competências ao perfil e associá-las a cada experiência. As mesmas regras do currículo valem aqui: **prefira poucas e verdadeiras**. Escolha como principais as que estão na vaga que você quer. Uma lista de cinquenta competências, com metade delas vista só num tutorial, tem o

mesmo problema da lista longa no currículo.

Formação, licenças e certificados

Formação vai em "Formação acadêmica", com o nome correto do curso. Cursos livres e certificados vão em "Licenças e certificados". Aqui, diferente do currículo, você pode ser um pouco mais generoso: o perfil tem espaço. Ainda assim, um curso que nada tem a ver com a sua área não ajuda.

Aberto a trabalho

O LinkedIn tem um recurso para indicar que você está procurando emprego, com os cargos, locais e modelos de trabalho que você aceita. Em outubro de 2026, há duas opções de visibilidade:

- **Todos os membros do LinkedIn:** aparece para qualquer pessoa e adiciona um selo verde (#OpenToWork) na sua foto.
- **Somente recrutadores:** aparece para recrutadores que usam as ferramentas pagas de recrutamento da plataforma, sem selo. O LinkedIn diz que tenta não mostrar essa informação para recrutadores da sua empresa atual, mas **não garante** isso.

Se você está empregado e não quer que a empresa saiba, a segunda opção é a mais discreta, sabendo que não há garantia. Se está desempregado ou em transição, o selo é uma escolha pessoal: há quem prefira mostrar, há quem prefira não mostrar. Não existe resposta certa comprovada.

Preencha os cargos com os nomes que as vagas usam ("Desenvolvedor Backend", "Desenvolvedor Python") e os locais e modelos que você aceita de verdade.

Recomendações

Uma recomendação é um texto que outra pessoa escreve sobre o seu trabalho e que fica no seu perfil. Duas ou três recomendações específicas valem mais do que dez genéricas.

Peça a quem viu o seu trabalho de perto: uma ex-chefe, um colega de projeto, a pessoa que revisou o seu pull request, o professor do curso. Facilite: diga o que você gostaria que a pessoa mencionasse ("se puder, fale da automação de extratos que fizemos juntas"). E ofereça escrever uma para ela também, se fizer sentido.

ERRO COMUM

Trocar recomendações genéricas com desconhecidos ("recomendo, ótimo profissional!") só para preencher a seção. Quem lê percebe, e o texto não diz nada sobre o seu trabalho.

Atividade

Você não precisa virar criador de conteúdo. Mas um perfil sem nenhuma atividade em meses e um perfil com algumas publicações sobre o que você está construindo passam impressões diferentes.

Algumas formas simples de ter atividade com substância:

- Publicar quando terminar um projeto: o problema, uma decisão técnica, uma limitação. Com link.
- Comentar com conteúdo em publicações da sua área, em vez de só reagir.

- Escrever sobre algo que você aprendeu e que outra pessoa iniciante acharia útil.

Evite publicar por publicar, e cuidado com o que já está lá: quem abre o seu perfil também vê o que você comentou e compartilhou.

Coerência: a verificação final

Antes de mandar uma candidatura importante, abra o seu perfil como se fosse a pessoa recrutadora e leia ao lado do currículo que você vai enviar. Procure três coisas:

1. Datas, cargos ou empresas que não batem.
2. Um título muito distante da vaga.
3. Projetos citados no currículo que não aparecem no perfil ou no GitHub, ou que aparecem abandonados.

Em resumo

- O perfil é um só e mais amplo; o currículo é o recorte por vaga. Cargos, empresas e datas são iguais nos dois.
- Título com função-alvo, tecnologia principal e um diferencial verdadeiro. As primeiras linhas do "Sobre" dizem o essencial.
- Experiências reais com tópicos de resultado; projetos em "Em destaque" e em "Projetos", nunca como emprego.
- "Aberto a trabalho" pode ser público ou só para recrutadores, sem garantia de sigilo total.
- Poucas recomendações específicas e alguma atividade com conteúdo valem mais do que volume.

CAPÍTULO 7

GitHub: a vitrine do código

Para quem está começando, o GitHub muitas vezes é a única prova direta de que você sabe programar. Para quem já trabalha na área, ele pesa menos: a experiência descrita no currículo costuma dizer mais, e muita gente tem todo o código em repositórios privados da empresa. Este capítulo é, principalmente, para o primeiro grupo.

O GitHub não é o único lugar para mostrar código (GitLab, Codeberg e outros servem igualmente), mas é o mais comum em vagas de desenvolvimento em outubro de 2026. Os recursos da plataforma mudam; confira a documentação atual.

O que quem avalia faz

Quem abre o seu GitHub a partir do currículo raramente lê todo o código. Na maioria das vezes, a pessoa:

1. olha o topo do perfil e os repositórios fixados;
2. abre um ou dois projetos citados no currículo;
3. lê o começo do README;
4. talvez abra um arquivo de código ou de testes para ver o estilo.

Isso leva poucos minutos. O seu trabalho é fazer esses minutos renderem: o que você quer mostrar precisa estar no topo, e o README precisa responder às perguntas certas.

O perfil

- **Nome e foto.** Um nome reconhecível, de preferência o mesmo do currículo, facilita ligar o perfil a você. A foto é opcional.
- **Bio curta e link.** Uma frase ("Desenvolvedora backend júnior · Python") e o link para o LinkedIn ou portfólio.
- **Repositórios fixados.** O perfil permite destacar alguns repositórios no topo. Fixe os dois a quatro que têm README bom e que você consegue explicar em detalhe.
- **Organize a vitrine.** Exercícios antigos e projetos abandonados podem continuar públicos, só não precisam estar em destaque. Arquivar ou tornar privado o que não representa você hoje não é esconder o passado; é organizar.

O README de perfil

O GitHub permite criar um repositório com o **mesmo nome do seu usuário**. O README desse repositório aparece no topo da sua página de perfil. Duas ou três frases bastam:

```
# Olá, eu sou a Ana
```

```
Desenvolvedora backend júnior com Python, vinda de seis anos de contabilidade.
```

Gosto de ferramentas que tiram trabalho repetitivo das pessoas.

- ****Concilia****: conciliação de extrato bancário em linha de comando, com testes.
- ****API de despesas****: FastAPI + SQLite, documentação automática.

Agora estudando: Docker e PostgreSQL.

Contato: [linkedin.com/in/anasouza-exemplo](https://www.linkedin.com/in/anasouza-exemplo)

ERRO COMUM

README de perfil com dezenas de ícones de tecnologias, contadores de visitas, gráficos automáticos de "linguagens mais usadas" e frases motivacionais. Ocupa a tela inteira e não diz nada sobre o que você construiu. Prefira texto curto e links para os projetos.

O README de cada projeto

O README é a porta de entrada do projeto. Um bom README responde, nesta ordem:

1. **O que é e que problema resolve**, em duas ou três frases.
2. **Como rodar**, com comandos que funcionam numa máquina limpa.
3. **Como testar**.
4. **Decisões técnicas**, com o porquê ("valores em centavos para evitar erro de arredondamento com float").
5. **Limitações conhecidas** e próximos passos.

A seção de limitações assusta muita gente, mas ela trabalha a seu favor. Mostra que você sabe onde o projeto é fraco, e isso é exatamente o que se espera de alguém que vai trabalhar num código de verdade.

```
# Concilia – conciliação de extrato bancário
```

```
Compara o extrato do banco com os lançamentos do sistema contábil (ambos em CSV) e lista o que casou e o que sobrou de cada lado.
```

```
## Como rodar
```

```
python -m venv .venv && .venv/bin/pip install -e .  
concilia extrato.csv lancamentos.csv --tolerancia-dias 2
```

```
## Como testar
```

```
pytest
```

```
## Decisões técnicas
```

- Valores em centavos (int), nunca float.
- Tolerância de datas configurável: compensações podem atrasar.

```
## Limitações conhecidas
```

- Só lê CSV; não lê OFX nem PDF.
- Não trata transferências divididas em vários lançamentos.

DICA

Teste o seu README: clone o projeto numa pasta nova e siga as instruções ao pé da letra. Anote cada passo que faltou. Depois, peça a alguém de fora da área que leia o começo e diga, em uma frase, o que o projeto faz.

Commits e licença

O histórico de commits conta como o projeto evoluiu. Mensagens como "ajustes" repetidas cinquenta vezes não reprovam ninguém, mas mensagens descritivas ("trata extrato vazio", "usa centavos em vez de float") mostram cuidado. Não reescreva o histórico para fingir que o projeto foi feito de outro jeito.

Um repositório público sem licença não dá permissão de uso a ninguém. Se você quer que outras pessoas possam reaproveitar o código, escolha uma licença. O site choosealicense.com, mantido pelo GitHub, explica as mais comuns de forma simples.

O gráfico de contribuições

O gráfico de quadradinhos verdes mede atividade em repositórios, não qualidade nem competência. Há quem passe meses sem atividade pública porque trabalha em repositórios privados, cuida de alguém da família ou está estudando de outro jeito. Não force commits diários para preencher o gráfico. Quem avalia com atenção percebe, e quem não avalia com atenção não está olhando para isso.

O que nunca publicar

- **Código de empregador ou de cliente.** Mesmo que você tenha escrito, ele normalmente pertence a quem pagou. Se quiser mostrar o que aprendeu num trabalho, recrie o **problema** do zero, com outra implementação e dados inventados, e diga isso no README.
- **Senhas, chaves de API e tokens.** Se algo assim já foi para o histórico, apagar o arquivo num commit novo não basta: a credencial continua no histórico e precisa ser revogada e trocada.
- **Dados pessoais reais.** Planilhas com nomes, CPFs ou dados financeiros de pessoas reais não viram "dados de exemplo". Gere dados fictícios.
- **Solução de teste técnico de empresa que pediu sigilo.**

Em resumo

- Quem avalia olha o topo do perfil, os repositórios fixados e o começo do README. Faça esses minutos renderem.
- README de perfil curto, com links para os projetos. README de projeto com o quê, como rodar, como testar, decisões e limitações.
- Fixe só o que você sabe explicar. O gráfico verde não mede competência.
- Nunca publique código de empregador, segredos ou dados pessoais reais.

CAPÍTULO 8

Erros que eliminam na triagem

Nenhum erro isolado garante que você será descartado, e nenhum acerto garante a entrevista. Mas alguns erros aparecem com tanta frequência, e passam uma impressão tão ruim, que vale revisá-los um a um. A lista abaixo junta o que os capítulos anteriores trataram, mais alguns que não couberam em lugar nenhum.

Erros de conteúdo

- 1. Não dizer que tipo de profissional você é.** Um currículo sem título-alvo e com um resumo genérico obriga quem lê a adivinhar. Se a pessoa não descobre em poucos segundos que você é dev backend júnior, ela segue para o próximo.
- 2. Descrever cargos em vez de realizações.** "Responsável por", "atuei em", "participei de". Lido em sequência, parece a descrição da vaga, não você (capítulo 3).
- 3. Mentir ou inflar.** Ferramenta que você não sabe usar, cargo que não teve, número que não mediu, projeto de curso apresentado como emprego. Às vezes passa da triagem. Quase nunca passa da entrevista, e a partir daí nada do que você disser terá crédito.
- 4. Lista de tecnologias enorme.** Trinta ferramentas sem hierarquia escondem o que você domina e convidam perguntas sobre o que você não domina.
- 5. Ignorar os obrigatórios da vaga.** Se a vaga pede FastAPI ou Django e você usou FastAPI, mas escreveu só "frameworks web", quem filtra não encontra. Use o nome certo, quando ele é verdade.
- 6. Mandar o mesmo currículo para tudo.** Uma versão base é necessária. Mas, para as vagas que você mais quer, reordenar os tópicos pela vaga leva pouco tempo e muda bastante a leitura.

Erros de forma

- 7. Erros de português.** Um erro de digitação não elimina ninguém sozinho. Vários, num documento de uma página, passam a impressão de descuido. Leia em voz alta, use o corretor e peça a outra pessoa para revisar.
- 8. Layout que não sobrevive à extração.** Duas colunas, texto dentro de imagens, informações no cabeçalho do editor, ícones no lugar de palavras. Faça o teste de copiar e colar do capítulo 2.
- 9. Arquivo errado.** Documento do editor em vez de PDF (a não ser que a vaga peça), foto do currículo impresso, arquivo chamado `curriculo.pdf` entre outros cem com o mesmo nome.

10. Links quebrados. GitHub com usuário errado, LinkedIn com a URL antiga, portfólio fora do ar. Clique em cada link do PDF antes de enviar.

11. Tamanho desproporcional. Três páginas para um júnior, ou um sênior espremido em meia página. O tamanho segue o conteúdo relevante (capítulo 2).

Erros de dados pessoais

12. Excesso de dados pessoais. CPF, RG, endereço completo, data de nascimento, estado civil, foto. Nada disso é exigido no currículo, e documentos facilitam fraudes.

13. E-mail pouco profissional. O apelido da adolescência no endereço de e-mail é o primeiro dado que muita gente lê. Crie um com o seu nome.

Erros de coerência

14. Currículo e LinkedIn contando histórias diferentes. Datas que não batem, cargos com nomes diferentes, um projeto que só existe num dos dois.

15. Projeto citado e abandonado. O currículo fala de um projeto e o repositório está sem README, com o último commit dizendo "não funciona ainda". Ou arrume, ou não cite.

Erros na candidatura

16. Responder perguntas eliminatórias no automático. Um "não" apressado para disponibilidade de modelo híbrido encerra a candidatura antes de qualquer leitura.

17. Candidatar-se a vagas muito distantes do seu perfil. Mandar currículo de júnior para todas as vagas de sênior "para tentar" gasta o seu tempo e o de quem lê. Candidate-se quando você atende à maior parte dos obrigatórios, mesmo sem atender a todos os desejáveis.

18. Truques contra o filtro. Texto branco escondido, a descrição da vaga colada no rodapé, palavras-chave repetidas. Aparecem para quem abre o arquivo e queimam a candidatura (capítulo 1).

DICA

Imprima o currículo, ou abra no celular, e leia como se fosse de outra pessoa. Marque com um círculo cada palavra que você não conseguiria explicar numa entrevista. Tudo o que estiver circulado sai ou é reescrito.

Em resumo

- Os erros que mais pesam são de conteúdo: falta de foco, cargos em vez de realizações, exageros e listas enormes.
- Os de forma são fáceis de evitar: revisão, PDF de texto, links testados e tamanho proporcional.
- Dados pessoais em excesso e incoerência entre currículo e perfil passam impressão de descuido.
- Nenhum truque contra o filtro compensa: a etapa seguinte é sempre humana.

CAPÍTULO 9

O checklist final

Esta é a página para imprimir, ou para deixar aberta ao lado do currículo. Passe por ela antes de cada candidatura importante. Se algum item não estiver marcado, o capítulo entre parênteses explica o porquê e o como.

Currículo: conteúdo

- ☐ O título abaixo do nome diz a função que eu busco, sem inventar cargo (cap. 2).
- ☐ O resumo diz que tipo de profissional eu sou, traz uma evidência concreta e o que eu busco (cap. 4).
- ☐ A seção com as evidências mais fortes para esta vaga vem primeiro (cap. 2).
- ☐ Cada tópico começa com verbo de ação e diz o que **eu** fiz (cap. 3).
- ☐ Em trabalho de equipe, está claro o que foi coletivo e o que foi meu (cap. 3).
- ☐ Todo número tem origem que eu sei explicar, e nenhum é confidencial (cap. 3).
- ☐ Projetos têm rótulo honesto (pessoal, curso, em grupo) e link (cap. 4).
- ☐ As competências são poucas, agrupadas por profundidade e aguentam "me conte mais" (cap. 3).
- ☐ O que estou estudando aparece como "Estudando", e não como domínio (cap. 3).
- ☐ Os obrigatórios da vaga aparecem com o nome que a vaga usa, quando é verdade (cap. 1).

Currículo: forma

- ☐ PDF com texto selecionável; copiar e colar sai na ordem certa (cap. 2).
- ☐ Uma coluna, títulos de seção comuns, datas no mesmo formato (cap. 2).
- ☐ Uma página (júnior) ou até duas (com mais experiência) (cap. 2).
- ☐ Sem barras, estrelas ou porcentagens de habilidade (cap. 2).
- ☐ Nome do arquivo com meu nome e o foco, como `nome-sobrenome-backend.pdf` (cap. 2).
- ☐ Revisado em voz alta e por outra pessoa; sem erros de português (cap. 8).
- ☐ Todos os links clicados e funcionando (cap. 8).

Currículo: dados pessoais

- ☐ Só nome, cidade e estado, modelo de trabalho, e-mail, telefone e links (cap. 2).
- ☐ Sem CPF, RG, endereço completo, foto, idade, estado civil ou filhos (cap. 2).
- ☐ E-mail com o meu nome (cap. 8).

LinkedIn

- ☐ URL personalizada com o meu nome (cap. 6).
- ☐ Título com função-alvo, tecnologia principal e um diferencial verdadeiro (cap. 6).
- ☐ As primeiras linhas do "Sobre" dizem o essencial (cap. 6).

- ☐ Cargos, empresas e datas iguais aos do currículo (cap. 6).
- ☐ Experiências com tópicos de resultado; projetos em "Em destaque", nunca como emprego (cap. 6).
- ☐ Competências principais são as da vaga que eu quero (cap. 6).
- ☐ "Aberto a trabalho" configurado com cargos, locais e visibilidade que eu escolhi (cap. 6).
- ☐ Pelo menos uma recomendação específica de quem viu meu trabalho (cap. 6).

GitHub

- ☐ Nome reconhecível, bio curta e link para o LinkedIn (cap. 7).
- ☐ README de perfil curto, com links para os projetos (cap. 7).
- ☐ Repositórios fixados são os que eu sei explicar em detalhe (cap. 7).
- ☐ Cada projeto citado no currículo tem README com o quê, como rodar, como testar, decisões e limitações (cap. 7).
- ☐ Segui o README do zero numa pasta nova e funcionou (cap. 7).
- ☐ Nenhum segredo, código de empregador ou dado pessoal real no histórico (cap. 7).

Antes de clicar em "enviar"

- ☐ Li a vaga inteira, inclusive o modelo de trabalho e as condições.
- ☐ Respondi cada pergunta eliminatória com atenção.
- ☐ Alguém da área leu o currículo ao lado da vaga e entendeu o que eu quis dizer (cap. 5).
- ☐ Anotei a vaga, a data e a versão do currículo enviada numa planilha.

CAPÍTULO 10

Modelos para copiar

Os modelos abaixo são pontos de partida. Copie para o seu editor de texto, troque os colchetes pelo seu conteúdo e apague o que não se aplica. Um modelo preenchido sem pensar vira um currículo igual a mil outros; use a estrutura e escreva o conteúdo com as regras dos capítulos 2 a 4.

Modelo de currículo em texto simples

Este é o esqueleto de um currículo de uma página para júnior, com ou sem transição de carreira. Para quem já tem experiência como dev, mova a seção EXPERIÊNCIA para logo depois do RESUMO e encurte PROJETOS ou retire.

```
[NOME SOBRENOME]
[Função-alvo] ([tecnologia principal])
[Cidade], [UF] · [remoto/híbrido/presencial] · [email@seunome] · ([DDD]) [telefone]
[github.com/usuario] · [linkedin.com/in/usuario] · [portfólio, se houver]

RESUMO
[Que tipo de profissional você é: área, nível, foco.] [Uma ou duas evidências concretas: projeto, resultado, automação.] [O que você busca ou onde contribui melhor.]

PROJETOS
[Nome do projeto] – [o que é, em poucas palavras] ([pessoal/curso/grupo], [ano])
- [Verbo] [o quê] com [tecnologias]; [para quê ou para quem].
- [Uma decisão técnica sua e o porquê.]
- [Testes, documentação ou limitação conhecida.]

[Segundo projeto] – [...] ([rótulo], [ano])
- [...]

EXPERIÊNCIA
[Cargo real] – [Empresa ou tipo de empresa] ([mês/ano] – [mês/ano ou atual])
- [Verbo] [o quê, em que contexto]; [resultado que você sabe explicar].
- [Verbo] [o quê]; [resultado].

FORMAÇÃO
[Curso], [instituição], [ano de conclusão ou previsão]
[Curso livre ou bootcamp relevante], [ano] ([o que você fez no projeto final])

COMPETÊNCIAS TÉCNICAS
[Linguagens e ferramentas que você usa com segurança.]
Estudando: [ferramenta] ([o que você fez com ela até agora]).

IDIOMAS
[Idioma]: [nível real: leitura técnica, conversação, fluente].
```

DICA

Para estágio, a seção FORMAÇÃO sobe para logo depois do cabeçalho, com curso, instituição, semestre atual, previsão de conclusão e turno. Acrescente uma linha de disponibilidade: "Disponibilidade para estágio: 6 horas diárias, no turno da manhã."

Modelo de tópico

Para cada linha de experiência ou de projeto, preencha mentalmente as três partes antes de escrever:

Parte	Pergunta	Exemplo
Verbo	O que eu fiz?	Automatizei
O quê e contexto	Em quê, com o quê, para quem?	a conferência de extratos do escritório, com macros e depois Python
Resultado	O que mudou, e como eu sei?	a equipe passou a usar os scripts no fechamento mensal

Modelo de título do LinkedIn

[Função-alvo] · [tecnologia 1], [tecnologia 2] e [tecnologia 3]
· [diferencial verdadeiro]

Diferenciais verdadeiros podem ser um domínio ("vinda da contabilidade"), um foco ("APIs e observabilidade") ou um tipo de trabalho ("design system e acessibilidade"). Não use adjetivos sobre você.

Modelo de "Sobre" do LinkedIn

[Primeira frase: quem você é e o que faz. É a que aparece antes do "ver mais".]

[Como você chegou aqui, em duas ou três frases. Para transição: o que você fazia e o que fez você mudar.]

[Evidências: um ou dois projetos ou resultados, com nomes. Onde encontrar.]

[O que você busca agora e onde contribui melhor.]

[Tecnologia 1] · [Tecnologia 2] · [Tecnologia 3] · [Tecnologia 4]

Modelo de README de perfil do GitHub

```
# Olá, eu sou [nome]

[Função ou foco em uma frase.] [Algo verdadeiro sobre o que te interessa.]

- **[Projeto 1]**: [o que é, em uma linha].
- **[Projeto 2]**: [o que é, em uma linha].

Agora estudando: [tema].
Contato: [linkedin.com/in/usuario]
```

Modelo de pedido de recomendação

Oi, [nome]! Tudo bem? Estou procurando minha primeira vaga como [função] e queria pedir uma recomendação no LinkedIn, se você se sentir à vontade. Como trabalhamos juntos em [contexto], se puder mencionar [algo específico, como a automação de extratos], ajudaria bastante. Se quiser, escrevo uma para você também. Obrigada!

PRÓXIMO PASSO

E agora, a entrevista

Se você chegou até aqui e aplicou o checklist, o seu currículo diz quem você é, mostra evidências e não promete o que você não consegue sustentar. O seu LinkedIn e o seu GitHub contam a mesma história. Isso é o que está ao seu alcance para conseguir a conversa.

A conversa é a próxima etapa, e ela tem outras regras. Vêm a conversa com recrutador, a entrevista comportamental, o código ao vivo e o teste para fazer em casa. Para pleno e sênior, há system design. No fim, a proposta e a negociação.

Para quem quiser continuar, escrevi um livro sobre essa parte: ***Como Passar em Entrevistas para Desenvolvedor: do currículo à proposta***. Ana, Bruno e Carla estão lá também, agora diante das entrevistas. O livro tem respostas fracas e melhores comentadas, exercícios de algoritmos com solução e testes, system design, modelos de mensagem e um plano de preparação. Ele não promete aprovação, pelo mesmo motivo que este livro não promete: ninguém pode prometer isso com honestidade.

PRÓXIMA LEITURA

Como Passar em Entrevistas para Desenvolvedor: do currículo à proposta

Jefferson Valandro · R\$ 97 · na Hotmart

go.hotmart.com/G107935880A

Se este livro gratuito ajudou, a forma mais simples de retribuir é passá-lo adiante para alguém que está procurando a primeira vaga. A versão mais recente fica sempre em jeffev.github.io/jeffersonvalandro.

Boa sorte nas candidaturas.

Jefferson Valandro