Como é que será que o cara que criou o Cloud Code tá usando o Cloud Code? O nome dele é Boris e o criador da ferramenta resolveu dar as caras e trouxe 13 dicas de como ele mesmo usa o Cloud Code. Eu vou mostrar todas elas pra vocês aqui nesse vídeo. E essa aqui é a dica número 1. Então basicamente ele falou, eu rodo 5 clouds em paralelo, eu abro o meu terminal, numero cada um deles, 1, 2, 3, 4, 5, igual a gente está vendo aqui na imagem, e aí depois ele ativa as notificações do sistema quando o sistema termina de desenvolver. Algumas vezes você pode estar trabalhando na mesma base de código, em áreas diferentes, ou com tarefas diferentes que não vão entrar em conflito umas com as outras, ou até mesmo você pode estar utilizando uma aba especificamente para executar um agente de implementação de testes, um agente de back-end. Eu vejo muita gente usando o Cloud Code de uma maneira excepcional e que, às vezes, não tem a projeção que o Boris tem para falar, olha, tá aqui minhas dicas. Então, eu queria te convidar, você que usa Cloud Code, que tem uma dica marota, a deixar essa dica aqui nos comentários. Bora fazer nesse vídeo o nosso repositório nacional de dicas de Cloud Code para que outras pessoas que já estão começando ou que já estão mais avançadas consigam vir aqui nos comentários e falar, caraca, velho, o vídeo foi legal, as dicas do Boris foram boas, mas esses comentários estão de explodir a cabeça. Então, tem uma dica marota? Deixa aqui e bora pro conteúdo. Dica número 2. Ele tá basicamente falando que ele roda 5 instâncias do Cloud Code no cloud.ai, que é a plataforma em nuvem, a plataforma web, em paralelo com os Cloud Code locais. Caraca, então ele usa 5 locais, mas de 5 a 10. Então, ele pode ter até 15 instâncias de cloud trabalhando em simultâneo. Interessante. Então, basicamente, ele tá falando o seguinte. Ele usa também o terminal e várias vezes ele vai back and forth, ou seja, Ele usa tanto no computador dele quanto também na nuvem. Ele faz o teleporte do Cloud Code e isso parece ser uma estratégia que ele usa rotineiramente. Ele comentou também que ele usa o aplicativo do iOS para algumas vezes iniciar tarefas a partir do próprio celular dele. Então basicamente o que ele está fazendo aqui, ele está usando o ecossistema do Cloud como um todo, independente se está no ambiente local dele, se está no celular ou se está na nuvem. Dica número 3. I use Opus 4.5 with Thinking for Everything. Então ele usa o Opus 4.5 para tudo, sempre no modo Thinking, no modo pensamento, no modo raciocínio, certo? E aqui ele fala que é o melhor modelo de código que ele já usou. Apesar de ser maior e mais lento do que o SONET, ainda assim ele considera mais rápido no final das contas, porque os resultados que ele traz são mais objetivos. E ele tem o melhor uso também das ferramentas. Dica número 4. Nosso time compartilha o cloud.md, que é o arquivo de memória do Cloud Code, em todo o repositório do Cloud Code. E aí eles colocam isso no Git, de uma maneira que todo o time consegue contribuir múltiplas vezes em uma única semana. Qualquer vez que alguém vê alguma coisa do cloud que está agindo de maneira incorreta, eles adicionam no cloud.md. E aí o cloud entende, não somente para fazer isso dessa vez, mas para não repetir isso de novo. Então, basicamente, pelo que eu entendi, eles têm um único cloud.md

como arquivo de memória para o repositório inteiro, e todo mundo vai contribuindo dentro desse arquivo. E aí ele traz que outros times também mantenham seus próprios cloud.mds, e que é o trabalho de cada um, cada time, de manter esse arquivo atualizado. Eu acho muito interessante isso. Eu tenho um pé atrás de estabelecer isso como somente uma regra do cloud.md, porque isso te trava na ferramenta do Cloud Code. Mas falando de Cloud Code, uma baita sacada. Interessante. Interessante ver que os próprios criadores do Cloud Code não estão usando o Cloud MD em cada pasta e subpasta, e sim um único arquivo geral. Dica número 5. Durante os Code Reviews, eu geralmente marco o arroba Cloud. Dica número 5. Durante o Code Review, eu geralmente marco o arroba Cloud. nos comentários das pull requests dos meus colegas de trabalho, para adicionar alguma coisa ao arquivo cloud.md como parte da pull request. A gente usa o Cloud Code GitHub Action, você consegue instalar ele com o barra install GitHub Action, para isso. É a sua versão do Denshipper Compounding Engineering. Então, basicamente, o que ele está falando aqui é que durante o Code Review ele marca o próprio cloud para iniciar uma tarefa e adicionar alterações na documentação do cloud.md. Isso é muito interessante porque você não precisa pedir para o cara, dono da pull request, de fazer a alteração. Você já dá a tarefa para o próprio cloud para ele mesmo ir lá e fazer um commit com essa alteração. Dica número 6. A maioria das sessões eu começo no modo de planejamento, Shift Tab Twice. E se o meu objetivo for escrever uma pull request, eu vou usar o Plan Mode muitas vezes indo e voltando até que eu goste desse plano. A partir daí eu vou trocar e entrar no modo Auto Accept Edits, e aí o Cloud Code vai geralmente fazer isso em one shot, uma vezada só ele desenvolve o código. Um bom plano é realmente importante. Então aqui ele está falando que ele usa bastante a ideia que eu falo para vocês do Previz, então planejamento e revisão, planeja e revisa, revisa e revisa. até o plano ficar redondinho. E quando o plano está redondinho, ele vai para a execução do código. E, óbvio, com um bom plano, com uma boa Spec Driven Development, você consegue ter um resultado em um one shot. Dica número 7. Eu uso slash commands, os comandos em barra, para todo inner loop workflow that end up in doing many times a day. Então, toda a tarefa que ele faz várias vezes no dia, ele cria um atalho para esse comando. This saves me from repeated prompting and makes it so Cloud can use these workflows too. Então ele fala que isso economiza tempo dele de fazer coisas repetitivas e faz de uma maneira que o Cloud consegue também evitar essa repetição de tarefas. Os comandos são comitados, então são adicionados no .cloud comands e, por exemplo, o Cloud e ele usam o commit push PR o comando dezenas de vezes durante o mesmo dia, que é um comando que ele criou aqui, commit-push-pr. The command uses inline-bash to precompute git status and a few other pieces of info to make the command run quickly and avoid back and forth with the model. Então, basicamente, ele usa o comando em linha para pré-computar o git status and a few other pieces e alguns outros pedaços de informação para fazer o comando rodar mais rápido e evitar os ciclos mais longos durante a execução com o modelo. Bem interessante isso aqui e é realmente uma mão na roda para quem está usando bastante o Cloud Code fazer esses atalhos. Dica número 8. I use a few sub-agents regularly. Eu uso alguns sub-agentes regularmente. Ele tem o Code Simplifier, que simplifica o código depois que o Cloud terminou de trabalhar. Ele tem o Verify App, que detalha as instruções para o Cloud Coach testar end-to-end. E so on, e outros também, outros agentes. Então, basicamente, ele fala

similarmente como o Slash Command, ele pensa nos sub-agentes como maneiras de automatizar a maioria das tarefas comuns do dia-a-dia, da maneira de automatizar suas pull requests. Dica número 9. We use a post-to-use hook to format cloud code. Interessante, hein? Então, eles usam uma post-to-use, um hook do Cloud Code, para formatar o código do próprio Cloud Code. Cloud usually generates well-formatted code out of the box, and the hook handles the last 10% to avoid formatting errors in CI later. Então, basicamente, eles usam o Cloud Code para rodar esse hook sempre que termina uma tarefa para formatar o código, porque apesar do Cloud, na maioria das vezes, formatar certinho o código, tem 10% ali que algumas vezes não são formatados corretamente e assim eles evitam erros na CI. Boa dica, hein? Dica número 10. I don't use dangerously skip permissions. Eu não uso o modo YOLO do Cloud Code. Ao invés disso, eu uso o barra permissions para pré-permitir comandos comuns que eu já uso no meu dia a dia. A maioria desses comandos estão configurados dentro do arquivo cloud-settings.json and shared with the team. Então, a maioria desses comandos estão compartilhados com o time dentro do repositório. E está aqui uma coisa. Geralmente, e eu tenho certeza que a maioria das pessoas fazem isso, também é por isso, eu uso o Dangerous Skipermissions quando eu estou com preguiça de sair configurando todas as permissões. Mas pensando numa codebase que você vai gerenciar e várias pessoas vão trabalhar nela, ou até mesmo só você, mas que você vai constantemente trabalhar nela, É realmente mais interessante você configurar as suas permissões do que deixar tudo liberado. Porque, eventualmente, se você precisar de dar uma permissão a mais, você adiciona no seu arquivo de configurações do Cloud dentro do repositório, e aí ele já fica ali persistido no seu repositório. Ou seja, você não vai precisar de novamente ir lá e permitir. É mais seguro, de fato, do que liberar a IA fazer tudo. Você tem a segurança de menos alucinação de inteligência artificial. Boa dica, deixa aqui nos comentários o que você pensa disso aí. Dica número 11. As pessoas que deixam like nesse vídeo vão ter menos alucinações de IA no ano de 2026. Tô zoando, mas tô zoando não, hein? Mas deixa o seu like aqui, pô, dá aquela colaboração pra gente. Dica número 11. O Cloud Code geralmente vai usar todas as minhas ferramentas por mim. It often searches and posts to Slack. Geralmente ele vai no Slack, busca, comenta e posta também no Slack via o MCP Server do Slack. Runs BigQuery queries to answer analytical questions using BigQuery CLI, grabs error logs from Sentry, etc. Então basicamente ele está falando que o Cloud Code dele tem acesso a todas as ferramentas que ele usa, o BigQuery, o Slack, o Sentry, e que basicamente ele usa isso como uma integração para fazer tudo por ele. O que pode ser interessante Se você precisa de debugar um bug em produção que você tem os dados lá no Sentry do que aconteceu, você já conecta diretamente no Sentry e já traz as informações para a base de código. Se você tem um colega trocando uma ideia ou uma demanda que veio lá do Slack, você consegue conectar e trazer dentro da sua base de código. Eu, como eu estou mais agora com um time pequeno, talvez não faça tanto sentido. Mas para você que está em uma empresa que já tem uma base de código maior, tem uma gestão mais complexa de times, isso aqui pode ser uma baita mão na roda mesmo, viu? Dica número 12, para very long running tasks, para tarefas que demoram muito tempo para terminar, I will either prompt Cloud to verify its work with background agent when it's done. Então, eu vou ou pedir o Cloud Code para verificar, um Cloud Code, um background agent para verificar

quando está pronto, ou eu vou usar o agent stop hook to do That More Deterministically, ou seja, ele vai usar o hook stop para continuar isso de uma maneira mais determinística. Ou vai usar o Half Wigan plugin, que é um plugin que foi originalmente dreamed up by Geoffrey Huntley, não conheço esse plugin, e que também vai usar com o PermissionMode.ask ou o DangerouslySkipPermissions no modo sandbox para evitar problemas. problemas de prompts com permissão na sessão, de uma maneira que o Cloud consegue cozinhar sem ser bloqueado, sem depender dele, sem ter nenhum tipo de impedimento por ele. Daqui uma maneira interessante, eu não costumo usar o Cloud Code para long running tasks, como ele está mostrando aqui, tipo uma tarefa de um dia e duas horas, mas se você está usando, comenta aqui. Funciona assim para você? E na dica final, a dica número 13, ele fala que provavelmente é a dica mais importante para ter great results, para ter excelentes resultados com o Cloud Code. Dê ao Cloud Code uma maneira de verificar o próprio trabalho dele. Ou seja, o Cloud Code vai trabalhar com feedback loops, então você vai falar ao Cloud Code, verifica o código que você fez, verifica de novo. E ele fala que isso melhora de duas a três vezes a qualidade do resultado final do Cloud Code só pelo fato de você dar a ele uma ferramenta para ele validar o próprio resultado dele. Ele fala que o Cloud vai testar qualquer coisa que ele colocar na plataforma do Cloud AI barra Code usando o Cloud como extension. Ele vai abrir o browser, ele vai testar a UI, ele vai interagir até que o código funcione e que a experiência de usuário feels good, então pareça boa. A verificação vai olhar por diferentes domínios e pode ser tão simples quanto rodar um comando bash ou rodando um test suite, ou até mesmo testando o app no browser ou no simulador de telefone. Garanta que você vai investir tempo, vamos dizer assim, em fazer isso rock solid, duro como uma rocha. Ou seja, faça isso ser solid, foque em trazer solidez para essas instruções de feedback loops. E agora que você ouviu as dicas do Boris, eu quero saber qual a dica que você tem pra ele e pra galera aqui do YouTube, nos comentários. Qual a arte manha, qual a engenharia que você tá usando com o Cloud Gold que tá sendo fenomenal pra você. No mais, tamo junto. Deixei um vídeo aqui no final pra você conferir que eu acho que vale muito a pena você assistir também. Valeu, grande abraço.