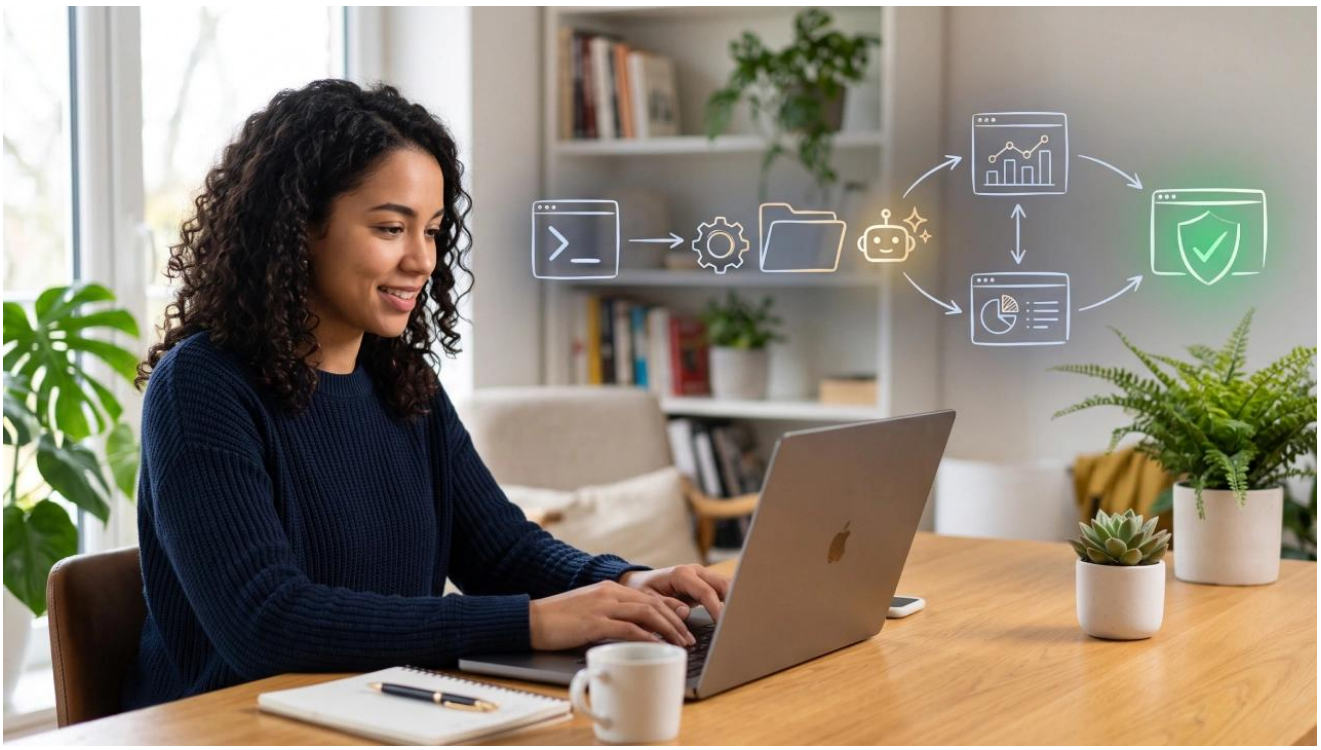

آموزش Claude Code؛ راهنمای کامل ساخت ابزار و اتوماسیون

تصور کنید کاری در شغل یا کسب‌وکارتان دارید که هر هفته چند ساعت زمان می‌گیرد: انتقال دستی داده‌ها به Google Sheets، بررسی رقبا، پاسخ‌گویی به پیام‌ها، تهیه گزارش یا دنبال کردن درخواست‌های مشتریان. حالا تصور کنید به‌جای یاد گرفتن چندین ابزار پیچیده یا استخدام برنامه‌نویس، فقط مسئله را برای هوش مصنوعی توضیح دهید، از آن بخواهید راه‌حل بسازد و در همان مسیر هم یاد بگیرید دقیقا چه اتفاقی در حال رخ دادن است.

در این مقاله، تصمیم گرفتیم یکی از کامل‌ترین نسخه‌های موجود از آموزش Claude Code در یوتیوب را به زبان فارسی ترجمه و به‌شکل راهنمایی قابل استفاده درآوریم.

هدف این آموزش Claude Code این نیست که شما را یک‌شبه به برنامه‌نویس تبدیل کند. هدف این است که بتوانید از هوش مصنوعی برای ساختن ابزارهایی استفاده کنید که واقعا ارزش ایجاد می‌کنند: «زمانتان را آزاد می‌کنند، به مشتریان

کمک می‌کنند یا یک مسئله مهم کاری را حل می‌کنند».

آنچه یاد می‌گیرید

- **Claude Code:** با اجازه شما فایل می‌سازد، کد اجرا می‌کند و ابزارهای واقعی می‌سازد.
- **مصاحبه هوش مصنوعی:** درباره کارتان، بهترین پروژه کوچک و کاربردی را پیدا کنید.
- **پرامپت‌های حساس:** آن‌ها را بررسی کنید و برای شروع از پوشه‌های محدود و Git استفاده کنید.
- **تبدیل داده:** با API و یک سرور محلی، داده واقعی را به داشبورد قابل استفاده تبدیل کنید.

Claude Code چیست و چرا با چت معمولی فرق دارد؟

Claude Code همان مدل هوش مصنوعی Claude است، اما به‌جای اینکه فقط در مرورگر با آن گفت‌وگو کنید، داخل Terminal یا خط فرمان کامپیوترتان اجرا می‌شود. تفاوت مهم این است که Claude Chat می‌تواند راهنمایی، متن و کد تولید کند؛ اما در Claude Code، هوش مصنوعی با اجازه شما می‌تواند فایل بسازد، کد بنویسد، برنامه را اجرا کند، خطاها را بررسی کند و نسخه اصلاح‌شده را دوباره اجرا کند.

به یک مثال ساده فکر کنید:

- **Claude Chat:** مثل معماری است که نقشه ساختمان را می‌فرستد و شما باید خودتان سازنده پیدا کنید.
- **Claude Code:** مثل همان معمار است که ابزار هم دارد، کنار شما می‌ایستد و با راهنمایی شما ساختمان را مرحله‌به‌مرحله می‌سازد.

این یعنی در **آموزش Claude Code** لازم نیست ابتدا بدانید Python چیست، چطور API صدا زده می‌شود یا چگونه یک سایت دیپلوی می‌شود. کافی است مسئله را روشن توضیح دهید، با کنجکاوی سوال بپرسید و هر اقدام مهم را پیش از تایید بررسی کنید.

مدل اصلی: چرخه یادگیری با هوش مصنوعی

یکی از بهترین ایده‌های این راهنما، «چرخه هوش مصنوعی» است. به‌جای اینکه بی‌هدف ده‌ها آموزش ببینید، یک مشکل واقعی انتخاب می‌کنید و از هوش مصنوعی می‌خواهید برای کشف راه‌حل با شما مصاحبه کند.

چرخه به این شکل کار می‌کند:

۱. انتخاب مشکل: یک کار تکراری، خسته‌کننده یا پرهزینه را در زندگی یا کسب‌وکار پیدا کنید.
۲. مصاحبه با Claude: از Claude بخواهید درباره جریان کاری فعلی، منابع اطلاعاتی،

خروجی موردنیاز و تصمیم‌هایی که می‌خواهید بگیرید سوال بپرسد.
۳. ساخت نسخه اولیه: یک نسخه کوچک از راه‌حل را بسازید.
۴. یادگیری مفاهیم: هر اصطلاح یا خطای نامفهوم را از خود Claude بخواهید توضیح دهد.
۵. توسعه تدریجی: پس از رسیدن به یک نتیجه کاربردی، قابلیت بعدی را اضافه کنید.

نتیجه این است یادگیری، از یک موضوع انتزاعی به یک پروژه واقعی تبدیل می‌شود. شما فقط درباره API یا سرور محلی مطالعه نمی‌کنید؛ بلکه وقتی واقعا به آن‌ها نیاز پیدا می‌کنید، مفهومشان را هم بهتر می‌فهمید.

اولین پرامپت برای پیدا کردن ایده پروژه

اگر نمی‌دانید چه چیزی بسازید، این نقطه شروع مناسبی برای **آ**موزش Claude Code است:

["message_box text_color="light] می‌خواهم استفاده عملی از هوش مصنوعی را یاد بگیرم؛ اما نمی‌خواهم صرفا آموزش ببینم. از من درباره کار، کسب‌وکار و کارهای تکراری‌ام سوال بپرس تا یک پروژه کوچک، اما مفید پیدا کنیم؛ پروژه‌ای که زمانم را ذخیره کند، برای مشتری ارزش بسازد یا به درآمدزایی کمک کند. در مسیر ساخت هم مفاهیم لازم را به من آموزش بده. [/message_box]

سپس آزادانه توضیح دهید روزتان چگونه می‌گذرد، چه داده‌هایی جمع می‌کنید، چه گزارش‌هایی می‌سازید و کدام کارها دایما به تعویق می‌افتند. گفتار به متن هم این کار را سریع‌تر می‌کند؛ به‌عنوان مثال می‌توانید از یک ابزار دیکته مانند Wispr Flow استفاده کنید تا به‌جای تایپ کردن، مسئله را با زبان طبیعی توضیح دهید.

پیش‌نیازهای شروع آموزش Claude Code

برای شروع لازم نیست محیط توسعه پیچیده‌ای آماده کنید. دو مورد اصلی کافی

است:

- **اپلیکیشن دسکتاپ Claude:** بهتر است به‌جای نسخه وب، اپ دسکتاپ Claude را از صفحه دانلود Claude دریافت کنید. در این محیط به Chat، Cowork و Code دسترسی دارید.
- **ابزار گفتار به متن:** اختیاری است، اما کمک می‌کند درخواست‌های دقیق‌تر و طولانی‌تر را سریع‌تر بیان کنید.

مهم‌تر از ابزارها، ذهنیت است: قرار نیست دستورهای ترمینال را کورکورانه کپی کنید. در این آموزش Claude Code، هر جا عبارت، فرمان یا پنجره‌ای نامفهوم دیدید، همان را به Claude بدهید و بپرسید: «این چیست؟ چرا به آن نیاز داریم؟ اگر تایید کنم چه اتفاقی می‌افتد؟»

Terminal چیست؟ ترس از صفحه سیاه را کنار بگذارید

Terminal یک کامپیوتر متفاوت یا محیط مخصوص هکرها نیست. همان فایل‌ها و همان کامپیوتر شماست؛ فقط به‌جای کلیک روی آیکن‌ها، با دستورهای متنی آن را کنترل می‌کنید.

- **رابط گرافیکی یا GUI:** مثل ماشین اتوماتیک است؛ با دکمه، پنجره و ماوس کار می‌کنید.
- **Terminal:** مثل ماشین دنده‌ای است؛ همان مقصد را دارید، اما کنترل مستقیم‌تری در اختیار شماست.

دو دستور ساده‌ای که ابتدای کار زیاد می‌بینید این‌ها هستند:

- **دستور mkdir:** ساختن یک پوشه جدید.
- **دستور cd:** ورود به آن پوشه؛ مخفف change directory.
- **دستور claude:** اجرای Claude Code در همان پوشه پروژه.

بهترین کار این است که برای هر پروژه یک پوشه جداگانه بسازید. این کار فایل‌ها را مرتب نگه می‌دارد و محدوده دسترسی Claude Code را هم مشخص‌تر می‌کند.

نصب و راه‌اندازی Claude Code

می‌توانید Claude Code را از اپ دسکتاپ نصب کنید یا از دستور نصب رسمی استفاده کنید. اگر از Terminal استفاده می‌کنید، ابتدا مطمین شوید فرمان را فقط از منبع رسمی Anthropic دریافت کرده‌اید. یک اصل مهم در **آموزش Claude Code** این است: هیچ دستور ناشناسی را بدون فهم کلی و بدون اعتماد به منبع آن اجرا نکنید.

پس از نصب، Terminal را باز کنید، وارد پوشه پروژه شوید و دستور claude را وارد کنید. در اولین اجرا معمولا لازم است وارد حساب Claude شوید و مشخص کنید که پوشه فعلی را می‌شناسید و به آن اعتماد دارید.

از این مرحله به بعد، Claude Code می‌تواند در همان پوشه فایل‌ها را بخواند یا برای تغییر فایل‌ها و اجرای دستورها از شما اجازه بگیرد.

امنیت در آموزش Claude Code: چه چیزهایی را تایید کنیم؟

Claude Code ابزار قدرتمندی است و همین قدرت نیاز به دقت دارد. برای خواندن فایل‌ها معمولا مجوز جداگانه‌ای لازم نیست، اما برای ساختن، ویرایش کردن، حذف فایل‌ها، نصب پکیج‌ها یا اجرای فرمان‌های حساس، باید تایید بدهید.

چند اصل عملی برای شروع امن:

- **شروع محدود:** با یک پوشه آزمایشی یا پروژه کوچک شروع کنید؛ نه فایل‌های اصلی کسب‌وکارتان.
- **بررسی مجوزها:** هر درخواست مجوز را بخوانید؛ مخصوصا فرمان‌های Bash.

- **استفاده از Git:** برای پروژه‌های مهم از Git استفاده کنید تا بتوانید به نسخه قبلی فایل‌ها برگردید.
- **حفظ اطلاعات حساس:** کلیدهای API، رمزها و اطلاعات مشتری را در مخزن عمومی GitHub قرار ندهید.
- **داده‌های مشتریان:** در پروژه‌هایی که داده مشتری دارند، سطح بررسی و محدودیت امنیتی را بالاتر ببرید.

دستور rm را جدی بگیرید

یکی از دستورهایی که باید همیشه با دقت بررسی شود، rm است که فایل‌ها را حذف می‌کند. برخلاف انتقال فایل به سطل زباله، حذف با rm معمولا مسیر بازیابی ساده‌ای ندارد.

rm test-file.txt rm -r temporary-folder

فرمان اول یک فایل مشخص را حذف می‌کند و فرمان دوم یک پوشه و تمام زیرپوشه‌هایش را. هر وقت چنین دستوری دیدید، ابتدا نام و مسیر دقیق مورد حذف را بخوانید. اگر ابهام دارید، فرمان را در یک پنجره گفت‌وگوی دیگر به Claude یا ابزار دیگری بدهید تا مستقل توضیحش دهد.

پروژه نمونه: ساخت داشبورد تحلیل رقبا در یوتیوب

برای فهم عملی آموزش Claude Code، یک پروژه واقعی را مرحله‌به‌مرحله بررسی کنیم: داشبوردی که ویدیوهای جدید چند کانال یوتیوب را می‌گیرد و عنوان، تصویر بندانگشتی، تعداد بازدید و تاریخ انتشار را در یک صفحه نمایش می‌دهد.

این پروژه کوچک است، اما چند مهارت مهم را همزمان پوشش می‌دهد: کار با API، ذخیره داده، ساخت فایل، اجرای کد، راه‌اندازی سرور محلی و توسعه تدریجی یک رابط وب.

گام اول: پروژه را کوچک تعریف کنید

اشتباه رایج این است که از همان ابتدا بخواهید یک داشبورد حرفه‌ای برای ۵۰ کانال، اعلان خودکار، تحلیل موضوعی و سیستم عضویت بسازید. بهتر است با سه کانال شروع کنید و فقط داده ۱۰ ویدیوی آخر را دریافت کنید.

[message_box text_color="light"] می‌خواهم یک ردیاب رقبا در یوتیوب بسازم. فعلا سه نام کانال دارم. با YouTube Data API ده ویدیوی اخیر هر کانال را دریافت کن؛ عنوان، لینک thumbnail، تعداد بازدید و تاریخ انتشار را ذخیره کن. داده‌ها را فعلا در یک فایل JSON محلی نگه دار. سپس یک صفحه HTML ساده بساز که این اطلاعات را به شکل کارت نمایش دهد. [/message_box]

Claude Code می‌تواند فایل اسکریپت، صفحه HTML و فایل داده را بسازد. سپس با تایید شما اسکریپت را اجرا می‌کند، خطاها را می‌خواند و نتیجه را بررسی می‌کند.

گام دوم: کلید YouTube Data API را بسازید

برای دریافت داده عمومی کانال‌ها به API Key نیاز دارید. مسیر کلی در Google Cloud Console این است:

- **ساخت Project:** یک Project جدید بسازید؛ مثلا YouTube Tracker.
- **بخش APIs & Services:** وارد Library شوید.
- **فعال‌سازی API:** سرویس YouTube Data API v۳ را جست‌وجو و فعال کنید.
- **دریافت کلید:** به بخش Credentials بروید و یک API Key بسازید.
- **محدودسازی:** کلید را به YouTube Data API v۳ محدود کنید.
- **امنیت:** کلید را مانند رمز عبور نگه دارید و هرگز در ویدیوی عمومی، فایل عمومی یا GitHub عمومی منتشر نکنید.

محیط Google Cloud برای زیرساخت‌های سازمانی طراحی شده و طبیعی است که گزینه‌های زیادی مثل صورت‌حساب، Terraform، حساب سرویس و سیاست‌های امنیتی ببینید. برای این پروژه، هدف فقط دریافت یک کلید API است؛ اگر صفحه

گیج‌کننده شد، محتوای آن را کپی کنید یا اسکرین‌شات بگیرید و از Claude بخواهید دقیقا بگوید کدام بخش را باید انتخاب کنید.

ساخت، اجرا و بررسی فایل‌ها

Claude Code برای این پروژه معمولا یک فایل Python برای دریافت داده و یک فایل HTML برای نمایش آن می‌سازد. حتی اگر کدنویس نیستید، لازم نیست تک‌تک خطوط را حفظ کنید. اما خوب است مفهوم کلی را بررسی کنید:

- **اسکریپت Python:** به YouTube Data API درخواست می‌فرستد.
- **یافتن کانال‌ها:** نام کانال‌ها را پیدا می‌کند و ویدیوهای جدیدشان را دریافت می‌کند.
- **ذخیره داده‌ها:** داده‌ها را در فایل videos.json ذخیره می‌کند.
- **نمایش اطلاعات:** فایل HTML داده‌های JSON را می‌خواند و کارت‌های ویدیویی را نمایش می‌دهد.

اگر Claude برای اجرای فایل درخواست اجازه کرد، ابتدا دستور را بخوانید. مثلا اجرای یک فایل Python ساخته‌شده در همان پروژه، معمولا فرمانی شبیه این است:

```
python۳ fetch_videos.py
```

این فرمان فقط می‌گوید Python نسخه ۳، فایل fetch_videos.py را اجرا کند. اگر پروژه شما از API استفاده می‌کند، خروجی باید شامل دریافت داده از کانال‌ها و ساخت یا به‌روزرسانی فایل JSON باشد.

چرا صفحه HTML مستقیم باز نمی‌شود؟

ممکن است با دوبار کلیک روی فایل index.html، صفحه نتواند فایل JSON را بخواند. دلیلش محدودیت امنیتی مرورگرها در خواندن فایل‌های محلی با JavaScript است. راه‌حل ساده، راه‌اندازی یک سرور محلی است:

python۳ -m http.server ۸۰۰۰

این دستور یک وب‌سرور بسیار کوچک روی کامپیوتر خودتان اجرا می‌کند. سپس با رفتن به localhost:۸۰۰۰ در مرورگر، داشبورد را می‌بینید. localhost یعنی کامپیوتر خودتان؛ این صفحه هنوز روی اینترنت عمومی قرار نگرفته و فقط تا وقتی فرمان فعال است اجرا می‌شود.

بهبود مرحله‌ای، نه بازنویسی کامل

پس از رسیدن به نسخه اولیه، گفت‌وگو را ادامه دهید. مثلا ممکن است متوجه شوید ویدیوهای Shorts در داشبورد ظاهر می‌شوند، در حالی که فقط ویدیوهای بلند می‌خواهید. کافی است درخواست کنید Shorts بر اساس مدت زمان ویدیو فیلتر شوند.

همین الگو برای قابلیت‌های بعدی هم کاربرد دارد:

- **مرتب‌سازی:** بر اساس تعداد بازدید
- **نمایش سن ویدیو:** افزودن زمان انتشار
- **برجسته‌سازی:** برجسته کردن ویدیوهای پربازدید
- **توسعه:** افزودن تعداد کانال بیشتر
- **برچسب‌گذاری:** برچسب‌گذاری موضوعی با هوش مصنوعی
- **به‌روزرسانی:** به‌روزرسانی زمان‌بندی‌شده داده‌ها
- **گزارش‌گیری:** ارسال گزارش روزانه یا هفتگی به Slack

اصل مهم در آموزش Claude Code این است: «شما مسیر را تعیین می‌کنید و Claude Code کار مکانیکی ساخت و اصلاح را انجام می‌دهد».

چطور داشبورد محلی را برای تیم منتشر کنیم؟

بعد از اینکه نسخه محلی خوب کار کرد، ممکن است بخواهید هم‌تیمی‌ها هم به آن دسترسی داشته باشند. در این مرحله Claude Code می‌تواند مفاهیم استقرار یا Deploy را توضیح دهد و گزینه‌هایی مثل GitHub Pages، Vercel یا Netlify را پیشنهاد کند.

برای یک پروژه استاتیک ساده، Vercel یا GitHub Pages می‌توانند میزبان رایگان مناسبی باشند. اما پیش از عمومی کردن صفحه، به این سوال‌ها پاسخ دهید:

- **محرمانگی:** آیا داده‌ها محرمانه هستند؟
- **امنیت کد:** آیا کلید API در کد یا فایل‌های قابل‌دسترسی وجود دارد؟
- **دسترسی تیمی:** آیا فقط اعضای تیم باید وارد شوند؟
- **احراز هویت:** آیا نیاز به نام کاربری، رمز عبور یا Google OAuth دارید؟

اگر داده‌ها خصوصی‌اند، باید احراز هویت را اضافه کنید. اگر هم قرار است محصول را به سرویس پولی تبدیل کنید، می‌توان در مرحله بعد سراغ سیستم پرداختی مانند Stripe رفت. اما ابتدا مطمین شوید ابزار کوچک شما واقعا یک مسئله را حل می‌کند؛ نه اینکه صرفا از نظر فنی جذاب باشد.

نمونه‌هایی از ابزارهای واقعی که می‌توان ساخت

قدرت آموزش Claude Code زمانی بهتر مشخص می‌شود که از یک پروژه ساده مانند ردیاب ویدیوهای یوتیوب فراتر برویم. با Claude Code می‌توان ابزارهای سفارشی ساخت که به سرویس‌ها، فایل‌ها و داده‌های مختلف متصل شوند و بخشی از کارهای تکراری را خودکار کنند.

نمونه‌های زیر نشان می‌دهند چنین ابزارهایی چگونه می‌توانند در عملیات کسب‌وکار، تولید محتوا، تحلیل اطلاعات و مدیریت کارهای روزمره کاربرد داشته باشند:

سیستم مدیریت تیکت‌های پشتیبانی

پیام‌ها و درخواست‌های جدید را از ابزارهای ارتباطی دریافت کند، آن‌ها را دسته‌بندی و اولویت‌بندی کند، برای موارد بدون پاسخ هشدار بفرستد و وضعیت رسیدگی را در یک داشبورد ساده نمایش دهد.

بات‌های تخصصی برای Slack

دستیارهایی بسازید که مستقیما در فضای کاری تیم فعالیت کنند؛ مثلا به اعضای تیم در نوشتن پیام‌های فروش، پاسخ به پرسش‌های تکراری، خلاصه‌سازی گفتگوها یا تولید پیش‌نویس محتوای لینکدین کمک کنند.

سیستم تحلیل و رصد رقبا

محتوای جدید رقبا را از منابع مشخص جمع‌آوری کند، تغییرات مهم مانند موضوعات پرتکرار، ویدیوهای پربازدید یا فعالیت‌های غیرعادی را شناسایی کند و یک گزارش خلاصه برای بررسی تیم آماده کند.

مرکز تحلیل محتوای تیم

محتوای منتشرشده در سال‌های گذشته در LinkedIn، Instagram، YouTube و دیگر کانال‌ها را در یک مجموعه قابل جست‌وجو جمع کند تا بتوان عملکرد موضوعات، فرمت‌ها و ایده‌های مختلف را بررسی و از آن برای تولید محتوای جدید استفاده کرد.

دستیار شخصی برای کارهای تکراری

ابزاری اختصاصی برای انجام وظایف مشخص روزانه، مانند جست‌وجو و خلاصه‌سازی اطلاعات، سازمان‌دهی فایل‌ها، آماده‌سازی گزارش یا اجرای یک گردش کار ثابت؛ در نتیجه لازم نیست برای هر مرحله به‌صورت دستی یک درخواست جدید ایجاد کنید.

دستیار تمرین و سلامت شخصی

داده‌هایی مانند سابقه تمرین، اهداف ورزشی و اطلاعاتی که کاربر وارد می‌کند را سازمان‌دهی و تحلیل کند تا پیشنهادهای شخصی‌سازی‌شده‌تری برای برنامه تمرینی یا پیگیری عادت‌ها ارائه دهد. البته چنین ابزاری نباید جایگزین پزشک یا متخصص

سلامت باشد.

این مثال‌ها قرار نیست یک الگوی ثابت برای همه باشند. نکته اصلی این است تقریبا در هر شغل یا کسب‌وکار، مجموعه‌ای از کارهای دستی، تکراری و زمان‌بر وجود دارد که می‌توان با یک ابزار سفارشی یا گردش کار خودکار، آن‌ها را ساده‌تر و سریع‌تر انجام داد.

["message_box text_color="light]

در واقع، بهترین نقطه برای شروع معمولا پیدا کردن یک فرایند پیچیده نیست؛ بلکه کافی است از خودتان بپرسید: «کدام کاری را بارها و بارها انجام می‌دهم که می‌توان یک ابزار برای ساده‌تر کردن آن ساخت؟»

[message_box/]

API و MCP Server را در حد کاربردی بفهمید

وقتی با Claude Code کار می‌کنید، احتمالا با عبارت‌هایی مثل API، HTTP، JSON، Git، SSH یا MCP Server روبه‌رو می‌شوید. لازم نیست همه را یک‌جا یاد بگیرید. کافی است هنگام نیاز، مفهوم را به‌اندازه کاربردش بفهمید.

API چیست؟

API مخفف Application Programming Interface است. ساده‌تر بگوییم: «یک درگاه رسمی است که اجازه می‌دهد نرم‌افزارها با هم صحبت کنند». مثلا YouTube Data API راه رسمی دریافت داده کانال‌ها و ویدیوهای عمومی یوتیوب است.

بدون API، ممکن است مجبور شوید اطلاعات را دستی کپی کنید یا اسکریپتی بسازید که مانند انسان روی دکمه‌ها کلیک کند. چنین روش‌هایی شکننده‌اند؛ یک تغییر کوچک در ظاهر سایت می‌تواند کل فرایند را خراب کند.

MCP Server چیست؟

MCP راهی برای متصل کردن هوش مصنوعی به ابزارها و منابع اطلاعاتی است. ایده اصلی این است که به‌جای وارد کردن دستی زمینه در هر گفت‌وگو، یک منبع ساختاریافته بسازید تا ابزارهای هوش مصنوعی بتوانند با مجوز مناسب به داده‌های به‌روز شما دسترسی داشته باشند.

مثلا یک MCP Server شخصی می‌تواند اطلاعات پروژه‌ها، هدف‌ها، کارهای باز و یادداشت‌ها را در اختیار Claude یا ChatGPT قرار دهد. این قابلیت بسیار قدرتمند است؛ اما باید با توجه جدی به حریم خصوصی، دسترسی‌ها و خطراتی مانند prompt injection پیاده‌سازی شود.

با اصطکاک فنی چگونه برخورد کنیم؟

بخش مهمی از آموزش Claude Code به مهارتی مربوط است که کمتر درباره آن صحبت می‌شود: تحمل اصطکاک. گرفتن API Key، پیدا نکردن یک دکمه در Google Cloud، خطای نصب یک پکیج یا تغییر رابط یک سرویس، همگی طبیعی هستند.

برنامه‌نویسان سال‌ها این کار را با جست‌وجو در گوگل و Stack Overflow انجام می‌دادند. امروز می‌توانید متن خطا، محتوای صفحه یا تصویر رابط را به Claude بدهید و بپرسید:

- «این خطا دقیقا چه می‌گوید؟»
- «گام بعدی چیست؟»
- «کدام گزینه را باید انتخاب کنم و چرا؟»
- «این دستور چه تغییری روی سیستم من می‌دهد؟»
- «آیا این راه‌حل با نسخه فعلی ابزار سازگار است؟»

هدف این نیست که هیچ‌وقت با مشکل روبه‌رو نشوید. هدف این است که هر

اصطکاک به یک سوال کوچک تبدیل شود، نه دلیلی برای رها کردن پروژه.

چند Claude Code هم‌زمان؛ چه زمانی مفید است؟

وقتی یک پروژه بزرگ‌تر می‌شود، می‌توانید چند پنجره Terminal باز کنید و در هرکدام یک Claude Code جداگانه اجرا کنید. مثلا یک پنجره روی داشبورد تحلیل رقبا کار کند و پنجره دیگر مستندات یا پروژه‌ای کاملا جدا را پیش ببرد.

اما در ابتدای مسیر، فقط با یک پنجره کار کنید. اجرای چند عامل روی یک فایل یا یک قابلیت مشترک می‌تواند باعث تداخل تغییرات شود. اول یاد بگیرید پروژه را کوچک نگه دارید، تغییرات را بررسی کنید و از Git برای نسخه‌بندی استفاده کنید؛ سپس به کار موازی برسید.

چه چیزی را نباید بسازید؟

در دنیای عامل‌های هوش مصنوعی، به‌راحتی می‌توان ساعت‌ها صرف ساختن سیستم‌هایی کرد که عامل‌های دیگر را مدیریت کنند، اما خروجی ملموسی ندارند. معیار ساده برای هر پروژه این است:

- **ارزش برای مشتری:** آیا برای مشتری ارزش ایجاد می‌کند؟
- **ذخیره زمان:** آیا زمان واقعی من یا تیمم را ذخیره می‌کند؟
- **رشد کسب‌وکار:** آیا به افزایش درآمد یا کیفیت تصمیم‌ها کمک می‌کند؟

اگر پاسخ هیچ‌کدام مثبت نیست، احتمالا پروژه هنوز اولویت ندارد. بهترین پروژه برای آموزش Claude Code معمولا پروژه‌ای کوچک، واقعی و کمی آزاردهنده است؛ همان کاری که همیشه می‌گویید «کاش لازم نبود خودم انجامش بدهم».

اگر ابزار ساخته‌شده را می‌خواهید برای تیم یا مشتریان آنلاین کنید، Hostinger Horizons مسیر ساخت و انتشار اپ را ساده‌تر می‌کند.

نقشه شروع یک آخر هفته‌ای

اگر می‌خواهید این **آموزش Claude Code** را به عمل تبدیل کنید، لازم نیست از روز اول یک SaaS بسازید. این برنامه کوتاه نقطه شروع خوبی است:

۱. **یک درد واقعی فهرست کنید:** سه کار تکراری یا وقت‌گیر هفته گذشته را بنویسید.
۲. **از Claude بخواهید مصاحبه کند:** مسئله‌ای را انتخاب کنید که هم مفید و هم در ابعاد کوچک قابل ساخت باشد.
۳. **پوشه پروژه بسازید:** یک فضای جدا و محدود برای آزمایش ایجاد کنید.
۴. **Claude Code را اجرا کنید:** از آن بخواهید ابتدا یک طرح اجرا و ساختار فایل‌ها پیشنهاد دهد.
۵. **نسخه اولیه را بسازید:** فقط یک ورودی، یک فرایند و یک خروجی مشخص داشته باشید.
۶. **اجازه‌ها را بخوانید:** هر فرمان حساس را بررسی کنید و اگر لازم شد درباره آن سوال بپرسید.
۷. **نتیجه را آزمایش کنید:** داده واقعی اما کم‌خطر وارد کنید و خروجی را بررسی کنید.
۸. **فقط یک قابلیت اضافه کنید:** مثلا فیلتر، مرتب‌سازی، اعلان و...

نقطه قوت Claude Code این نیست شما را از فکر کردن بی‌نیاز کند. نقطه قوتش این است فاصله بین «ایده دارم» و «یک ابزار عملی دارم» را به‌شدت کم می‌کند. هرچه بهتر سوال بپرسید، کار را به نسخه‌های کوچک تقسیم کنید و با دقت امنیت را مدیریت کنید، ابزارهای مفیدتری می‌سازید.

در نهایت، بهترین آموزش Claude Code همان پروژه‌ای است که فردا واقعا از آن

استفاده می‌کنید. با یک مسئله کوچک شروع کنید، از اصطکاک نترسید و اجازه دهید هر پروژه، ایده پروژه بعدی را به شما نشان دهد.

یک قدم جلوتر بروید

آنچه در این آموزش دیدیم، فقط درباره ساختن یک داشبورد یا ابزار کوچک نبود؛ درباره یک تغییر ذهنیت بود: «به‌جای نگاه کردن به هوش مصنوعی به‌عنوان یک چت‌بات ساده، از آن به‌عنوان همکاری استفاده کنیم که می‌تواند مسائل واقعی کاری را حل کند».

این نگاه، وقتی از سطح ابزارهای شخصی به سطح کل کسب‌وکار برود، می‌تواند اثر بسیار بزرگ‌تری داشته باشد؛ از خودکارسازی فرایندهای بازاریابی و فروش گرفته تا تحلیل داده و بهبود تجربه مشتری. اگر می‌خواهید بدانید چطور می‌توان همین طرز فکر را از یک پروژه شخصی به یک استراتژی سازمانی گسترش داد، مقاله «چگونه از هوش مصنوعی در کسب‌وکار استفاده کنیم؟» را در وبسایت مدیر سبز مطالعه کنید.