AI-संचालित डेवलपमेंट

आइडिया से प्रोडक्शन तक

शुरू से अपनी वेबसाइट, ऐप या स्टार्टअप बनाने के लिए संपूर्ण ऑल-इन-वन गाइड। AI एजेंट्स और टूल्स में महारत हासिल करें, पेमेंट इंटीग्रेट करें, अपना सर्वर डिप्लॉय करें, और ऑर्गेनिक ग्रोथ के लिए मार्केटिंग ट्रिक्स सीखें।

apifreellm.com

संस्करण 1.0 — फरवरी 2026

विषय सूची

1. परिचय

> *आप अभी यह इसलिए पढ़ रहे हैं क्योंकि कहीं न कहीं, किसी न किसी तरह, मार्केटिंग रणनीतियों, SEO तकनीकों और स्मार्ट पोजिशनिंग के संयोजन ने इस कोर्स को आपकी स्क्रीन तक पहुंचाया। बस इतना ही आपको बता देता है: इस गाइड में बताई गई रणनीतियाँ वाकई काम करती हैं। और हाँ, आप इनमें से हर एक सीखेंगे।*

दुनिया बदल चुकी है

सॉफ्टवेयर डेवलपमेंट अब वो नहीं रहा जो पहले हुआ करता था। कुछ साल पहले, एक वेब एप्लिकेशन बनाने के लिए महीनों का काम, डेवलपर्स की एक टीम और एक बड़ा बजट चाहिए होता था। आज, सही टूल्स और सही ज्ञान के साथ एक अकेला व्यक्ति कुछ दिनों में एक प्रोडक्शन-रेडी एप्लिकेशन बना और लॉन्च कर सकता है, महीनों में नहीं।

यह कोर्स उन प्रोफेशनल्स द्वारा लिखा गया है जो इंडस्ट्री में काम करते हैं और असली प्रोडक्ट्स बनाने के लिए रोज़ाना AI एजेंटिक टूल्स का उपयोग करते हैं। हम किसी किताब से थ्योरी नहीं सिखाते। हम वो सिखाते हैं जो वास्तविक दुनिया में, अभी, काम करता है।

यह कोर्स क्यों मौजूद है

AI और LLMs अब इंसानों से बेहतर और तेज़ executor हैं। वे कोड लिख सकते हैं, उसे डीबग कर सकते हैं, रिफैक्टर कर सकते हैं और उसे ऐसी स्पीड से डिप्लॉय कर सकते हैं जो कोई इंसान मैच नहीं कर सकता। लेकिन एक चीज़ है जो उनमें मौलिक रूप से नहीं है: **आइडियाज़**।

> AI मॉडल असाधारण executor हैं, लेकिन वे innovator नहीं हैं। वे मार्केट में कोई गैप नहीं देखते और सोचते नहीं कि "मैं इसे ठीक करने के लिए कुछ बना सकता हूं।" यह आपका काम है। आपकी भूमिका आइडियाज़ के आर्किटेक्ट बनने की है। AI आपका बिल्डर है।

जब आप किसी LLM को खराब निर्देश देते हैं, तो भी वह कुछ न कुछ बना देगा। वह रुककर यह नहीं बताएगा कि वास्तव में क्या बेहतर होता। आप जो बनाते हैं उसकी गुणवत्ता सीधे आपके ज्ञान की गुणवत्ता के समानुपाती है। यह कोर्स उस अंतर को पाटता है।

यह सिर्फ एक डेवलपमेंट कोर्स नहीं है। यह एक आइडिया से एक लाइव, रेवेन्यू जनरेट करने वाले प्रोडक्ट तक पहुंचने का एक संपूर्ण ब्लूप्रिंट है। इसमें बेहद प्रभावी मार्केटिंग और SEO रणनीतियाँ शामिल हैं — वही तकनीकें जिन्होंने आपको इसी पेज तक पहुंचाया।

2. AI-फर्स्ट डेवलपमेंट स्टैक

> *आप जो टूल्स चुनते हैं वे तय करेंगे कि आप कितनी तेज़ी से आगे बढ़ेंगे। इस अध्याय में, हम आज उपलब्ध AI कोडिंग एजेंट्स का विश्लेषण करते हैं, सही एजेंट चुनने में आपकी मदद करते हैं, और वे प्रॉम्प्टिंग रणनीतियाँ सिखाते हैं जो शौकियों को प्रोफेशनल्स से अलग करती हैं।*

> **नोट**: यह गाइड Windows का उपयोग करके लिखी गई है, लेकिन सभी टूल्स और स्टेप्स macOS और Linux पर बिल्कुल उसी तरह काम करते हैं। Google Antigravity, Claude Code, और इस कोर्स में चर्चा की गई हर अन्य एप्लिकेशन पूरी तरह से क्रॉस-प्लेटफॉर्म है। यदि आप Mac या Linux मशीन पर हैं, तो बस वही स्टेप्स फॉलो करें — इंटरफेस और वर्कफ्लो समान हैं।

AI एजेंट लैंडस्केप

AI कोडिंग टूल्स का क्षेत्र विस्फोटक रूप से बढ़ा है। लेकिन सभी टूल्स एक जैसे नहीं हैं। कुछ बस महिमामंडित ऑटोकम्प्लीट इंजन हैं। अन्य पूर्ण स्वायत्त एजेंट हैं जो आपका पूरा कोडबेस पढ़ सकते हैं, एक रणनीति बना सकते हैं, कई फाइलों में बदलाव कर सकते हैं, टेस्ट चला सकते हैं, और अपने आप एरर ठीक कर सकते हैं। इस अंतर को समझना महत्वपूर्ण है।

AI कोडिंग टूल्स की दो मौलिक श्रेणियाँ हैं:

- **इनलाइन असिस्टेंट** — ये आपके एडिटर के अंदर बैठते हैं और जैसे आप टाइप करते हैं वैसे कोड सुझाते हैं। ओरिजिनल GitHub Copilot के बारे में सोचें। ये रिएक्टिव हैं: ये आपके लिखने का इंतज़ार करते हैं, फिर अनुमान लगाने की कोशिश करते हैं कि आगे क्या आएगा। उपयोगी, लेकिन सीमित।
- **एजेंटिक टूल्स** — ये पूरी तरह से अलग चीज़ हैं। आप उन्हें प्राकृतिक भाषा में एक टास्क देते हैं, और वे स्वायत्त रूप से प्लान बनाते हैं, लिखते हैं, एडिट करते हैं, डीबग करते हैं, और iterate करते हैं। वे सिर्फ कोड की एक लाइन सुझाते नहीं। वे फीचर्स बनाते हैं। असली शक्ति यहीं है।

यहाँ वे टूल्स हैं जो अभी मायने रखते हैं:

Claude Code (Anthropic द्वारा)

Claude Code, हमारे अनुभव में, आज उपलब्ध सबसे शक्तिशाली AI कोडिंग एजेंट है। यह एक टर्मिनल-आधारित एजेंट है जो सीधे आपकी प्रोजेक्ट डायरेक्टरी में काम करता है। आप इसे प्राकृतिक भाषा में निर्देश देते हैं, और यह आपकी फाइलें पढ़ता है, कोड लिखता है, कमांड चलाता है, कमिट बनाता है, और स्वायत्त रूप से एरर ठीक करता है। इसे आपके फाइलसिस्टम और शेल तक पूरी पहुंच है, जो इसे वास्तविक डेवलपमेंट कार्य के लिए अविश्वसनीय रूप से प्रभावी बनाती है। आप इसे npm (`npm install -g @anthropic-ai/claude-code`) के माध्यम से इंस्टॉल कर सकते हैं या VS Code एक्सटेंशन के रूप में उपयोग कर सकते हैं, जिसके बारे में हम जल्दी ही चर्चा करेंगे।

Google Antigravity

Antigravity Google का एक डेवलपमेंट एनवायरनमेंट है जो AI-संचालित चैट और एजेंट क्षमताओं को सीधे आपके वर्कफ्लो में लाता है। इसे एक इंटेलिजेंट वर्कस्पेस के रूप में सोचें जहाँ आप चैट इंटरफेस के माध्यम से AI एजेंट्स के साथ इंटरैक्ट कर सकते हैं, और प्रत्येक एजेंट बातचीत से आप एक इंटीग्रेटेड VS Code एडिटर खोल सकते हैं। यह महत्वपूर्ण है: Antigravity आपको conversational AI लेयर देता है, जबकि VS Code कोड एडिटिंग पावर प्रदान करता है। यह कॉम्बिनेशन सीमलेस है — आप एजेंट के साथ चर्चा करते हैं कि क्या बनाना है, फिर बिना विंडो बदले सीधे कोड में कूद जाते हैं।

Cursor

Cursor, VS Code का एक फोर्क है जिसमें AI गहराई से इंटीग्रेटेड है। इसमें इनलाइन सुझाव और एक एजेंटिक "Composer" मोड दोनों हैं जहाँ आप कई फाइलों में बदलाव का वर्णन कर सकते हैं। यदि आप पहले से VS Code उपयोग करते हैं तो UI परिचित है, और यह कई मॉडल्स (Claude, GPT, आदि) को सपोर्ट करता है। यह एक अच्छा टूल है, खासकर अगर आप पूरी तरह से विज़ुअल वर्कफ्लो पसंद करते हैं। हालांकि, इसकी एजेंटिक क्षमताएं, अच्छी होते हुए भी, Claude Code जितनी स्वायत्त नहीं हैं। आपको अक्सर व्यक्तिगत बदलावों की एक-एक करके समीक्षा और अनुमोदन करना होगा।

> हमारा सुझाया गया सेटअप: **Google Antigravity + Claude Code VS Code एक्सटेंशन के रूप में।** अपने काम की प्लानिंग और चर्चा के लिए Antigravity के एजेंट चैट्स का उपयोग करें, फिर इंटीग्रेटेड VS Code खोलें और Claude Code को भारी execution संभालने दें। यह आपको दोनों दुनिया का सर्वश्रेष्ठ देता है: प्लानिंग के लिए बुद्धिमान बातचीत, और बिल्डिंग के लिए स्वायत्त कोड execution।

अपना एजेंट चुनना और कॉन्फ़िगर करना

एजेंटिक टूल को इंस्टॉल और चलाना आसान हिस्सा है। इससे अधिकतम लाभ उठाने के लिए यह समझना ज़रूरी है कि यह कैसे काम करता है, सही सब्सक्रिप्शन चुनना, और इसे सही तरीके से सेट अप करना।

Claude सब्सक्रिप्शन: Pro से शुरू करें

Claude Code के लिए एक Anthropic अकाउंट चाहिए। हम **Claude Pro सब्सक्रिप्शन** से शुरू करने की सलाह देते हैं, जो एंट्री-लेवल पेड टियर है। यह किफ़ायती है और आपको Claude Code की सभी सुविधाओं तक पहुंच देता है। यह प्रयोग करने, वर्कफ्लो सीखने, और अपना पहला सरल प्रोजेक्ट बनाने के लिए सही शुरुआती बिंदु है।

हालांकि, ध्यान रखें कि Pro प्लान में **सीमित उपयोग** है। यदि आप Claude Code का गहन उपयोग करते हैं, तो आप उपयोग सीमा तक अपेक्षाकृत जल्दी पहुंच जाएंगे। सीखने और छोटे प्रोजेक्ट्स के लिए, यह पर्याप्त से अधिक है। लेकिन अगर आप पहले से जानते हैं कि आप Claude Code को अपने प्राइमरी डेवलपमेंट टूल के रूप में उपयोग करना चाहते हैं और हर दिन घंटों काम करने की योजना है, तो शुरू से ही उच्च-स्तरीय प्लान (जैसे Max) में अपग्रेड करने पर विचार करें। उच्च प्लान काफी अधिक उपयोग प्रदान करते हैं, जिसका मतलब है कम रुकावटें और बड़े प्रोजेक्ट्स बनाते समय एक सुचारू वर्कफ्लो।

मॉडल: Claude Opus 4.6

हम वर्तमान में **Claude Opus 4.6** को अपने प्राइमरी डेवलपमेंट मॉडल के रूप में उपयोग करने की सलाह देते हैं। यह अभी, वेबसाइट और एप्लिकेशन बनाने के लिए सबसे अच्छा मॉडल है। यह जटिल आर्किटेक्चर समझता है, साफ और प्रोडक्शन-रेडी कोड लिखता है, कई फाइलों में बदलाव उल्लेखनीय सटीकता से संभालता है, और पहली बार में ही शायद ही कभी सुधार की ज़रूरत होती है। Claude Code के साथ काम करते समय, Opus 4.6 को अपना डिफॉल्ट मॉडल सेट करें। छोटे मॉडल्स की तुलना में आउटपुट गुणवत्ता में अंतर तुरंत ध्यान देने योग्य है।

Antigravity सेट अप करना

इस बिंदु से आगे, यह गाइड **Google Antigravity** को हमारे प्राइमरी डेवलपमेंट एनवायरनमेंट के रूप में उपयोग करते हुए आगे बढ़ती है। यहाँ बताया गया है कि सब कुछ कैसे तैयार करें।

स्टेप 1: अपना प्रोजेक्ट फोल्डर बनाएं। Antigravity खोलने से पहले, अपने कंप्यूटर पर एक फोल्डर बनाएं जहाँ आपका पहला प्रोजेक्ट रहेगा। उदाहरण के लिए, Windows पर आप `C:\Projects\my-first-app` बना सकते हैं, या Mac/Linux पर `~/Projects/my-first-app`। यह वह फोल्डर है जिसे Antigravity वर्कस्पेस के रूप में खोलेगा। यह अभी खाली हो सकता है — हम इसे कोर्स में बाद में कोड से भरेंगे।

स्टेप 2: Antigravity इंस्टॉल और लॉन्च करें। Google Antigravity डाउनलोड करें, इंस्टॉल करें, और खोलें। प्रॉम्प्ट किए जाने पर अपने Google अकाउंट से साइन इन करें।

इंस्टॉलेशन प्रक्रिया के दौरान, Antigravity आपसे कुछ पॉलिसीज कॉन्फ़िगर करने के लिए कहेगा। एक तेज़, स्वायत्त डेवलपमेंट वर्कफ्लो के लिए, हम **कस्टम कॉन्फ़िगरेशन** चुनने और ये विकल्प सेट करने की सलाह देते हैं:

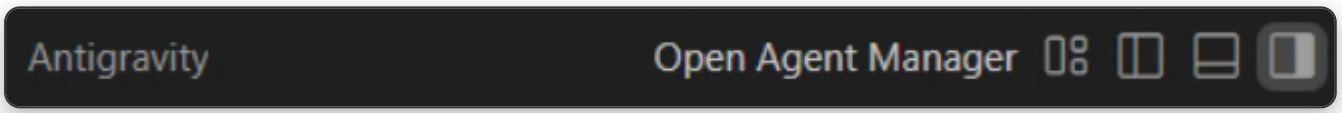
- **Terminal Execution Policy** → Always Proceed
- **Review Policy** → Always Proceed

- **JavaScript Execution** → Always Proceed

इन सेटिंग्स के साथ, Antigravity के एजेंट्स हर कदम पर पुष्टि मांगे बिना स्वायत्त रूप से कमांड execute कर सकेंगे, फाइलें लिख सकेंगे, और कोड चला सकेंगे। यही वह चीज़ है जो इस कोर्स में हम जिस तेज़, प्रवाहमय डेवलपमेंट अनुभव का लक्ष्य रखते हैं उसे सक्षम बनाती है।

> **सुरक्षा चेतावनी**: जब आप एजेंट्स को स्वायत्त रूप से आगे बढ़ने की अनुमति देते हैं, तो Antigravity और Claude Code दोनों बिना मैनुअल अनुमोदन के आपके सिस्टम पर कार्रवाई कर सकते हैं। इसका मतलब है कि आपको सावधान रहना होगा कि आप उन्हें किस चीज़ के संपर्क में लाते हैं। ऐसे कोडबेस पर एजेंट्स न लगाएं जिनमें मैलवेयर हो सकता है, और अविश्वसनीय स्रोतों के लिंक या रिसोर्सेज से सावधान रहें। AI के युग में, नए अटैक वेक्टर हैं — दुर्भावनापूर्ण लोग विशेष रूप से LLMs को हानिकारक कमांड execute करने के लिए बनाई गई सामग्री तैयार कर सकते हैं। हमेशा अपने एजेंट्स क्या कर रहे हैं इस पर नज़र रखें। हम स्पीड के लिए "Always Proceed" सेटअप की सलाह देते हैं, लेकिन सतर्क रहें और नियमित रूप से आउटपुट की समीक्षा करें।

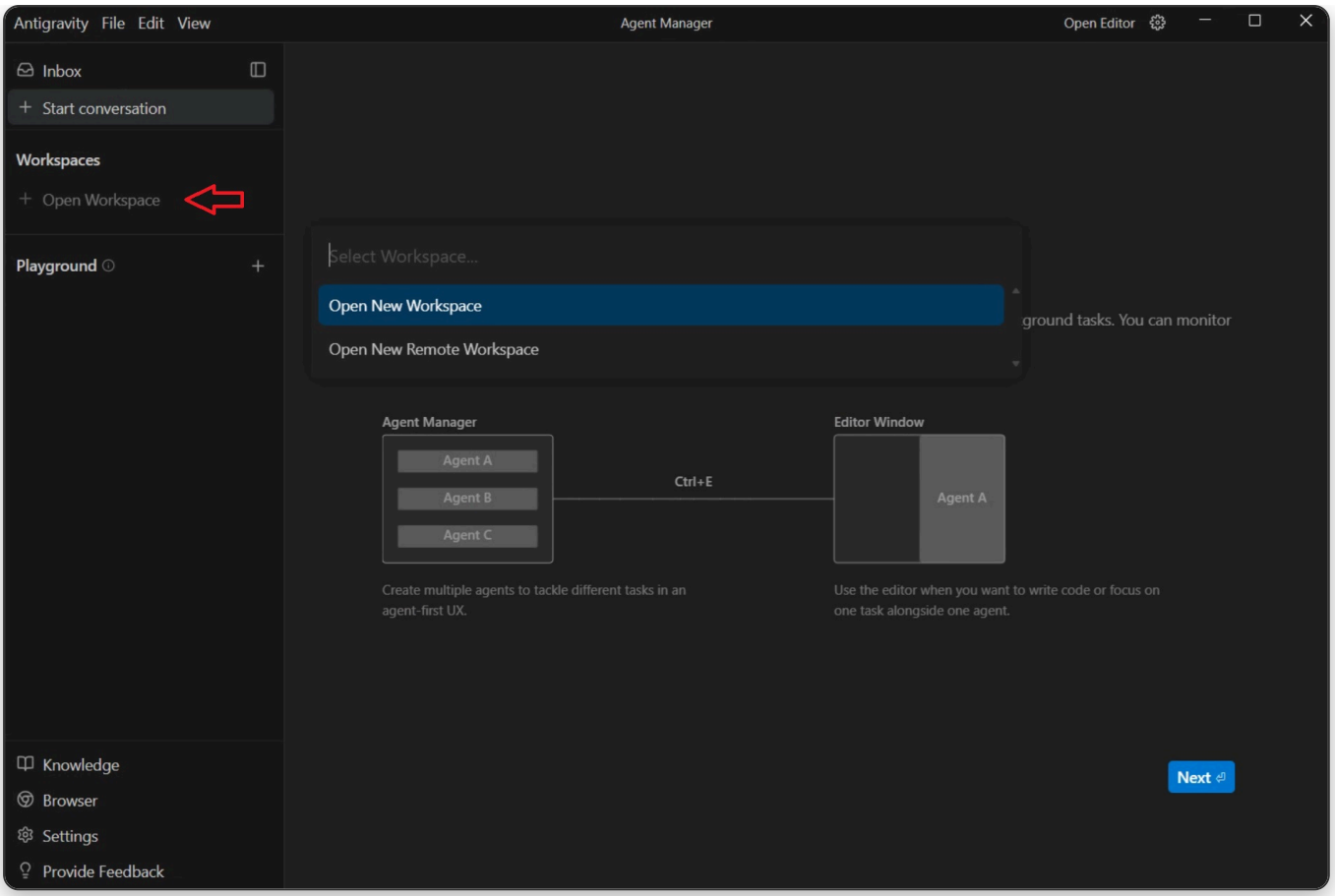
स्टेप 3: Agent Manager खोलें। Antigravity चलने के बाद, विंडो के टॉप बार में "**Open Agent Manager**" पर क्लिक करें।



Antigravity के टॉप बार में "Open Agent Manager" बटन।

Agent Manager Antigravity का केंद्रीय हब है। यहाँ से आप अपने प्रोजेक्ट्स मैनेज करते हैं, AI बातचीत शुरू करते हैं, और अपने डेवलपमेंट वर्कस्पेस खोलते हैं।

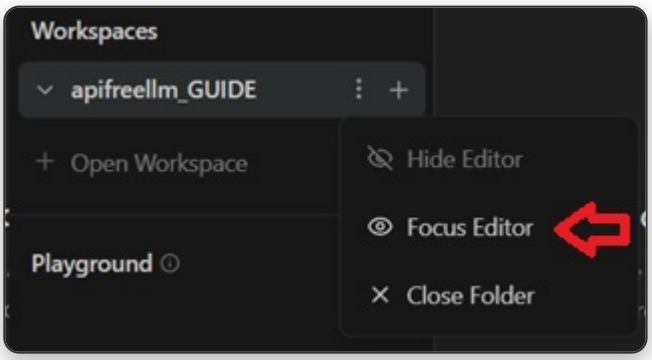
स्टेप 4: अपना पहला वर्कस्पेस बनाएं। Agent Manager में, लेफ्ट साइडबार देखें। "**Workspaces**" सेक्शन के अंतर्गत, "**+ Open Workspace**" पर क्लिक करें। एक ड्रॉपडाउन दिखाई देगा — "**Open New Workspace**" चुनें।



लेफ्ट साइडबार में "+ Open Workspace" पर क्लिक करें, फिर "Open New Workspace" चुनें।

Antigravity आपसे एक फोल्डर चुनने के लिए कहेगा। स्टेप 1 में बनाए गए प्रोजेक्ट फोल्डर तक नेविगेट करें और उसे चुनें। आपका नया वर्कस्पेस "Workspaces" के अंतर्गत लेफ्ट साइडबार में दिखाई देगा, जिसमें आपके द्वारा चुने गए फोल्डर का नाम होगा। प्रत्येक वर्कस्पेस को एक व्यक्तिगत डेवलपमेंट सेशन के रूप में सोचें — प्रति प्रोजेक्ट एक। आप जितने चाहें उतने वर्कस्पेस बना सकते हैं, और किसी भी समय Agent Manager से उनके बीच स्विच कर सकते हैं।

स्टेप 5: एडिटर खोलें। जब आपका वर्कस्पेस बन जाए, तो साइडबार में उसका नाम दिखाई देगा। वर्कस्पेस नाम के बगल में **तीन वर्टिकल डॉट्स** (⋮) पर क्लिक करें, फिर **"Focus Editor"** चुनें। यह उस वर्कस्पेस के लिए पूरा VS Code एनवायरनमेंट खोलता है, जहाँ आप अपना कोड लिखेंगे और एडिट करेंगे।

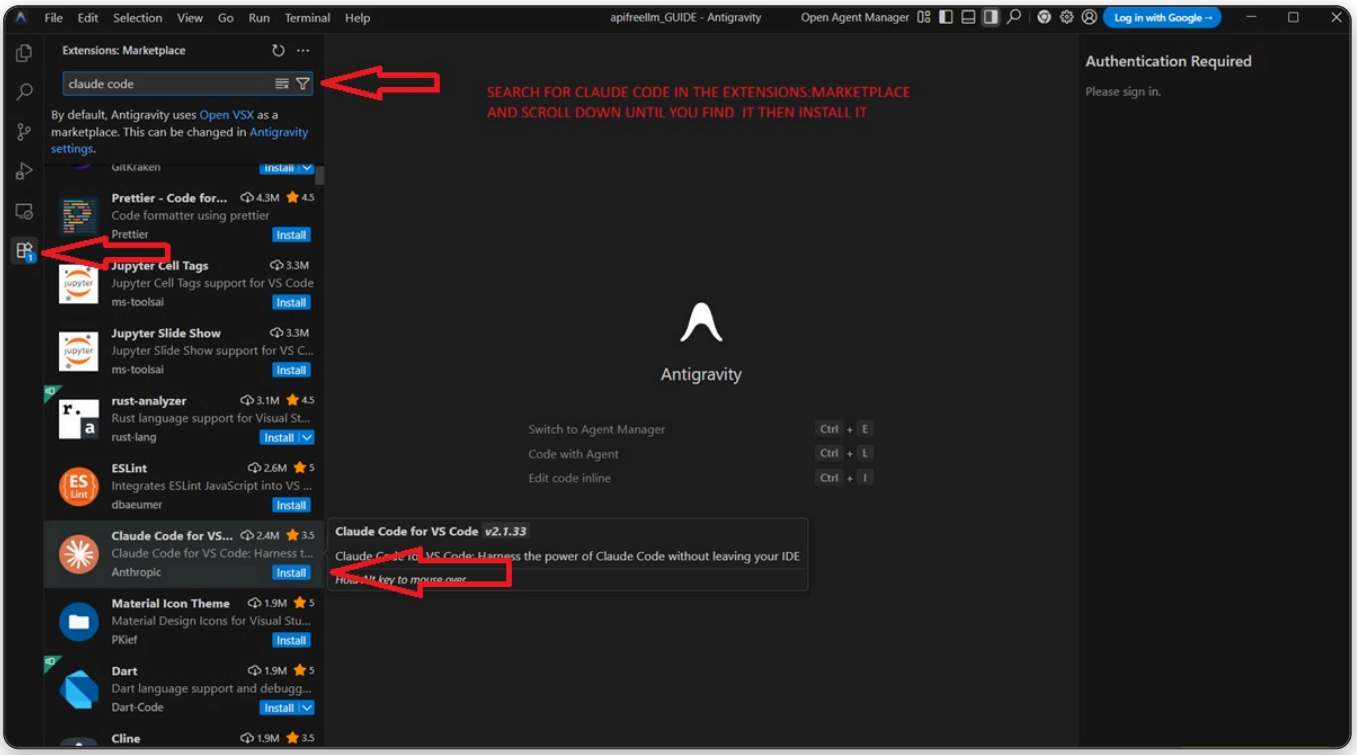


अपने वर्कस्पेस नाम के बगल में तीन डॉट्स पर क्लिक करें और "Focus Editor" चुनें।

Claude Code को VS Code एक्सटेंशन के रूप में इंस्टॉल करना

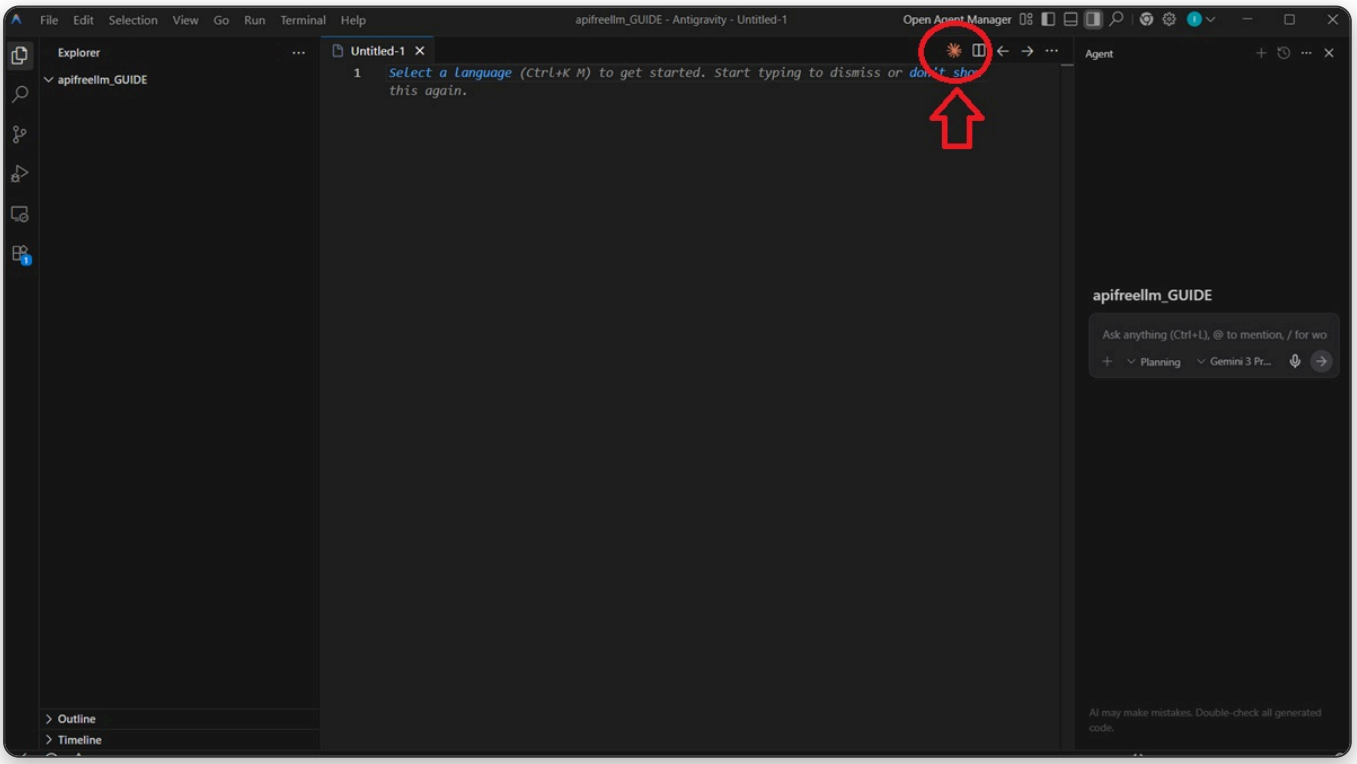
अब जब एडिटर खुला है, तो Claude Code इंस्टॉल करने का समय है। चूंकि एडिटर VS Code पर आधारित है, आपको एक्सटेंशन मार्केटप्लेस तक पहुंच है। लेफ्ट साइडबार में **Extensions आइकन** पर क्लिक करें (या `Ctrl+Shift+X` दबाएं)।

सर्च बार में, "**claude code**" टाइप करें। आपको रिजल्ट्स में नीचे स्क्रॉल करना पड़ सकता है — Anthropic द्वारा "**Claude Code for VS Code**" खोजें। जब मिल जाए, **Install** बटन पर क्लिक करें।



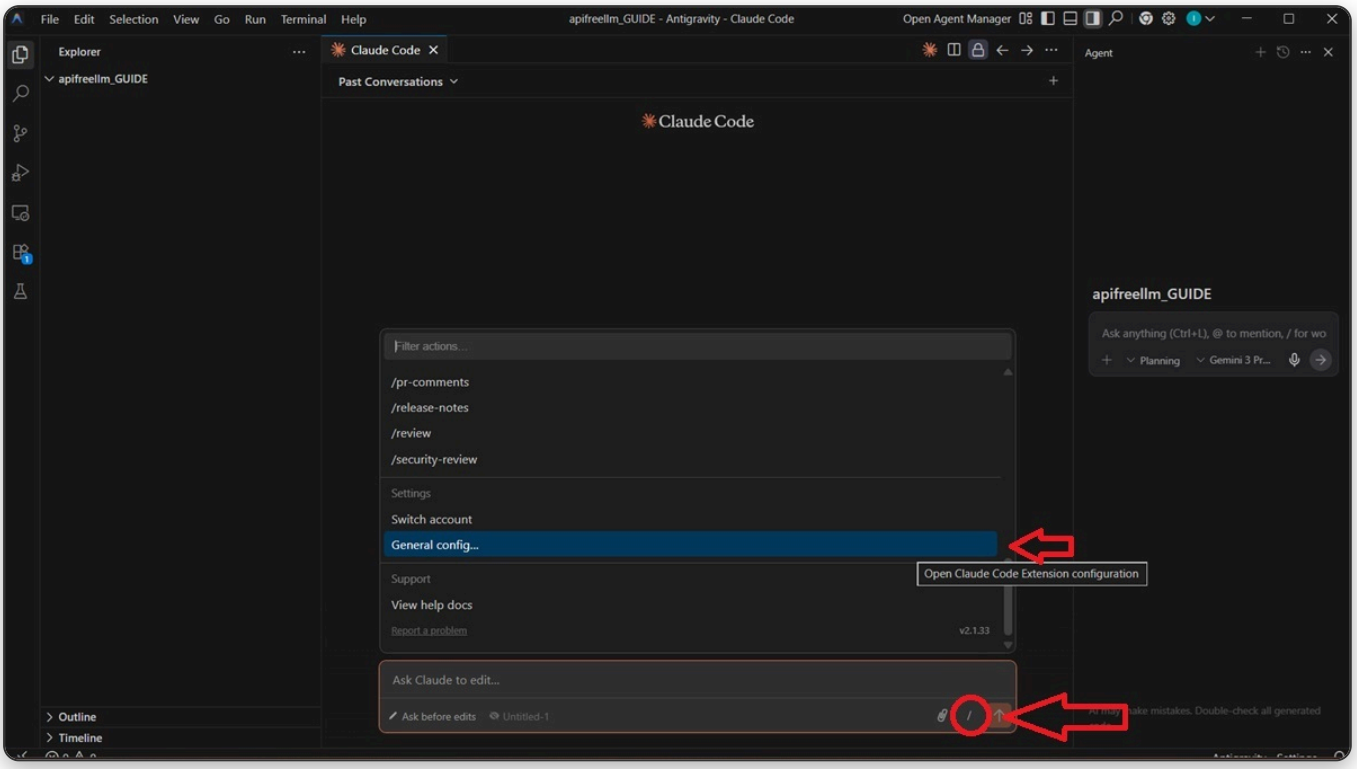
Extensions मार्केटप्लेस में "claude code" सर्च करें, नीचे स्क्रॉल करके खोजें, और Install पर क्लिक करें।

इंस्टॉल होने के बाद, Extensions पैनल बंद करें और एक नई फाइल खोलें (या अपने प्रोजेक्ट में कोई भी फाइल)। आपको एडिटर के ऊपर-दाएं क्षेत्र में एक छोटा Claude Code **आइकन** दिखाई देगा — यह एक छोटे नारंगी सिंबल जैसा दिखता है। Claude Code चैट पैनल खोलने के लिए इस पर क्लिक करें।



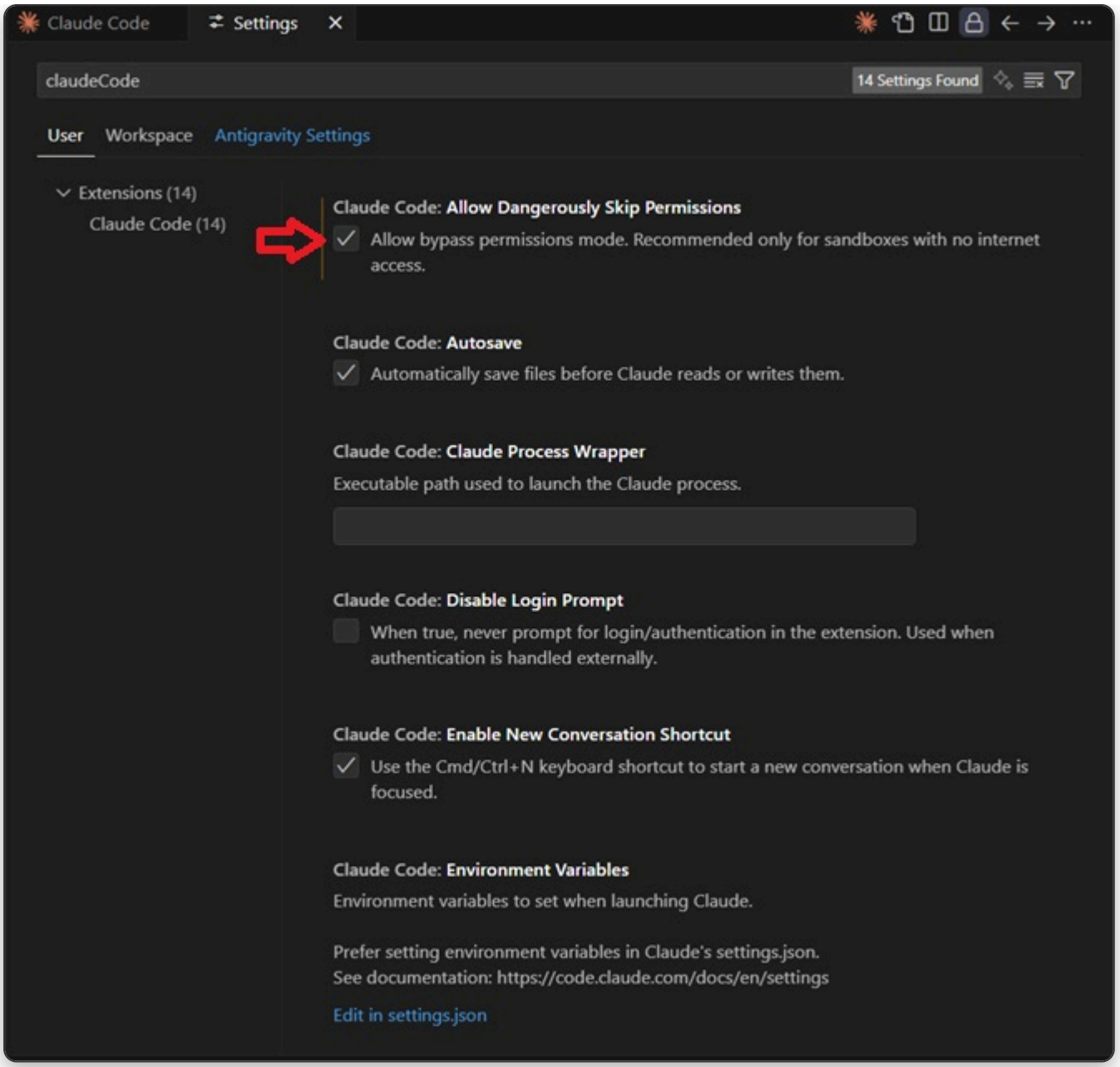
Claude Code बटन (गोले में) एडिटर के ऊपर-दाएं में दिखाई देता है। Claude Code चैट खोलने के लिए इस पर क्लिक करें।

Claude Code आपसे अपने Anthropic अकाउंट (Pro सब्सक्रिप्शन वाले) से साइन इन करने के लिए कहेगा। साइन इन करने के बाद, आपको इसे कॉन्फ़िगर करना होगा। Claude Code चैट पैनल में, पैनल के नीचे **/** बटन पर क्लिक करें। विभिन्न कमांड्स के साथ एक मेन्यू दिखाई देगा। **Settings** सेक्शन तक नीचे स्क्रॉल करें और Claude Code एक्सटेंशन कॉन्फ़िगरेशन खोलने के लिए **"General config..."** पर क्लिक करें।



नीचे "/" बटन पर क्लिक करें, फिर Settings तक स्क्रॉल करें और कॉन्फ़िगरेशन खोलने के लिए "General config..." चुनें।

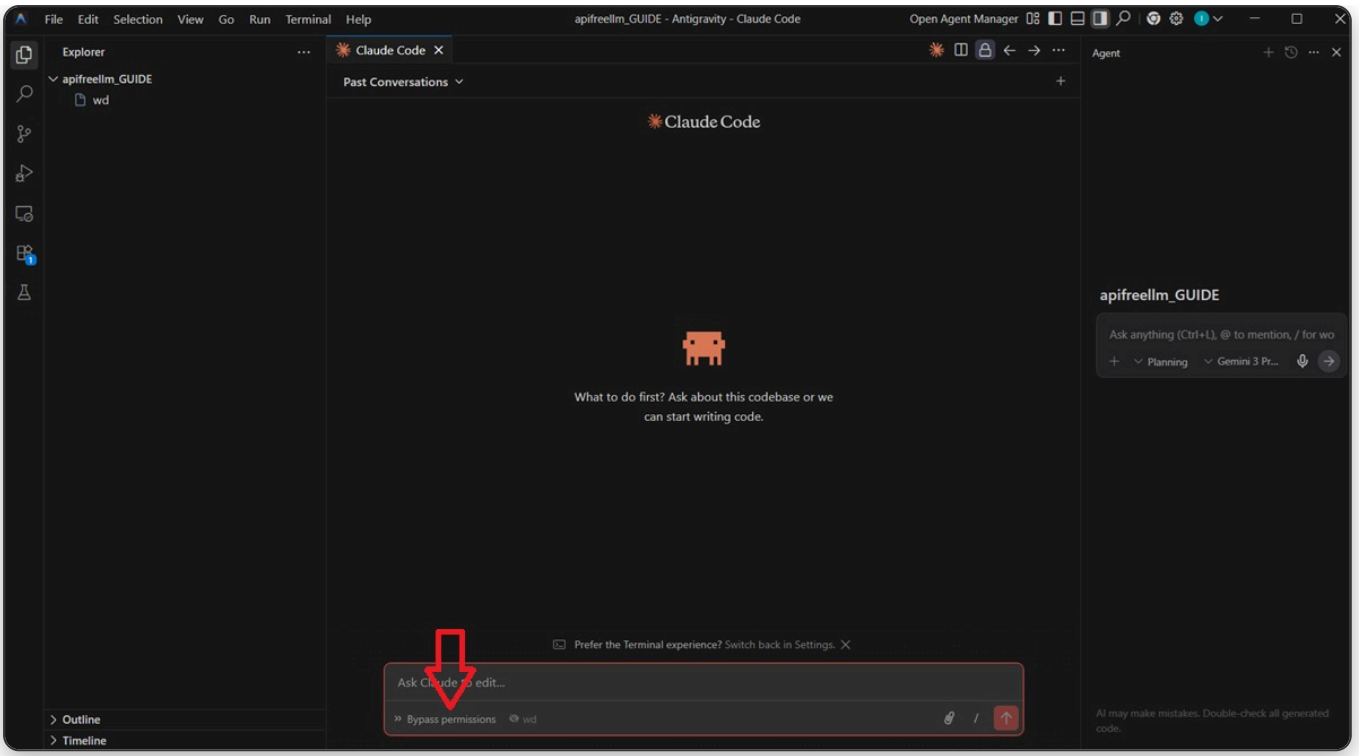
खुलने वाले कॉन्फ़िगरेशन पैनल में, **"Allow Dangerously Skip Permissions"** नामक विकल्प खोजें। इस टॉगल को सक्षम करें। सक्रिय होने के बाद, आप अपने परमिशन मोड के रूप में **"Bypass permissions"** चुन सकेंगे, जो Claude Code को पूरी तरह से स्वायत्त रूप से काम करने की अनुमति देता है — फाइलें पढ़ना, कोड लिखना, टर्मिनल कमांड चलाना, और हर कदम पर पुष्टि मांगे बिना बदलाव करना।



Claude Code सेटिंग्स में "Allow Dangerously Skip Permissions" टॉगल सक्षम करें, फिर "Bypass permissions" चुनें।

> **महत्वपूर्ण**: Bypass permissions सक्षम होने पर, Claude Code बिना मैनुअल अनुमोदन के आपके सिस्टम पर कार्रवाई करेगा। यह एक प्रवाहमय डेवलपमेंट वर्कफ्लो के लिए आवश्यक है, लेकिन यह जिम्मेदारी के साथ आता है। जो कोड आप इसे चलाने के लिए कहते हैं उसके बारे में सावधान रहें, और कभी भी इसे अविश्वसनीय रिपॉजिटरी या संदिग्ध URL पर न लगाएं। AI एजेंट्स को prompt injection के माध्यम से exploit किया जा सकता है — फाइलों या वेबसाइटों में छिपी दुर्भावनापूर्ण सामग्री जो एजेंट को हानिकारक कमांड execute करने के लिए ट्रिक करती है। हमेशा Claude Code क्या कर रहा है इसकी समीक्षा करें, खासकर बाहरी रिसोर्सेज के साथ काम करते समय।

हम यह भी सलाह देते हैं कि नई चैट्स के लिए "Bypass permissions" **को डिफॉल्ट सिलेक्शन** बनाने वाली सेटिंग सक्षम करें, ताकि हर बार नई बातचीत शुरू करते समय इसे मैनुअली चुनने की ज़रूरत न हो। अब Claude Code बंद करें और फिर से खोलें (Claude Code आइकन बटन पर दोबारा क्लिक करके)। अब से, आपको Claude Code चैट पैनल के नीचे परमिशन सिलेक्टर बटन में "Bypass permissions" विकल्प उपलब्ध दिखाई देगा। इसे सक्रिय करने के लिए इस पर क्लिक करें।



स्क्रीनशॉट में दिखाए गए बटन से "Bypass permissions" चुनें।

Bypass permissions सक्रिय होने पर, Claude Code कमांड execute करेगा, फाइलें बनाएगा, कोड एडिट करेगा, और टर्मिनल ऑपरेशन पूरी तरह से अपने आप चलाएगा — हर कदम पर आपकी अनुमति मांगे बिना। यही वह चीज़ है जो आपको **डेवलपमेंट वर्कफ्लो का 100% ऑटोमेट** करने की अनुमति देती है: आप बताते हैं कि आप क्या बनाना चाहते हैं, और Claude Code शुरू से अंत तक स्वायत्त रूप से बनाता है। हर फाइल बदलाव या कमांड execution पर "Accept" क्लिक करने की ज़रूरत नहीं। आप निर्देश दें, और एजेंट बाकी संभालता है।

अपने उपयोग को स्मार्ट तरीके से मैनेज करना

शुरू से एक बात जान लें: Claude Code के साथ हर इंटरैक्शन **टोकन** खर्च करता है, और आपके सब्सक्रिप्शन की एक उपयोग सीमा है। अच्छी बात यह है कि आप बिल्कुल मॉनिटर कर सकते हैं कि आपने कितना उपयोग किया है। Claude Code चैट पैनल में, **/** बटन पर क्लिक करें — उपलब्ध कमांड्स में आपको अपना **वर्तमान उपयोग** मिलेगा, जो दिखाता है कि आपने कितने टोकन खर्च किए हैं और कितनी क्षमता बची है।

सभी मॉडल एक ही दर से टोकन खर्च नहीं करते। **Claude Opus 4.6** सबसे बुद्धिमान और सक्षम मॉडल है, लेकिन प्रति इंटरैक्शन सबसे अधिक टोकन भी खर्च करता है। छोटे मॉडल जैसे **Sonnet** या **Haiku** कम शक्तिशाली हैं लेकिन उनकी उपयोग सीमा अधिक है और काफी कम टोकन खर्च करते हैं। हमारी सलाह: **जटिल कार्यों के लिए Opus** का उपयोग करें जिनमें गहन reasoning, मल्टी-फाइल बदलाव, या आर्किटेक्चरल निर्णय आवश्यक हों — यहीं इसकी बुद्धिमत्ता वास्तविक अंतर लाती है। सरल कार्यों जैसे त्वरित सुधार, छोटे एडिट, या सीधे सवालों के लिए, अपने Opus टोकन को तब तक बचाने के लिए हल्के मॉडल पर स्विच करें जब वे वास्तव में मायने रखते हों।

आपके Claude टोकन बचाने की एक और रणनीति है: उन सवालों के लिए **Antigravity के बिल्ट-इन एजेंट** का उपयोग करें जिनमें Claude स्तर की बुद्धिमत्ता की ज़रूरत नहीं है। Antigravity कई मॉडल्स को सपोर्ट करता है, लेकिन हम इन त्वरित सवालों के लिए **Gemini** उपयोग करने की सलाह देते हैं — चूंकि Antigravity Google का प्रोडक्ट है, Gemini की उपयोग सीमा सबसे अधिक है और यह प्लेटफॉर्म पर उपलब्ध सबसे तेज़ मॉडल भी है। CSS का कोई quick रिमाइंडर चाहिए? Git कमांड का syntax जानना है? किसी लाइब्रेरी के बारे में उत्सुक हैं? Claude Code के बजाय Antigravity से पूछें। इस तरह, आप अपने Claude टोकन उस भारी डेवलपमेंट कार्य के लिए बचाते हैं जहाँ वे सबसे बड़ा प्रभाव डालते हैं।

जैसा कि आप पहले के स्क्रीनशॉट में देख सकते हैं, हम **Claude Code चैट को बाईं ओर** और **Antigravity के एजेंट चैट को दाईं ओर** रखने की सलाह देते हैं। यह साइड-बाय-साइड लेआउट आपको हर समय दोनों एजेंट्स तक तुरंत पहुंच देता है।

> स्मार्ट वर्कफ्लो: **बिल्डिंग के लिए Opus, त्वरित कार्यों के लिए हल्के मॉडल, सामान्य सवालों के लिए Antigravity।** इस तरह आप अपने Claude सब्सक्रिप्शन की वैल्यू को अधिकतम करते हैं। यदि आपकी Claude उपयोग सीमा कम हो रही है, तो याद रखें कि सरल सवालों के लिए Antigravity का Gemini एजेंट हमेशा आपके बगल में है — अपने Claude टोकन उन डेवलपमेंट कार्यों के लिए बचाने के लिए इसका उपयोग करें जिन्हें वास्तव में उनकी ज़रूरत है।

आवश्यक कॉन्फ़िगरेशन

चाहे आप Claude Code कैसे भी इंस्टॉल करें, ये कॉन्फ़िगरेशन स्टेप्स आपके परिणामों में नाटकीय सुधार करेंगे:

- **CLAUDE.md फाइल का उपयोग करें** — अपने प्रोजेक्ट रूट में एक `CLAUDE.md` फाइल रखें। यह फाइल हर सेशन की शुरुआत में Claude Code द्वारा स्वचालित रूप से पढ़ी जाती है। अपनी प्रोजेक्ट संरचना, कोडिंग कन्वेंशन, टेक स्टैक, और एजेंट को फॉलो करने के लिए कोई भी नियम बताने के लिए इसका उपयोग करें। इसे अपने AI डेवलपर के लिए onboarding डॉक्यूमेंटेशन के रूप में सोचें।
- **अपने प्रोजेक्ट को व्यवस्थित रखें** — AI एजेंट्स साफ, अच्छी तरह से संरचित कोडबेस के साथ नाटकीय रूप से बेहतर काम करते हैं। यदि आपका कोड गड़बड़ है, तो एजेंट गड़बड़ आउटपुट देगा। अच्छी फोल्डर संरचना, स्पष्ट नामकरण कन्वेंशन, और सुसंगत पैटर्न बड़ा अंतर लाते हैं।
- **वर्शन कंट्रोल का उपयोग करें** — हमेशा Git initialized के साथ काम करें। यह आपको एक सेफ्टी नेट देता है। यदि एजेंट गलती करता है, तो आप तुरंत revert कर सकते हैं। यह एजेंट को आपके लिए कमिट बनाने की भी अनुमति देता है, जो यह ट्रैक करने के लिए आश्चर्यजनक रूप से उपयोगी है कि क्या बदला और क्यों।

आवश्यक टूल्स: Git & GitHub CLI

कुछ भी बनाना शुरू करने से पहले, आपके सिस्टम पर दो टूल्स इंस्टॉल होने चाहिए: **Git** और **GitHub CLI (gh)**। ये किसी भी आधुनिक डेवलपमेंट वर्कफ्लो के लिए मौलिक हैं, और ये वो हैं जो आपके AI एजेंट्स को स्वायत्त रूप से आपके कोड रिपॉजिटरी मैनेज करने की अनुमति देते हैं।

Git और GitHub CLI क्यों मायने रखते हैं

Git वह वर्शन कंट्रोल सिस्टम है जो आपके प्रोजेक्ट में हर बदलाव को ट्रैक करता है। यह आपका सेफ्टी नेट है: अगर Claude Code गलती करता है, तो आप तुरंत revert कर सकते हैं। यह एजेंट को कमिट बनाने, ब्रांच मैनेज करने, और आपके प्रोजेक्ट के विकास का एक साफ इतिहास रखने की भी अनुमति देता है — सब स्वचालित रूप से।

GitHub CLI (gh) एक कमांड-लाइन टूल है जो टर्मिनल से सीधे GitHub तक पहुंच देता है। यह पूर्ण ऑटोमेशन की कुंजी है: एक बार `gh` इंस्टॉल और authenticated हो जाने पर, Claude Code रिपॉजिटरी बना सकता है, कोड पुश कर सकता है, pull requests मैनेज कर सकता है, रिपॉजिटरी सेटिंग्स कॉन्फ़िगर कर सकता है, डिप्लॉयमेंट के लिए GitHub Actions सेट अप कर सकता है, secrets जोड़ सकता है, और बहुत कुछ — सब अपने टर्मिनल से, बिना आपको ब्राउज़र में GitHub खोलने की ज़रूरत।

Git और GitHub CLI इंस्टॉल करना

इन टूल्स को इंस्टॉल करने का सबसे सरल तरीका है **Claude Code या Antigravity से करने के लिए कहना**। बस अपने एजेंट को बताएं: *"मेरे सिस्टम पर Git और GitHub CLI इंस्टॉल करो।"* एजेंट आपका ऑपरेटिंग सिस्टम detect करेगा और उपयुक्त इंस्टॉलेशन कमांड चलाएगा। Windows पर, यह आमतौर पर `winget` का उपयोग करेगा या installers डाउनलोड करेगा; macOS पर, `brew` ; Linux पर, `apt` या आपके सिस्टम का पैकेज मैनेजर।

यदि आप उन्हें मैनुअली इंस्टॉल करना पसंद करते हैं, तो आप Git को आधिकारिक वेबसाइट से और GitHub CLI को इसके GitHub releases पेज से डाउनलोड कर सकते हैं। लेकिन AI को इसे संभालने देना तेज़ है और सामान्य इंस्टॉलेशन गलतियों से बचाता है।

GitHub CLI को authenticate करना

इंस्टॉलेशन के बाद, आपको लॉग इन करना होगा ताकि `gh` आपके GitHub अकाउंट तक पहुंच सके। फिर से, आप बस अपने एजेंट से कह सकते हैं: *"मुझे GitHub CLI में लॉग इन कराओ।"* एजेंट `gh auth login` चलाएगा और authentication flow के माध्यम से आपका मार्गदर्शन करेगा, जिसमें आमतौर पर एक ब्राउज़र लिंक खोलना और एक कोड दर्ज करना शामिल है। एक बार authenticated होने के बाद, एजेंट के पास आपके GitHub रिपॉजिटरी तक पूरी पहुंच है।

> **यह एक गेम-चेंजर है।** `gh` authenticated होने पर, आप Claude Code को ऐसी चीज़ें बता सकते हैं जैसे: *"my-app नाम से एक नई private repository बनाओ, प्रोजेक्ट initialize करो, और कोड पुश करो।"* या बाद में: *"एक GitHub Action सेट अप करो जो main में हर push पर मेरे सर्वर पर डिप्लॉय करे।"* एजेंट सब कुछ संभालता है — फाइलें बनाना, secrets कॉन्फ़िगर करना, workflows सेट अप करना — बिना आपको कभी एडिटर छोड़ने की ज़रूरत। यही सच्चा डेवलपमेंट ऑटोमेशन है।

3. आपका पहला बिल्ड: ज़ीरो से लाइव तक

> *आपका एनवायरनमेंट तैयार है। Claude Code खुला है, Antigravity चल रहा है, Git और GitHub CLI कॉन्फ़िगर हैं। अब कुछ असली बनाने का समय है। इस बिंदु से आगे, यह कोर्स सेटअप से रणनीति की ओर शिफ्ट होता है — व्यावहारिक तकनीकें और अंतर्दृष्टि जो AI एजेंट्स के साथ बनाते समय आपको वास्तविक लाभ देंगी।*

कभी शुरू से न बनाएं

इस पूरे कोर्स में सबसे महत्वपूर्ण सलाह यही है: **कभी भी शुरू से न बनाएं।**

भले ही Claude Opus एक अविश्वसनीय रूप से उन्नत और बुद्धिमान मॉडल है, यह गलतियाँ करेगा। हर AI मॉडल करता है। यह आपकी प्रोजेक्ट संरचना को गलत समझ सकता है, एक पुरानी लाइब्रेरी का उपयोग कर सकता है, एक असंगत फाइल लेआउट बना सकता है, या ऐसा कोड जनरेट कर सकता है जो प्रोजेक्ट बढ़ने पर एक साथ नहीं जुड़ता। एक खाली फोल्डर से शुरू करने का मतलब है AI को बिना किसी संदर्भ बिंदु के सैकड़ों निर्णय लेने होंगे — और उनमें से कुछ निर्णय अनिवार्य रूप से गलत होंगे।

वास्तविकता यह है: **जो आप बनाना चाहते हैं उसका 99.9% पहले से किसी न किसी रूप में मौजूद है।** चाहे वह ई-कॉमर्स स्टोर हो, SaaS डैशबोर्ड हो, पोर्टफोलियो साइट हो, सोशल मीडिया ऐप हो, या बुकिंग प्लेटफॉर्म हो — किसी ने पहले से कुछ ऐसा ही बना रखा है। और इनमें से कई प्रोजेक्ट्स ओपन-सोर्स हैं, GitHub पर टेम्पलेट के रूप में उपलब्ध हैं, क्लोन करने और कस्टमाइज़ करने के लिए तैयार हैं।

आपका वर्कफ्लो हमेशा एक ही तरीके से शुरू होना चाहिए: **पहले टेम्पलेट खोजें।** GitHub पर जाएं और ओपन-सोर्स प्रोजेक्ट्स खोजें जो आप जो बनाना चाहते हैं उससे मेल खाते हों। एक ऐसा खोजें जो आपकी विज़न के करीब हो, इसे अपने प्रोजेक्ट फोल्डर में क्लोन करें, और फिर Claude Code को इसकी ओर पॉइंट करें। अब ज़ीरो से बनाने के बजाय, एजेंट एक मौजूदा, काम करने वाले कोडबेस को modify, improve और customize कर रहा है। गुणवत्ता और गति में अंतर बहुत बड़ा है।

> **टेम्पलेट्स चीटिंग नहीं हैं — ये रणनीति हैं।** प्रोफेशनल डेवलपर्स हमेशा बॉयलरप्लेट और स्टार्टर किट्स का उपयोग करते हैं। आप किसी का प्रोडक्ट कॉपी नहीं कर रहे। आप एक संरचनात्मक नींव का उपयोग कर रहे हैं और उसके ऊपर अपना अनूठा प्रोडक्ट बना रहे हैं। AI नाटकीय रूप से बेहतर प्रदर्शन करता है जब उसके पास फॉलो करने के लिए मौजूदा पैटर्न हों बजाय सब कुछ शून्य से बनाने के।

सही स्टैक चुनना

यदि आपकी टेम्पलेट खोज खाली आती है और आपको वास्तव में एक नया प्रोजेक्ट शुरू करना है, तो आप जो तकनीक चुनते हैं वह अंतर ला सकती है — खासकर AI एजेंट्स के साथ काम करते समय। कहा जाता है, कोई एक अनिवार्य विकल्प नहीं है। सबसे अच्छा स्टैक वो है जो आपको सबसे तेज़ी से एक काम करने वाले प्रोडक्ट तक पहुंचाए।

Claude Code, सभी LLMs की तरह, मौलिक रूप से **वेब-आधारित कोड** लिखने में बेहतर है: HTML, CSS, JavaScript, और TypeScript। यह इंटरनेट की भाषा है, और इन मॉडल्स को इसी पर सबसे अधिक प्रशिक्षित किया गया है। आपका प्रोजेक्ट वेब तकनीकों के जितना करीब रहे, AI उतना बेहतर प्रदर्शन करेगा।

वेब एप्लिकेशन के लिए, **Next.js** एक उत्कृष्ट विकल्प है। यह बिल्ट-इन सर्वर-साइड रेंडरिंग के साथ एक आधुनिक React फ्रेमवर्क है, जो SEO के लिए महत्वपूर्ण है। Claude Code Next.js प्रोजेक्ट्स के साथ असाधारण रूप से अच्छा काम करता है: यह फाइल-आधारित routing, API routes, server components, और पूरे ecosystem को समझता है। आपको GitHub पर वर्चुअली किसी भी प्रकार के एप्लिकेशन के लिए बड़ी संख्या में Next.js टेम्पलेट मिलेंगे।

डेस्कटॉप एप्लिकेशन (Windows, macOS, या Linux के लिए executables) के लिए, **Electron** एक बढ़िया विकल्प है। Electron आपको HTML, CSS, और JavaScript का उपयोग करके डेस्कटॉप ऐप्स बनाने देता है — वही वेब तकनीकें जिनमें Claude उत्कृष्ट है। चूंकि UI अनिवार्य रूप से एक native window के अंदर rendered वेब पेज है, AI वेबसाइट बनाने जितनी ही आसानी से सुंदर, कार्यात्मक डेस्कटॉप एप्लिकेशन बना सकता है।

लेकिन यहाँ एक महत्वपूर्ण बारीकी है: **पुराने स्टैक में एक अच्छी तरह से बना टेम्पलेट लगभग हमेशा एक आधुनिक स्टैक में शुरू से बनाने से बेहतर होता है।** आपको एक PHP, jQuery, या Laravel प्रोजेक्ट मिल सकता है जो बिल्कुल वही है जो आपको चाहिए — पूर्ण, अच्छी तरह से संरचित, battle-tested, सभी फीचर्स पहले से implemented। ऐसे मामले में, इसका उपयोग करें। AI एजेंट्स किसी भी तकनीक के साथ काम कर सकते हैं, और एक ठोस, रेडी-मेड नींव से बचाया गया समय एक नए फ्रेमवर्क के सैद्धांतिक लाभों से कहीं अधिक है। Claude Code PHP, Python, Ruby, या किसी अन्य कोडबेस को बिल्कुल ठीक से समझ और modify कर सकता है।

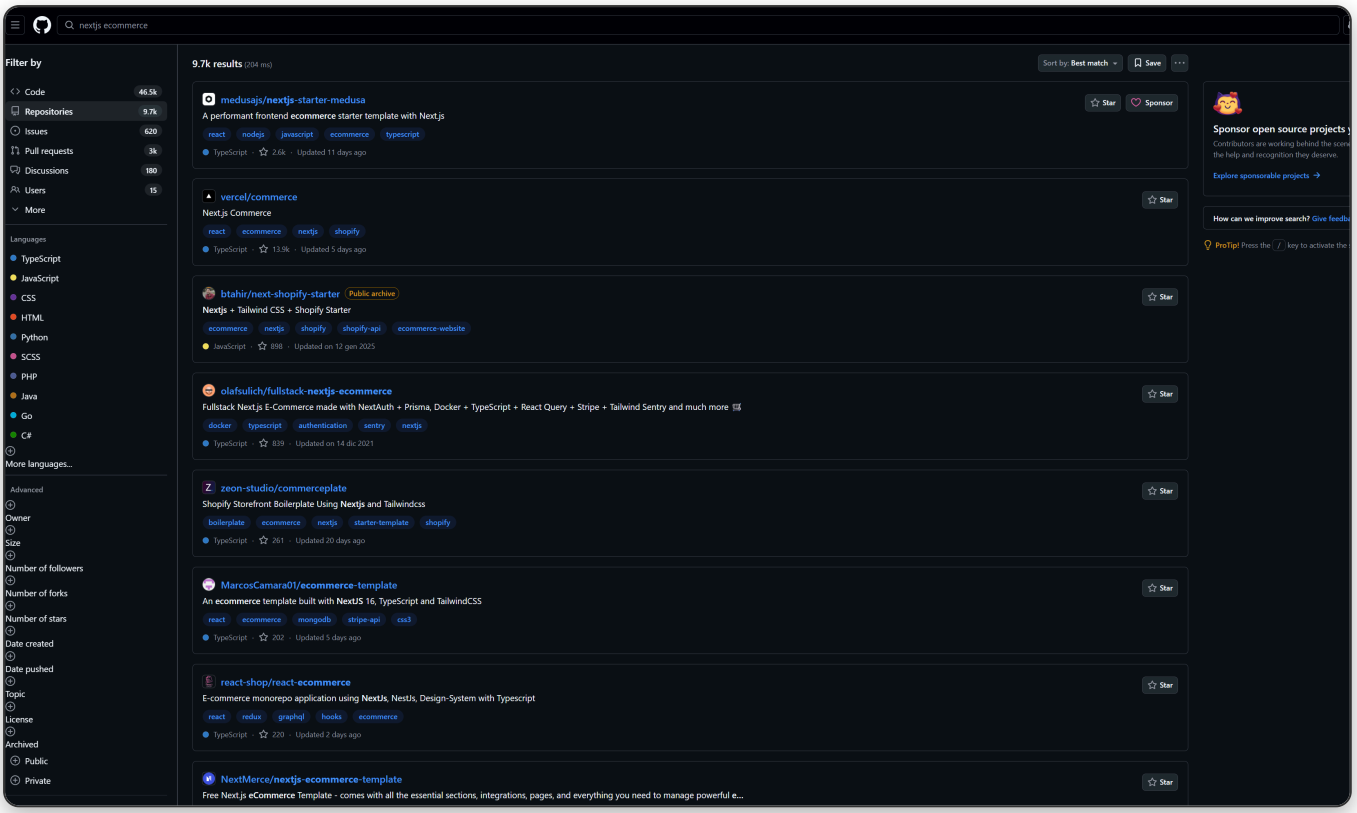
प्राथमिकता सरल है: **सबसे अच्छा उपलब्ध टेम्पलेट खोजें।** यदि आपको समान गुणवत्ता के दो टेम्पलेट मिलते हैं और एक Next.js का उपयोग करता है जबकि दूसरा PHP का, तो Next.js चुनें। लेकिन अगर PHP टेम्पलेट अधिक पूर्ण, अधिक फीचर-रिच, और बेहतर maintained है — बिना किसी हिचकिचाहट के उसे चुनें। शुरुआती बिंदु की गुणवत्ता और पूर्णता स्टैक की आधुनिकता से अधिक मायने रखती है।

> **अंगूठे का नियम:** जो भी मिले उसका उपयोग करें जो आपको आपके लक्ष्य के सबसे करीब ले जाए। जब उपलब्ध हो तो आधुनिक वेब तकनीकों (Next.js, Electron, React Native) का लक्ष्य रखें, लेकिन सिर्फ इसलिए कि कोई पुरानी स्टैक का उपयोग करता है, किसी बढ़िया टेम्पलेट को कभी न ठुकराएं। एक ठोस नींव से बचाया गया समय हमेशा स्टैक शुद्धता से अधिक मूल्यवान होता है।

व्यावहारिक उदाहरण: टेम्पलेट खोजना

मान लीजिए आप एक ई-कॉमर्स स्टोर बनाना चाहते हैं। Claude Code को *"मेरे लिए शुरू से एक ई-कॉमर्स साइट बनाओ"* बताने के बजाय, GitHub खोलें और **"nextjs ecommerce"** जैसा कुछ सर्च करें। आपको प्रोडक्ट लिस्टिंग, शॉपिंग कार्ट,

चेकआउट फ्लो, और पेमेंट इंटीग्रेशन पहले से बने हुए दर्जनों रेडी-मेड प्रोजेक्ट्स मिलेंगे।



"nextjs ecommerce" के लिए एक त्वरित GitHub सर्च पहले से कई आशाजनक टेम्पलेट्स दिखाता है।

एक महत्वपूर्ण बात ध्यान में रखें: GitHub पर कई टेम्पलेट्स **फ्रीमियम प्रोजेक्ट्स** हैं — कोर ओपन-सोर्स और फ्री है, लेकिन कुछ फीचर्स या प्रीमियम वर्शन के लिए पेमेंट आवश्यक है। किसी टेम्पलेट पर commit करने से पहले हमेशा रिपॉजिटरी की README और लाइसेंस चेक करें। ऐसी किसी भी चीज़ से बचें जिसमें paid subscriptions या hidden costs हों जिनकी आपको ज़रूरत नहीं है।

सर्च रिजल्ट्स को देखते हुए, हम **fullstack-nextjs-ecommerce** स्पॉट कर सकते हैं — और यह एक उत्कृष्ट शुरुआती बिंदु है। आइए विश्लेषण करें क्यों। प्रोजेक्ट TypeScript के साथ Next.js का उपयोग करता है, जो बिल्कुल वही है जो हम AI-सहायित डेवलपमेंट के लिए चाहते हैं। लेकिन जो वास्तव में अलग बनाता है वह है जो पहले से इंटीग्रेटेड है: **पेमेंट के लिए** Stripe, डेटाबेस के रूप में Prisma **के साथ** PostgreSQL, और **authentication के लिए** NextAuth। ये किसी भी वेब एप्लिकेशन के तीन सबसे महत्वपूर्ण — और सबसे अधिक error-prone — हिस्से हैं।

Stripe पहले से इंटीग्रेटेड होना विशेष रूप से मूल्यवान है। पेमेंट प्रोसेसिंग में webhooks, session management, error handling, और edge cases शामिल हैं जो सही करने में आश्चर्यजनक रूप से मुश्किल हो सकते हैं। जब पेमेंट में कुछ गलत होता है, तो आपके ग्राहक पीड़ित होते हैं — फेल चार्ज, डुप्लिकेट पेमेंट, या टूटे checkout flows ऐसे bugs हैं जो विश्वास नष्ट करते हैं और आपको असली पैसे खर्च करा सकते हैं। एक काम करने वाले Stripe इंटीग्रेशन वाले टेम्पलेट से शुरू करना आपको इन सिरदर्दों से बचाता है।

प्रोजेक्ट अपने डेटाबेस के रूप में **PostgreSQL** का भी उपयोग करता है, जो एक ठोस विकल्प है। Postgres विश्वसनीय, अच्छी तरह से documented है, MySQL जैसे विकल्पों की तुलना में थोड़ा बेहतर सुरक्षा defaults प्रदान करता है, और Amazon AWS, Hetzner, या समान providers पर आसानी से होस्ट किया जा सकता है — हम अगले अध्याय में सर्वर सेटअप और डिप्लॉयमेंट को कवर करेंगे। **NextAuth** पहले से wired up होने का मतलब है यूज़र authentication आउट ऑफ द बॉक्स संभाला गया है, एक और जटिल टुकड़ा जो आपको शुरू से नहीं बनाना है।

सही टेम्पलेट से शुरू करने की यही शक्ति है: पेमेंट, डेटाबेस, और authentication सेट अप करने में दिन (या सप्ताह) बिताने के बजाय — जो सभी गलत होने में आसान हैं — आप एक ऐसे प्रोजेक्ट से शुरू करते हैं जहाँ ये महत्वपूर्ण सिस्टम पहले से काम

करते हैं। फिर आप पूरी तरह से अपनी विज़न के अनुसार प्रोडक्ट को customize करने पर ध्यान केंद्रित कर सकते हैं: डिज़ाइन बदलना, अपने प्रोडक्ट्स जोड़ना, बिज़नेस लॉजिक modify करना, और वो फीचर्स बनाना जो आपके स्टोर को अनूठा बनाते हैं।

जब आपको अपना टेम्पलेट मिल जाए, तो अगला कदम सरल है: **इसे डाउनलोड करें और अपने वर्कस्पेस फोल्डर में रखें।** उस फोल्डर को Claude Code (या Antigravity) से खोलें और काम शुरू करें। आप Claude Code को सीधे टेम्पलेट modify करने के लिए कह सकते हैं — branding बदलें, नए पेज जोड़ें, लेआउट rework करें, नए फीचर्स इंटीग्रेट करें — या आप टेम्पलेट को अपने प्रोजेक्ट के लिए एक **reference** के रूप में उपयोग कर सकते हैं। भले ही आप कुछ थोड़ा अलग बना रहे हों, एक ही वर्कस्पेस में टेम्पलेट होने का मतलब है Claude Code इसे देख सकता है, कोड पैटर्न study कर सकता है, और जो लिखता है उसके लिए उन्हें नींव के रूप में उपयोग कर सकता है।

यह एक महत्वपूर्ण बिंदु है: **हमेशा Claude Code को reference के लिए कुछ दें।** जब एजेंट के पास देखने के लिए एक ठोस, काम करने वाला कोडबेस है, तो वह काफी बेहतर कोड लिखता है। वह समान पैटर्न फॉलो करता है, समान conventions का उपयोग करता है, और ऐसा आउटपुट देता है जो सुसंगत और विश्वसनीय है। जब उसके पास reference के लिए कुछ नहीं है और उसे शुरू से सब कुछ generate करना है, तभी गलतियाँ होती हैं — असंगत फाइल structures, गलत library versions, कोड जो एक साथ नहीं जुड़ता। एक टेम्पलेट एक anchor की तरह काम करता है जो AI को grounded और उच्च-गुणवत्ता, coherent results देने में मदद करता है।

> **क्या होगा अगर टेम्पलेट में पेमेंट या डेटाबेस नहीं है?** वो भी ठीक है। इस कोर्स में रेडी-मेड Stripe इंटीग्रेशन फाइलें शामिल हैं जो आप सीधे Claude Code को किसी भी प्रोजेक्ट में पेमेंट इंटीग्रेट करने के लिए दे सकते हैं। डेटाबेस के लिए, हम PostgreSQL या जो भी डेटाबेस टेम्पलेट पहले से उपयोग करता है उसकी सलाह देते हैं — जब तक कोई मज़बूत कारण न हो, टेम्पलेट की choices से न लड़ें। Authentication के लिए भी यही बात लागू होती है: अगर टेम्पलेट NextAuth, Clerk, या कोई अन्य auth system उपयोग करता है, तो उसके साथ काम करें। लक्ष्य है जो पहले से बना है उसका लाभ उठाना, सब कुछ बदलना नहीं।

4. कोड से प्रोडक्शन तक

> *आपका प्रोडक्ट बन चुका है। अब इसे पेमेंट स्वीकार करने, सर्वर पर चलने, और दुनिया भर के यूज़र्स के लिए तेज़ और सुरक्षित होने की ज़रूरत है। यह अध्याय उन आवश्यक सेवाओं को कवर करता है जो आपके प्रोजेक्ट को लोकल कोड से एक लाइव, प्रोडक्शन-रेडी बिज़नेस में बदलती हैं।*

> **इस अध्याय के बारे में एक नोट।** हम यहाँ चीज़ों को संक्षिप्त और सीधा रख रहे हैं। हमारा लक्ष्य आपको बताना है **क्या** करना है और **क्यों** — लंबे ट्यूटोरियल नहीं लिखना जो आपका समय बर्बाद करें। कोई भी LLM (Claude, Gemini, ChatGPT) विवरण समझा सकता है, हर कदम पर आपका मार्गदर्शन कर सकता है, और आपके सवालों का जवाब एक स्थिर पेज से कहीं बेहतर दे सकता है। जब भी कुछ स्पष्ट न हो, बस अपने एजेंट से पूछें: *"मैं एक कोर्स फॉलो कर रहा हूँ और इसमें कहा गया है कि मुझे [X करना] है। क्या आप समझा सकते हैं इसका क्या मतलब है और मुझे step-by-step बता सकते हैं?"* यह तेज़, अधिक personalized, और हमेशा अप-टू-डेट है।

Stripe के साथ पेमेंट

अगर आपका प्रोडक्ट कुछ बेचता है, तो आपको एक पेमेंट प्रोसेसर चाहिए। **Stripe** इंडस्ट्री स्टैंडर्ड है: विश्वसनीय, अच्छी तरह से documented, और वस्तुतः हर AI एजेंट द्वारा सपोर्टेड क्योंकि यह कितना व्यापक रूप से उपयोग किया जाता है।

आपको यह करना होगा:

- **stripe.com पर एक Stripe अकाउंट बनाएं।** आपको **API keys** (frontend के लिए एक publishable key, backend के लिए एक secret key) और **test mode** (डेवलपमेंट के लिए नकली पेमेंट) और **live mode** (असली पैसे) दोनों तक पहुँच मिलेगी।
- **Stripe dashboard में webhooks सेट अप करें।** Webhooks आपके सर्वर पर URLs हैं जिन्हें Stripe तब कॉल करता है जब कुछ होता है — पेमेंट सफल होता है, subscription renew होता है, charge फेल होता है। इस तरह आपका ऐप जानता है कब subscription activate करना है, ऑर्डर confirm करना है, या failure handle करना है। आपको **लोकल टेस्टिंग** (Stripe के पास एक CLI टूल है जो events को आपके localhost पर forward करता है) और **प्रोडक्शन** (आपके लाइव सर्वर पर pointing) दोनों के लिए webhooks चाहिए। लाइव जाने से पहले हमेशा सब कुछ locally काम करा लें।
- **अपने प्रोजेक्ट में Stripe इंटीग्रेट करें।** अगर आपके टेम्पलेट में पहले से Stripe है, तो बस अपनी API keys लगाएं। अगर नहीं है, तो इस कोर्स में रेडी-मेड Stripe इंटीग्रेशन फाइलें शामिल हैं — उन्हें अपने वर्कस्पेस में डालें और Claude Code को इंटीग्रेट करने के लिए कहें। Stripe one-time payments और recurring subscriptions दोनों सपोर्ट करता है, दोनों hosted checkout pages का उपयोग करते हैं जो card validation, 3D Secure, और PCI compliance आपके लिए handle करते हैं।

एक महत्वपूर्ण नियम: **हमेशा webhooks के माध्यम से server-side पेमेंट verify करें**, frontend पर कभी भरोसा न करें। Webhook Stripe का आपके सर्वर को सीधे बताना है कि पैसा वास्तव में हाथ बदल चुका है — यह सत्य का एकमात्र विश्वसनीय स्रोत है।

होस्टिंग: AWS, Hetzner और अन्य

आपके ऐप को कहीं रहना होगा। अच्छी खबर: **AWS आपको साइन अप करने पर Free Tier देता है — पूरे एक साल** के लिए, आपको एक **t2.micro सर्वर** और एक **micro डेटाबेस** पूरी तरह से मुफ्त मिलता है। यह आपके पहले प्रोजेक्ट को होस्ट करने के लिए पर्याप्त है जब तक आप आइडिया validate करते हैं और यूज़र्स आने शुरू करते हैं।

AWS अकाउंट कैसे सेट अप करें या Free Tier कैसे उपयोग करें नहीं जानते? बस Claude या Antigravity से पूछें — वे आपको हर कदम पर गाइड करेंगे।

जब आप free tier से आगे बढ़ जाएं, तो सर्वर sizing के बारे में यह जानना ज़रूरी है:

- **t3.small** अधिकांश ऐप्स के लिए sweet spot है — अच्छा CPU, पर्याप्त RAM, और उचित pricing (AWS पर ~$16/माह)। "t" instance generation को refer करता है; पुरानी generations (t2, आदि) कभी-कभी कम performance के लिए अधिक कीमत ले सकती हैं, इसलिए नवीनतम उपलब्ध पर बने रहें।
- **Hetzner** एक यूरोपीय विकल्प है जो *काफी सस्ता* है — आप t3.small के बराबर performance लगभग **$4/माह** में पा सकते हैं। यह समान specs के लिए AWS से लगभग 4 गुना सस्ता है।
- **हर ऐप को सर्वर की ज़रूरत नहीं।** अगर आपका प्रोजेक्ट एक static site या JAMstack ऐप है, तो आपको dedicated server की बिल्कुल ज़रूरत नहीं हो सकती। Vercel, Netlify, या यहाँ तक कि Cloudflare Pages जैसे प्लेटफॉर्म इसे मुफ्त या लगभग मुफ्त में होस्ट कर सकते हैं। हमेशा अपने LLM से पूछें: *"मेरे specific ऐप के लिए सबसे अच्छी होस्टिंग क्या है?"*

हमारी सलाह: **अपने पहले साल के लिए AWS Free Tier से शुरू करें**, फिर evaluate करें कि Hetzner या कोई अन्य provider आपके बजट और audience location के लिए अधिक समझदारी भरा है या नहीं। अगर आपके अधिकांश यूज़र्स यूरोप में हैं, तो Hetzner के जर्मन सर्वर आपको कम latency देंगे। अगर आपकी audience ग्लोबल या US-based है, तो AWS regions बेहतर fit हो सकते हैं।

> **SSH keys और AI सर्वर प्रबंधन।** Claude Code को अपने सर्वर से remotely कनेक्ट और manage करने देने के लिए, आपको एक **SSH key** सेट अप करनी होगी। Claude से एक generate करने और अपने सर्वर पर configure करने के लिए कहें। एक बार कनेक्ट होने के बाद, Claude कोड deploy कर सकता है, services manage कर सकता है, issues troubleshoot कर सकता है — सब आपके terminal से। सर्वर management tasks के लिए, हम सर्वोत्तम परिणामों के लिए **Opus** उपयोग करने की सलाह देते हैं, हालांकि Sonnet भी काम करता है अगर आप tokens बचाना चाहते हैं (errors की थोड़ी अधिक संभावना के साथ)।

Cloudflare: Performance और Protection

जब आपका ऐप सर्वर पर है, तो आपको इसे protect करने और speed up करने के लिए कुछ चाहिए जो इसके सामने बैठे। वह है **Cloudflare**। अच्छी खबर: **free plan अधिकांश वेबसाइटों के लिए पर्याप्त से अधिक है**। आपको paid plan की ज़रूरत नहीं है।

लेकिन पहले, आपको एक **domain name** चाहिए। हम सीधे **Cloudflare** या **Namecheap** से खरीदने की सलाह देते हैं — दोनों विश्वसनीय और उचित कीमत वाले हैं। एक बार आपके पास domain हो, तो आपको **DNS** configure करना होगा जो आपके AWS या Hetzner सर्वर के IP address पर point करे। बस अपने LLM से पूछें: *"मैंने [Cloudflare/Namecheap] पर एक domain खरीदा है। मैं DNS कैसे सेट करूं जो मेरे सर्वर [आपका IP] पर point करे?"* वह आपको step-by-step बताएगा।

Cloudflare इतना महत्वपूर्ण क्यों है? यह आपकी साइट को **DDoS attacks**, विभिन्न **security exploits**, और आम वेब vulnerabilities से protect करता है। यह एक **global cache** भी प्रदान करता है, जिसका मतलब है आपका content आपके users के पास के servers से serve होता है, जिससे आपकी साइट worldwide तेज़ हो जाती है। Cloudflare जैसी service के बिना, आप अपनी साइट को गंभीर जोखिमों से expose कर रहे हैं — खासकर अभी, AI era में, जहाँ कोई भी AI tools का उपयोग करके vulnerabilities खोज सकता है, misconfigurations exploit कर सकता है, या खराब तरीके से protected वेबसाइटों से डेटा चुरा भी सकता है। इसे skip न करें।

> **Quick tip: Flexible SSL mode।** Cloudflare सेट अप करते समय, आप **Flexible** SSL mode चुन सकते हैं। इसका मतलब है Cloudflare और आपके सर्वर के बीच connection plain HTTP उपयोग करता है, लेकिन Cloudflare और आपके end users के बीच connection HTTPS है — तो visitors को एक secure padlock दिखता है। यह आपको अपने सर्वर पर SSL certificates configure करने से बचाता है, जो जल्दी शुरू करने के लिए बहुत अच्छा है। संवेदनशील डेटा handle करने वाले प्रोडक्शन ऐप्स के लिए, आपको अंततः **Full** mode (end-to-end encrypted) पर switch करना चाहिए। लेकिन आपके पहले tests और launches के लिए, Flexible बिल्कुल ठीक है और बहुत setup time बचाता है। अगर आप unsure हैं कि कौन सा mode आपके प्रोजेक्ट के लिए fit है, तो अपने LLM से differences समझाने के लिए कहें।

Cloudflare सिर्फ सुरक्षा से कहीं अधिक प्रदान करता है। मुफ्त प्लान में शक्तिशाली फीचर्स शामिल हैं जो कई डेवलपर्स को पता भी नहीं हैं:

- **Cloudflare Pages** — frontend apps (React, Next.js static export, Vue, आदि) के लिए मुफ्त static hosting। आप GitHub पर push करें, Cloudflare अपने आप build और deploy करता है। शून्य

configuration, शून्य लागत। Landing pages, portfolios और JAMstack apps के लिए बिल्कुल सही।

- **Edge Functions (Workers)** — मुफ्त प्लान पर edge पर serverless code चलाएं, अपने users के करीब। API routes, redirects, A/B testing और हल्की backend logic के लिए बहुत अच्छा — बिना किसी dedicated server के।
- **CDN & Caching** — आपके static assets (images, CSS, JS) globally cache होते हैं, जिससे आपकी site दुनिया में कहीं से भी तेज़ लोड होती है।

मुख्य सवाल यह है: **क्या आपकी app को वाकई एक dedicated server की ज़रूरत है, या Cloudflare इसे मुफ्त में handle कर सकता है?** Claude से पूछें: *"मैं [अपनी app का वर्णन करें] बना रहा हूं। क्या मुझे AWS/Hetzner पर dedicated server चाहिए, या मैं Cloudflare Pages और Workers मुफ्त में इस्तेमाल कर सकता हूं?"* Claude आपके specific case का विश्लेषण करेगा और आपको सबसे अच्छा विकल्प बताएगा। आप हैरान हो सकते हैं कि कितने projects पूरी तरह Cloudflare के मुफ्त tier पर चल सकते हैं।

API Keys: सब कुछ Automate करें

यहाँ एक game-changing tip है जो ज़्यादातर tutorials छोड़ देते हैं: **अपने hosting providers के लिए API keys बनाएं** और उन्हें Claude को दें। इससे Claude आपके infrastructure को सीधे terminal से manage कर सकता है — कोई dashboards पर clicking नहीं, कोई copy-paste नहीं, कोई manual काम नहीं।

- **Cloudflare API Key** — अपने Cloudflare dashboard पर जाएं → My Profile → API Tokens → एक token बनाएं। इसके साथ, Claude automatically DNS records configure कर सकता है, Pages projects सेट अप कर सकता है, Workers manage कर सकता है, SSL settings update कर सकता है और भी बहुत कुछ। Claude से कहें: *"यह मेरा Cloudflare API token है। मेरे domain के लिए DNS सेट करो जो मेरे server IP की ओर point करे।"* सेकंडों में हो जाएगा।
- **AWS Access Keys** — AWS IAM पर जाएं → एक access key बनाएं। इसके साथ, Claude EC2 instances manage कर सकता है, security groups configure कर सकता है, RDS databases सेट अप कर सकता है, S3 buckets manage कर सकता है और आपकी पूरी AWS infrastructure को programmatically handle कर सकता है। Claude से कहें: *"ये मेरे AWS credentials हैं। us-east-1 में एक t3.small EC2 instance launch करो और एक web app के लिए security groups configure करो।"*
- **Hetzner API Token** — अपनी Hetzner Cloud Console पर जाएं → project → Security → API Tokens → एक generate करें। Claude तब servers बना सकता है, firewalls configure कर सकता है, snapshots manage कर सकता है और आपकी पूरी Hetzner infrastructure handle कर सकता है। Claude से कहें: *"यह मेरा Hetzner API token है। Nuremberg में Ubuntu के साथ एक CX22 server बनाओ।"*

> API keys **पर सुरक्षा नोट।** ये API keys शक्तिशाली हैं — इन्हें passwords की तरह treat करें। **इन्हें कभी Git में commit न करें**, सार्वजनिक रूप से share न करें, और जिन chat interfaces पर भरोसा नहीं है उनमें paste न करें। इन्हें `.env` file में store करें या सीधे Claude को अपनी terminal session में pass करें। अगर कोई key compromise हो जाए, तो provider के dashboard से तुरंत revoke करें और नई generate करें। साथ ही, API tokens बनाते समय, **हमेशा least privilege principle का उपयोग करें**: केवल वही permissions दें जो वास्तव में ज़रूरी हैं, पूरा admin access नहीं।

इन API keys का Claude के साथ combination अविश्वसनीय रूप से शक्तिशाली है। आप literally कह सकते हैं: *"मेरे पास एक Next.js app है। इसे Cloudflare Pages पर deploy करो, custom domain सेट करो और DNS*

configure करो — ये मेरी API keys हैं।" और Claude सब कुछ automatically कर देगा। या: *"AWS पर एक EC2 instance बनाओ, Node.js और PM2 install करो, nginx configure करो, Cloudflare DNS सेट करो और मेरी app deploy करो।"* पूरा infrastructure setup मिनटों में, घंटों में नहीं।

GitHub Actions के साथ CI/CD

याद है जब हमने अध्याय 2 में Git और GitHub CLI सेट अप किया था? यहीं वह सब फायदा देता है। `gh` authenticated होने के बाद, आप Claude Code को कह सकते हैं कि **आपके प्रोजेक्ट को एक private GitHub repository में push करे**, **GitHub Actions** सेट अप करे, सभी ज़रूरी **secrets** (आपके सर्वर की SSH key, environment variables, आदि) configure करे, और एक automatic deployment pipeline बनाए — सब उसके terminal से, बिना आपके GitHub interface छूए।

आइडिया सरल है: एक बार GitHub Actions configure हो जाए, **हर बार जब Claude Code आपकी repository में कोड push करता है, यह automatically आपके सर्वर पर deploy हो जाता है।** कोई manual SSH नहीं, कोई files copy करना नहीं, कोई सर्वर पर खुद commands चलाना नहीं। Claude push करता है, GitHub Actions उसे pick करता है, SSH के माध्यम से आपके AWS या Hetzner सर्वर से कनेक्ट होता है, और सब कुछ deploy करता है। पूरी तरह automated।

बस Claude को बताएं: *"इस प्रोजेक्ट को GitHub पर एक नई private repository में push करो, एक GitHub Action सेट अप करो जो main पर हर push पर मेरे सर्वर पर deploy करे। यह रहा मेरा server IP और SSH key।"* Claude पहले से जानता है यह कैसे करना है — वह workflow file बनाएगा, SSH key और server IP को GitHub secrets के रूप में जोड़ेगा, और पूरी pipeline configure करेगा। इसीलिए हमने पहले GitHub CLI install किया था।

> **Dynamic IP की समस्या।** Static IPs आमतौर पर AWS और Hetzner दोनों पर extra cost होती हैं, तो आप पैसे बचाने के लिए **dynamic IP** उपयोग कर रहे होंगे। इसका मतलब है हर बार जब आप अपना सर्वर stop और restart करते हैं, IP बदल जाता है। जब ऐसा होता है, आपको **दो जगहों** पर update करना होगा: Cloudflare DNS record और `SERVER_IP` GitHub secret। Claude को बताएं कि server IP को GitHub Actions में `SERVER_IP` secret के रूप में store करे, ताकि जब बदले तो आपको केवल एक variable update करना हो। Claude को यह भी बताएं कि अगर deployment fail हो तो IP check और update करने की याद दिलाए — बदला हुआ IP लगभग हमेशा इसका कारण होता है।

5. मार्केटिंग और SEO रणनीतियाँ

> *आपका प्रोडक्ट लाइव है। अब दुनिया को जानना होगा कि यह exists करता है। इंडी प्रोडक्ट्स के लिए सबसे प्रभावी मार्केटिंग organic है — मुफ्त, creative, और सही तरीके से करने पर आश्चर्यजनक रूप से powerful। यह अध्याय उन exact tactics को कवर करता है जो हम उपयोग करते हैं।*

Organic Growth Strategies

आइए ईमानदार रहें: **सबसे अच्छी मार्केटिंग मार्केटिंग जैसी नहीं दिखती।** इंटरनेट ads से भरा है, और लोगों ने जो भी promotion जैसा लगता है उसे ignore करने का reflex विकसित कर लिया है। जो वास्तव में काम करता है वह है **stealth**

marketing — ऐसा content जो विज्ञापन के बजाय genuine information, एक सवाल, या एक recommendation जैसा दिखता है। लोग स्वाभाविक रूप से curious हैं और suggestions से मदद करना पसंद करते हैं। इसका उपयोग करें।

Reddit इसके लिए सबसे powerful platforms में से एक है। यह विशाल है, Google द्वारा highly indexed है, और niche communities से भरा है जहाँ आपकी target audience पहले से रहती है। लेकिन key यह है: **कभी भी aggressive marketing post न करें।** *"मेरी amazing नई साइट check करो!"* न लिखें — वह downvote, remove, या ban हो जाएगा। इसके बजाय, कुछ ऐसा लिखें:

- *"क्या कोई [well-known competitor] या [आपकी साइट] जैसी साइट्स recommend कर सकता है? Alternatives ढूंढ रहा हूँ।"*
- *"क्या किसी ने [आपकी product category] try की है? मुझे [आपकी साइट] मिली लेकिन अन्य options के बारे में curious हूँ।"*
- Relevant subreddits में सवालों का जवाब दें और naturally अपने product का mention करें जहाँ यह genuinely मदद करता है।

Reddit पर अधिकांश सफल indie marketing इसी तरह काम करती है। आप झूठ नहीं बोल रहे — आप अपने product को genuine conversation में एक discovery के रूप में frame कर रहे हैं। लोग click करते हैं क्योंकि वे curious हैं, इसलिए नहीं कि उन्हें sell किया जा रहा है। और यहाँ एक bonus है: **Reddit posts Google द्वारा index होते हैं**, तो एक अच्छी तरह से placed thread आपको महीनों या सालों तक organic search traffic ला सकता है।

आप एक छोटे budget के साथ **Reddit post sponsor** भी कर सकते हैं ताकि इसे temporarily boost किया जा सके। यह इसे search results में ऊपर push करता है, Google को तेज़ी से index करने देता है, और initial visibility देता है। अपने LLM से पूछें कि Reddit post promotion कैसे काम करता है और कितना budget समझदारी भरा है।

> **Funnel trick।** अपने homepage या landing page पर, कुछ ऐसा शामिल करें जो **आपके main product से स्वतंत्र रूप से लोगों को आकर्षित करे** — एक free tool, एक trending topic, उपयोगी जानकारी, या कुछ जो वर्तमान में लोकप्रिय है। यह एक **funnel** की तरह काम करता है: लोग free/interesting content के लिए आते हैं, आपका product discover करते हैं, और उनमें से एक percentage convert होता है। इसे ऐसा bait समझें जो genuine value प्रदान करता है जबकि जो आप बेच रहे हैं उसकी ओर ले जाता है।

SEO, Content और Distribution

कोई भी marketing शुरू करने से पहले, अपना **analytics और tracking** सेट करें। आपको दो चीज़ें चाहिए:

- **Google Analytics** — अपनी साइट पर tracking code जोड़ें (Claude से integrate करने के लिए कहें)। एक **mobile app** भी है ताकि आप कभी भी अपने phone से traffic check कर सकें। यह आपको कुल visits, user behavior, traffic sources, और conversion data दिखाता है।
- **Google Search Console** — इसे सेट करें और Google Analytics से connect करें। Search Console specifically आपका **organic Google traffic** track करता है: कौन से search queries लोगों को आपकी साइट पर लाते हैं, आप search results में कितनी बार appear होते हैं, और आपकी click-through rates। अपने LLM से पूछें कि दोनों tools कैसे setup और connect करें।

अगर आपके पास budget उपलब्ध है, तो **Google Ads** आपको एक initial boost दे सकता है। Ads briefly चलाने से Google के algorithm के साथ trust बनता है और जब तक आपका organic SEO बढ़ता है early traffic drive होता है। लेकिन costs जल्दी बढ़ते हैं, इसलिए इसे एक short-term accelerator मानें, long-term strategy नहीं। आपका LLM Google Ads setup और budgeting विस्तार से समझा सकता है।

SEO के लिए, basics से शुरू करें। अपने browser के **DevTools** (F12) खोलें, **Lighthouse** पर जाएं, और एक report generate करें। यह आपको Performance, Accessibility, Best Practices, और **SEO** के लिए scores देता है। जो बताया जाए वह fix करें — और हाँ, आप Claude Code से fixes handle करने के लिए कह सकते हैं।

लेने योग्य key SEO actions:

- **अपने pages में translations जोड़ें**। Multi-language support आपकी reach को नाटकीय रूप से बढ़ाता है। Claude Code से अपने project के लिए internationalization सेट अप करने के लिए कहें।
- **Proper meta tags जोड़ें** — title, description, social sharing के लिए Open Graph tags, structured data। ये सीधे प्रभावित करते हैं कि आपकी साइट Google search results में कैसी दिखती है।
- **Sitemap बनाएं और submit करें।** एक up-to-date sitemap generate करें और Google Search Console पर upload करें। यह Google को बताता है कि आपकी साइट पर ठीक कौन से pages हैं और उन्हें तेज़ी से index करने में मदद करता है।

> **SEO में धैर्य चाहिए।** रातों-रात organic traffic की उम्मीद न करें — Google को आपके pages को properly index और rank करने में हफ्ते या महीने लगते हैं। लेकिन जब यह kick करता है, तो यह **free traffic है जो आती रहती है** बिना एक पैसा खर्च किए। जो clicks आप Google Ads के माध्यम से सैकड़ों रुपये देकर पाते, वे organic search से free आएंगे। इस बीच, Reddit जैसे social platforms पर initial visibility और backlinks बनाने का काम करें। SEO एक long game है, लेकिन यह सबसे मूल्यवान marketing investment है जो आप कर सकते हैं।

उभरता trend: AI-driven traffic

Traffic का एक नया और तेज़ी से बढ़ता source है जिस पर अधिकांश लोग अभी ध्यान नहीं दे रहे: **LLMs आपकी साइट recommend कर रहे हैं।** ChatGPT, Gemini, Claude, और अन्य AI assistants बढ़ते हुए search engines के रूप में उपयोग किए जा रहे हैं। जब कोई पूछता है *"[आपकी product category] के लिए अच्छी साइट क्या है?"*, ये models specific websites recommend कर सकते हैं और करते हैं — और इससे real traffic आता है। हमारी अपनी visits का एक significant portion ChatGPT और अन्य LLMs से आता है।

इसका फायदा उठाने के लिए, अपने **Cloudflare dashboard** पर जाएं और सुनिश्चित करें कि आप **AI crawlers को block करने वाला option disable करें**। कई साइटें default से AI bots को block करती हैं, जो LLMs को आपके content के बारे में सीखने से रोकता है। AI crawlers को अपनी साइट access करने देकर, आप इन models को अपने pages index करने, समझने कि आप क्या offer करते हैं, और भविष्य में users को potentially recommend करने दे रहे हैं। यह अनिवार्य रूप से **AI era के लिए free SEO** है — और boost enormous हो सकता है।

> **Google से आगे सोचें।** Traditional SEO Google search को target करता है। लेकिन AI-driven recommendations एक major traffic source बन रही हैं — और यह trend केवल accelerate हो रहा है। सुनिश्चित करें कि आपकी साइट AI crawlers के लिए accessible है, clear और descriptive content है, और अच्छी तरह structured है ताकि LLMs आसानी से समझ सकें कि आप क्या offer करते हैं। जो साइटें अभी AI discovery के लिए खुद को position करती हैं उन्हें massive advantage होगा जैसे-जैसे यह channel बढ़ता है।

धन्यवाद!

इस कोर्स को खरीदने और अपना समय और पैसा हम पर भरोसा करने के लिए धन्यवाद। हम सच में आशा करते हैं कि आपको यह मूल्यवान लगा और यह आपको कुछ real बनाने में मदद करेगा।

हम आपसे सुनना चाहेंगे। **हमारी Discord community में शामिल हों** apifreellm.com पर — यही वह जगह है जहाँ हम रहते हैं, updates share करते हैं, और एक-दूसरे की मदद करते हैं।

अगर आपके पास एक पल है, तो **हमें एक review लिखें** और बताएं आप क्या सोचते हैं। आपको सबसे उपयोगी क्या लगा? क्या missing था? क्या बेहतर हो सकता था? हम सच में आपकी feedback में interested हैं — यह कोर्स एक living project है और हम इसे **आपकी** actual ज़रूरतों के आधार पर सुधारते रहना चाहते हैं।

Discord पर मिलते हैं। अब जाइए और कुछ amazing बनाइए।