# Yapay Zeka Destekli Gelistirme

## FİKİRDEN URETİME

Kendi web sitenizi, uygulamanizi veya girisiminizi sifirdan olusturmak icin eksiksiz ve kapsamli rehber. Yapay zeka ajanlarini ve araclarini ogrenin, odeme entegrasyonu yapin, sunucunuzu yayin hayatina gecirin ve organik buyume icin pazarlama taktiklerini kesfedin.

apifreellm.com

Surum 1.0 — Subat 2026

# Icindekiler

# 1. Giris

> *Su anda bunu okuyorsunuz cunku bir yerlerde, bir sekilde, pazarlama stratejileri, SEO teknikleri ve akilli konumlandirmanin bir kombinasyonu bu kursu ekraniniza getirdi. Bu bile size bir sey anlatmali: bu rehberdeki taktikler gercekten ise yariyor. Ve evet, her birini ogreneceksiniz.*

## Dunya Degisti

Yazilim gelistirme artik eskisi gibi degil. Birkac yil once bir web uygulamasi olusturmak aylar suren calisma, bir gelistirici ekibi ve ciddi bir butce gerektiriyordu. Bugun, dogru araclara ve dogru bilgiye sahip tek bir kisi, uretim ortamina hazir bir uygulamayi aylar degil, gunler icinde olusturup yayinlayabiliyor.

Bu kurs, sektorde calisan ve gercek urunler olusturmak icin yapay zeka ajanlik araclarini her gun kullanan profesyoneller tarafindan yazilmistir. Ders kitabindan teori ogretmiyoruz. Su anda gercek dunyada neyin ise yaradigini ogretiyoruz.

## Bu Kurs Neden Var

Yapay zeka ve LLM'ler artik insanlardan daha iyi ve daha hizli uygulayicilar. Kod yazabilir, hata ayiklayabilir, yeniden duzenleyebilir ve hicbir insanin yetisemeyecegi bir hizda dagitabilirler. Ancak temelden yoksun olduklari bir sey var: **fikirler**.

> Yapay zeka modelleri olaganustu uygulayicilar, ancak yenilikci degiller. Pazardaki bir boslugu gorup "Bunu cozmek icin bir sey yapabilirim" diye dusunmezler. Bu sizin isiniz. Sizin rolunuz fikirlerin mimari olmaktir. Yapay zeka sizin insaatcinizdir.

Bir LLM'ye zayif talimatlar verdiginizde, yine de bir seyler uretir. Durup neyin daha iyi olacagini aciklamaz. Olusturduklerinizin kalitesi, dogrudan bilginizin kalitesiyle orantilidir. Bu kurs o boslugu kapatir.

Bu sadece bir gelistirme kursu degil. Bu, bir fikirden canli, gelir getiren bir urune giden yolun eksiksiz planidir. Son derece etkili pazarlama ve SEO taktikleri dahil — sizi tam da bu sayfaya getiren tekniklerin aynilarini.

# 2. Yapay Zeka Oncelikli Gelistirme Yigini

> *Sectiginiz araclar ne kadar hizli ilerlediginizi belirler. Bu bolumde, bugun mevcut olan yapay zeka kodlama ajanlarini inceliyoruz, dogru olani secmenize yardimci oluyoruz ve amatorleri profesyonellerden ayiran yonlendirme stratejilerini ogretiyoruz.*

> **Not:** Bu rehber Windows kullanilarak yazilmistir, ancak tum araclar ve adimlar macOS ve Linux'ta birebir ayni sekilde calisir. Google Antigravity, Claude Code ve bu kursta tartisilan diger tum uygulamalar tamamen platformlar arasi calisir. Mac veya Linux kullaniyorsaniz, ayni adimlari takip edin — arayuzler ve is akislari aynidir.

## Yapay Zeka Ajan Ekosistemi

Yapay zeka kodlama araclari alani patlama yasadi. Ancak tum araclar esit yaratilmamistir. Bazilari sadece gelismis otomatik tamamlama motorlaridir. Digerleri ise tum kod tabaninizi okuyabilen, bir strateji planlayabilen, birden fazla dosyada degisiklik uygulayabilen, testler calistirip hatalari kendi baslarina duzeltebilen tam otonom ajanlardir. Bu farki anlamak cok onemlidir.

Yapay zeka kodlama araclarinin iki temel kategorisi vardir:

- **Satirici asistanlar** — Bunlar editorunuzun icinde otururlar ve siz yazarken kod onerirler. Orijinal GitHub Copilot'u dusunun. Tepkiseldirler: yazmanizi beklerler, sonra sirada ne gelecegini tahmin etmeye calisirlar. Kullanisli ama sinirli.
- **Ajanlik araclar** — Bunlar tamamen farkli bir canavar. Onlara dogal dilde bir gorev verirsiniz ve otonom olarak planlar, yazar, duzenler, hata ayiklar ve iterasyon yaparlar. Sadece bir satir kod onermezler. Ozellikler insa ederler. Gercek guc buradadir.

Su anda onemli olan araclar sunlardir:

### Claude Code (Anthropic)

Claude Code, deneyimlerimize gore, bugun mevcut olan en guclu yapay zeka kodlama ajanidir. Dogrudan proje dizininizde calisan terminal tabanli bir ajandir. Dogal dilde talimat verirsiniz ve dosyalarinizi okur, kod yazar, komutlar calistirir, commit'ler olusturur ve hatalari otonom olarak duzeltir. Dosya sisteminize ve kabugunuza tam erisime sahiptir, bu da onu gercek gelistirme isi icin inanilmaz derecede etkili kilar. npm ile kurabilirsiniz ( `npm install -g @anthropic-ai/claude-code` ) veya kisaca bahsedecegimiz bir VS Code eklentisi olarak kullanabilirsiniz.

### Google Antigravity

Antigravity, Google'in yapay zeka destekli sohbet ve ajan yeteneklerini dogrudan is akisiniza getiren bir gelistirme ortamidir. Yapay zeka ajanlariyla sohbet arayuzleri uzerinden etkilesime girebileceginiz akilli bir calisma alani olarak dusunun ve her ajan konusmasindan entegre bir VS Code editoru

acabilirsiniz. Bu cok onemli: Antigravity size konusmali yapay zeka katmanini sunarken, VS Code kod duzenleme gucunu saglar. Kombinasyon kusursuz — ajanla ne insa edeceginizi tartisirsiniz, sonra pencere degistirmeden dogrudan koda atlarsiniz.

### Cursor

Cursor, yapay zekanin derinlemesine entegre edildigi bir VS Code fork'udur. Hem satirici oneriler hem de birden fazla dosyada degisiklikleri tanimlayabileceginiz ajanlik bir "Composer" modu vardir. Zaten VS Code kullaniyorsaniz arayuz tanidiktir ve birden fazla modeli destekler (Claude, GPT, vb.). Ozellikle tamamen gorsel bir is akisi tercih ediyorsaniz saglam bir aractir. Ancak ajanlik yetenekleri, iyi olmakla birlikte, Claude Code kadar otonom degildir. Genellikle bireysel degisiklikleri adim adim incelemeniz ve onaylamaniz gerekir.

> Onerimiz: **Google Antigravity + VS Code eklentisi olarak Claude Code.** Isinizi planlamak ve tartismak icin Antigravity'nin ajan sohbetlerini kullanin, sonra entegre VS Code'u acin ve agir isleri Claude Code'a birakin. Bu size her iki dunyanin en iyisini sunar: planlama icin akilli konusma ve insa icin otonom kod yurutme.

## Ajaninizi Secme ve Yapilandirma

Bir ajanlik arac kurmak ve calistirmak kolay kisimdir. Ondan en iyi sekilde yararlanmak, nasil calistigini anlamayi, dogru aboneligi secmeyi ve dogru yapilandirmayi gerektirir.

### Claude aboneligi: Pro ile baslayin

Claude Code bir Anthropic hesabi gerektirir. **Claude Pro aboneligi** ile baslamanizi oneririz, bu giris seviyesi ucretli katmandir. Uygun fiyatli ve tum ozellikleriyle Claude Code'a erisim saglar. Deney yapmak, is akisini ogrenmek ve ilk basit projenizi olusturmak icin mukemmel bir baslangic noktasidir.

Ancak su anda, Pro planinin **sinirli kullanimi** oldugunu bilin. Claude Code'u yogun kullanirseniz, kullanim sinirina nispeten hizli ulasirsiniz. Ogrenme ve kucuk projeler icin fazlasiyla yeterlidir. Ancak Claude Code'u birincil gelistirme araciniz olarak kullanmayi ve her gun saatlerce uzerinde calismayi planliyorsaniz, basindan itibaren ust katman planlardan birine (ornegin Max) gecmeyi dusunun. Ust planlar onemli olcude daha fazla kullanim sunar, bu da daha buyuk projeler olustururken daha az kesinti ve daha akici bir is akisi anlamina gelir.

### Model: Claude Opus 4.6

Su anda birincil gelistirme modeliniz olarak **Claude Opus 4.6** kullanmanizi oneririz. Su anda web siteleri ve uygulamalar olusturmak icin en iyi modeldir. Karmasik mimarileri anlar, temiz ve uretim ortamina hazir kod yazar, coklu dosya degisikliklerini dikkat cekici bir dogrulukla yonetir ve ilk denemede nadiren duzeltme gerektirir. Claude Code ile calisirken, Opus 4.6'yi varsayilan modeliniz olarak ayarlayin. Daha kucuk modellerle karsilastirildiginda cikti kalitesindeki fark hemen fark edilir.

### Antigravity'yi ayarlama

Bu noktadan itibaren, bu rehber birincil gelistirme ortamimiz olarak **Google Antigravity** kullanarak devam ediyor. Her seyi nasil hazirlayacaginiz asagidadir.

**Adim 1: Proje klasorunuzu olusturun.** Antigravity'yi acmadan once, bilgisayarinizda ilk projenizin yasayacagi bir klasor olusturun. Ornegin, Windows'ta `C:\Projects\my-first-app` veya Mac/Linux'ta `~/Projects/my-first-app` olusturabilirsiniz. Bu, Antigravity'nin calisma alani olarak acacagi klasordur. Simdilik bos olabilir — kursta daha sonra kodla dolduracagiz.

**Adim 2: Antigravity'yi kurun ve baslatin.** Google Antigravity'yi indirin, kurun ve acin. Istendiginde Google hesabinizla giris yapin.

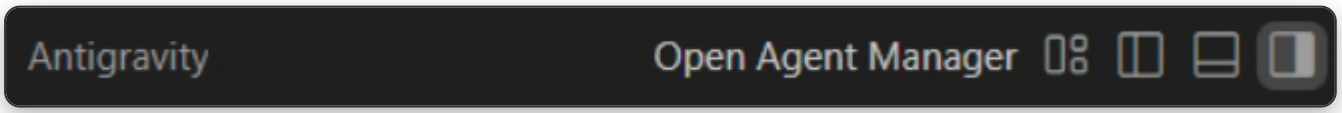
Kurulum surecinde, Antigravity sizden birkac politika yapilandirmanizi isteyecektir. Hizli, otonom bir gelistirme is akisi icin **ozel yapilandirma** secmenizi ve bu secenekleri ayarlamanizi oneririz:

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

Bu ayarlarla, Antigravity'nin ajanlari her adimda onay istemeden komutlari calistirabilir, dosyalar yazabilir ve kod calistirabilir. Bu kursta hedefledigimiz hizli, akici gelistirme deneyimini saglayan budur.

> **Guvenlik uyarisi:** Ajanlarin otonom olarak devam etmesine izin verdiginizde, hem Antigravity hem de Claude Code sisteminizde manuel onay olmadan islemler yurutebilir. Bu, onlara neyi actiginiza dikkat etmeniz gerektigini gosterir. Ajanlari zararli yazilim icerebilecek kod tabanlarina yonlendirmeyin ve guvenilmeyen kaynaklardan gelen baglanti veya kaynaklara karsi dikkatli olun. Yapay zeka caginda yeni saldiri vektorleri var — kotuniyetli kisiler, LLM'leri zararli komutlar calistirmaya kandirmak icin ozel olarak tasarlanmis icerik olusturabilir. Ajanlarinizin ne yaptigini her zaman gozlemleyin. Hiz icin "Always Proceed" ayarini oneririz, ancak dikkatli olun ve ciktiyi duzenli olarak inceleyin.

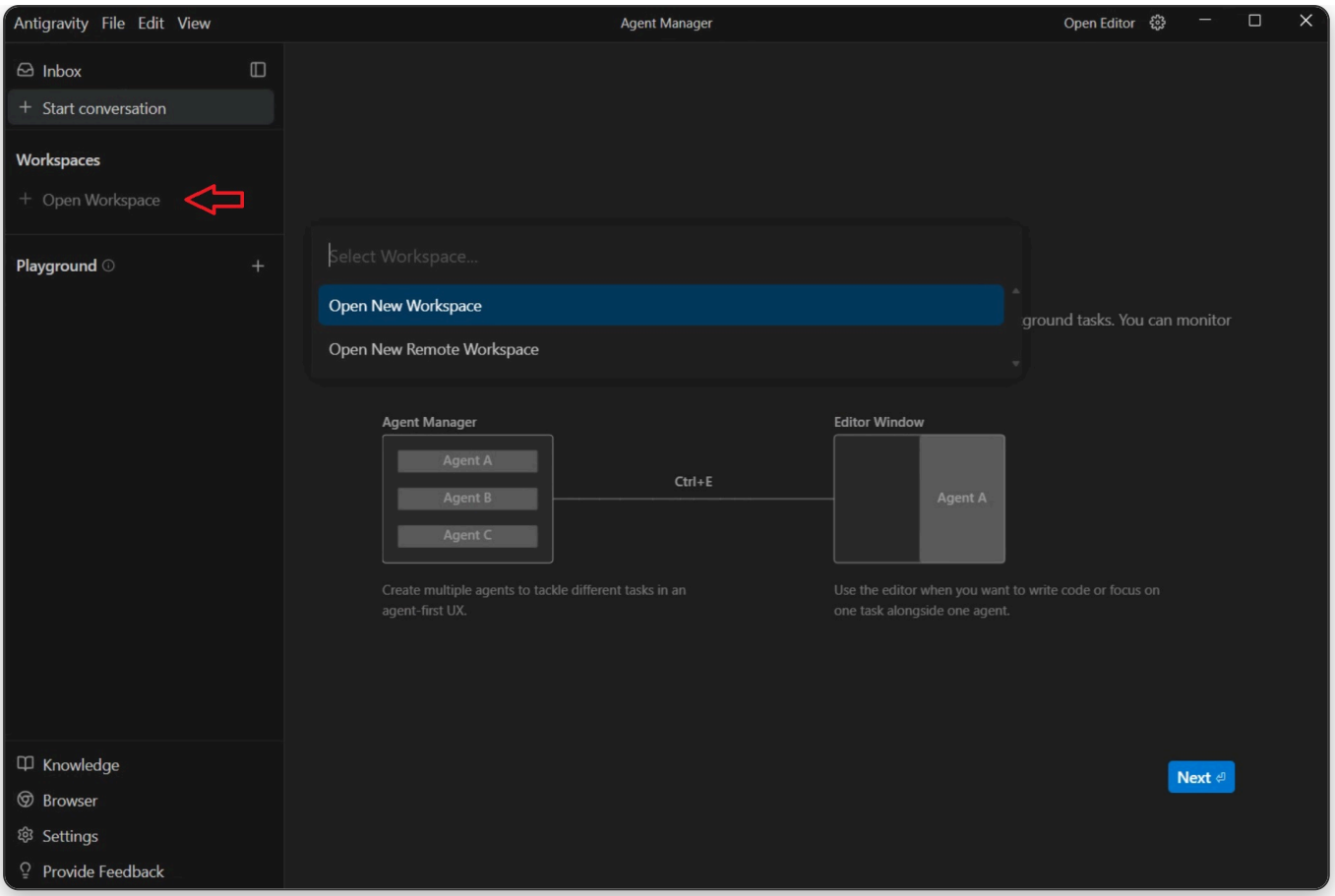
**Adim 3: Agent Manager'i acin.** Antigravity calistiktan sonra, pencerenin ust cubugundaki **"Open Agent Manager"** dugmesine tiklayin.




*Antigravity'nin ust cubugundaki "Open Agent Manager" dugmesi.*

**Agent Manager**, Antigravity'nin merkezi merkezidir. Projelerinizi yonettiginiz, yapay zeka konusmalari baslattiginiz ve gelistirme calisma alanlarinizi actiginiz yer burasidur.

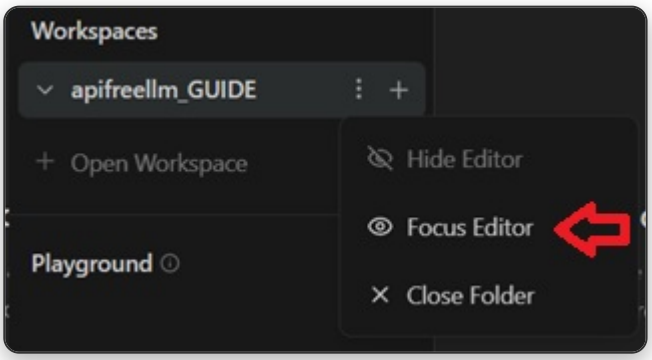
**Adim 4: Ilk calisma alaninizi olusturun.** Agent Manager'da sol kenar cubuguna bakin. **"Workspaces"** bolumunun altinda, **"+ Open Workspace"** dugmesine tiklayin. Bir acilir menu gorunecektir — **"Open New Workspace"** secenegini secin.




*Sol kenar cubugunda "+ Open Workspace" dugmesine tiklayin, sonra "Open New Workspace" secin.*

Antigravity sizden bir klasor secmenizi isteyecektir. Adim 1'de olusturdugunuz proje klasorune gidin ve onu secin. Yeni calisma alaniniz, sol kenar cubugunda "Workspaces" altinda, sectiginiz klasorun adiyla gorunecektir. Her calisma alanini bireysel bir gelistirme oturumu olarak dusunun — proje basina bir tane. Ihtiyaciniz kadar calisma alani olusturabilir ve istediginiz zaman Agent Manager'dan aralarinda gecis yapabilirsiniz.

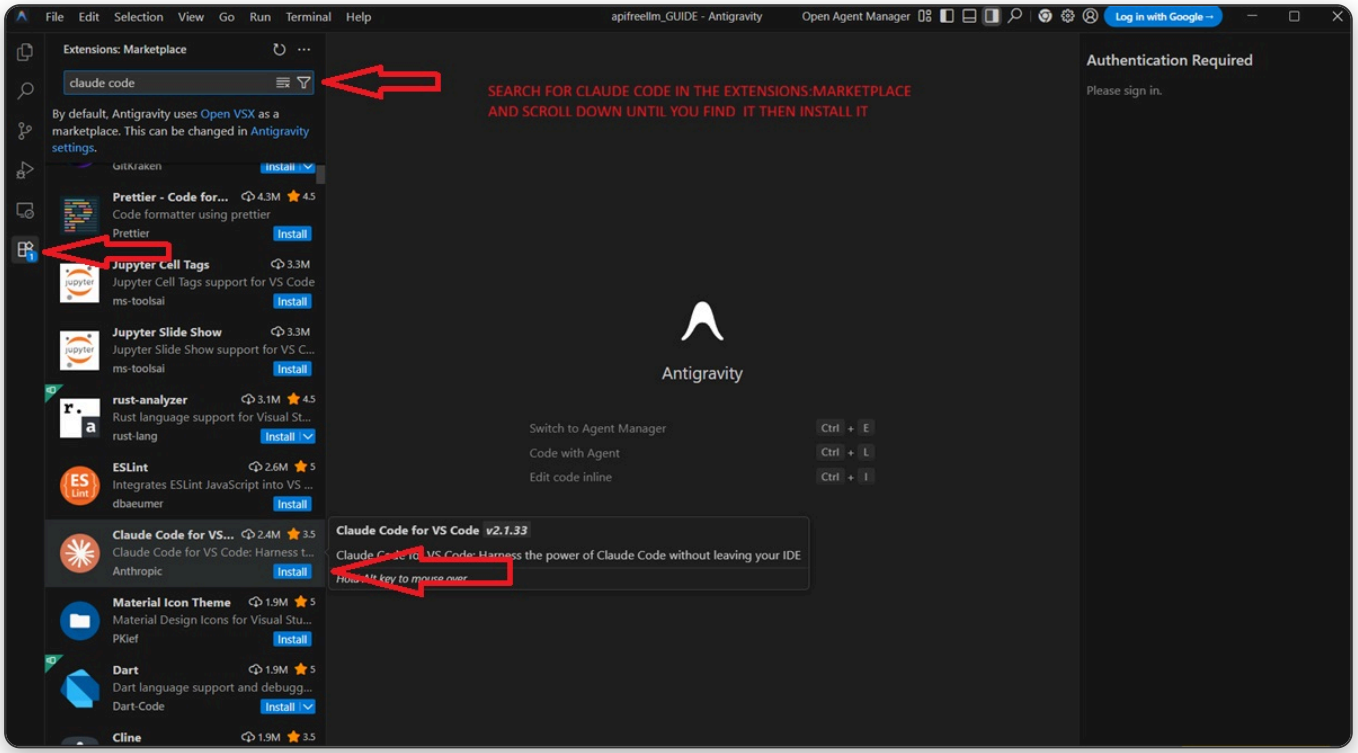
**Adim 5: Editoru acin.** Calisma alaniniz olusturulduktan sonra, sol kenar cubugunda adini goreceksiniz. Calisma alani adinin yanindaki **uc dikey noktaya** (:) tiklayin, sonra **"Focus Editor"** secin. Bu, kodunuzu yazacaginiz ve duzenleyeceginiz o calisma alani icin tam VS Code ortamini acar.




*Calisma alani adinin yanindaki uc noktaya tiklayin ve "Focus Editor" secin.*

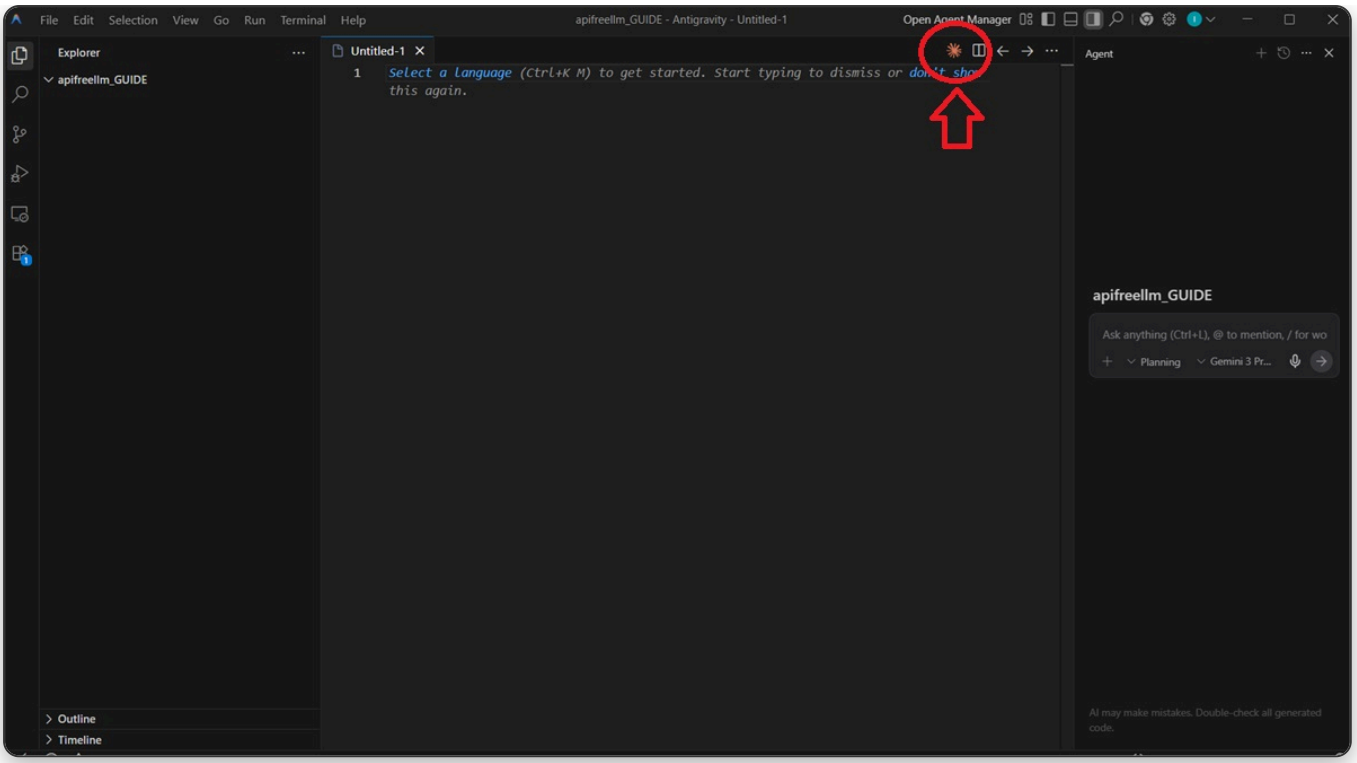
### Claude Code'u VS Code eklentisi olarak kurma

Artik editor acik olduguna gore, Claude Code'u kurmanin zamani geldi. Editor VS Code tabanli oldugu icin, eklenti marketine erisiniz vardir. Sol kenar cubugundaki **Extensions simgesine** tiklayin (veya `Ctrl+Shift+X` basin). Arama cubuguna **"claude code"** yazin. Sonuclar arasinda asagi kaymaniz gerekebilir — Anthropic'in **"Claude Code for VS Code"** eklentisini arayin. Bulduginuzda, **Install** dugmesine tiklayin.



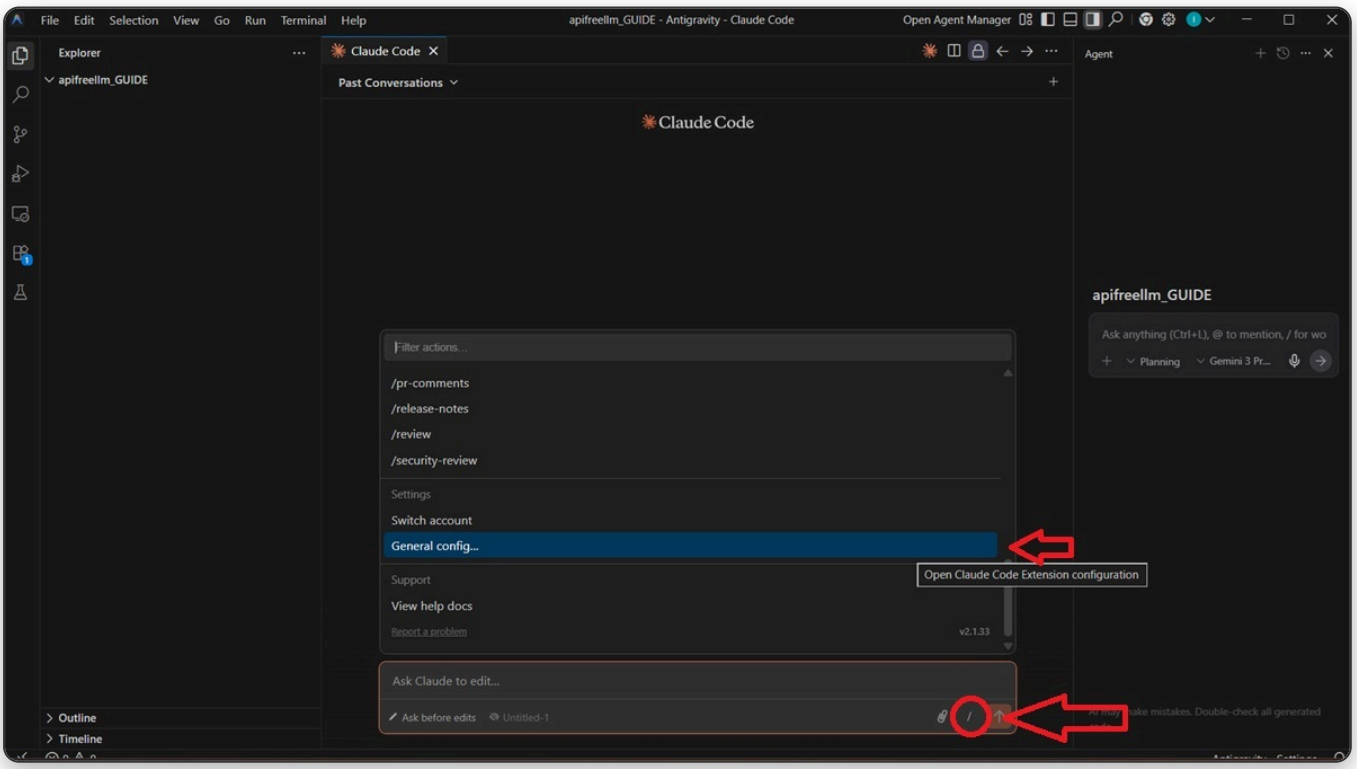

*Eklenti marketinde "claude code" arayin, bulmak icin asagi kaydin ve Install'a tiklayin.*

Kurulduktan sonra, Extensions panelini kapatin ve yeni bir dosya (veya projenizdeki herhangi bir dosya) acin. Editor'un sag ust alaninda kucuk bir **Claude Code simgesinin** gorundugunuzu fark edeceksiniz — kucuk turuncu bir sembole benzer. Claude Code sohbet panelini acmak icin uzerine tiklayin.



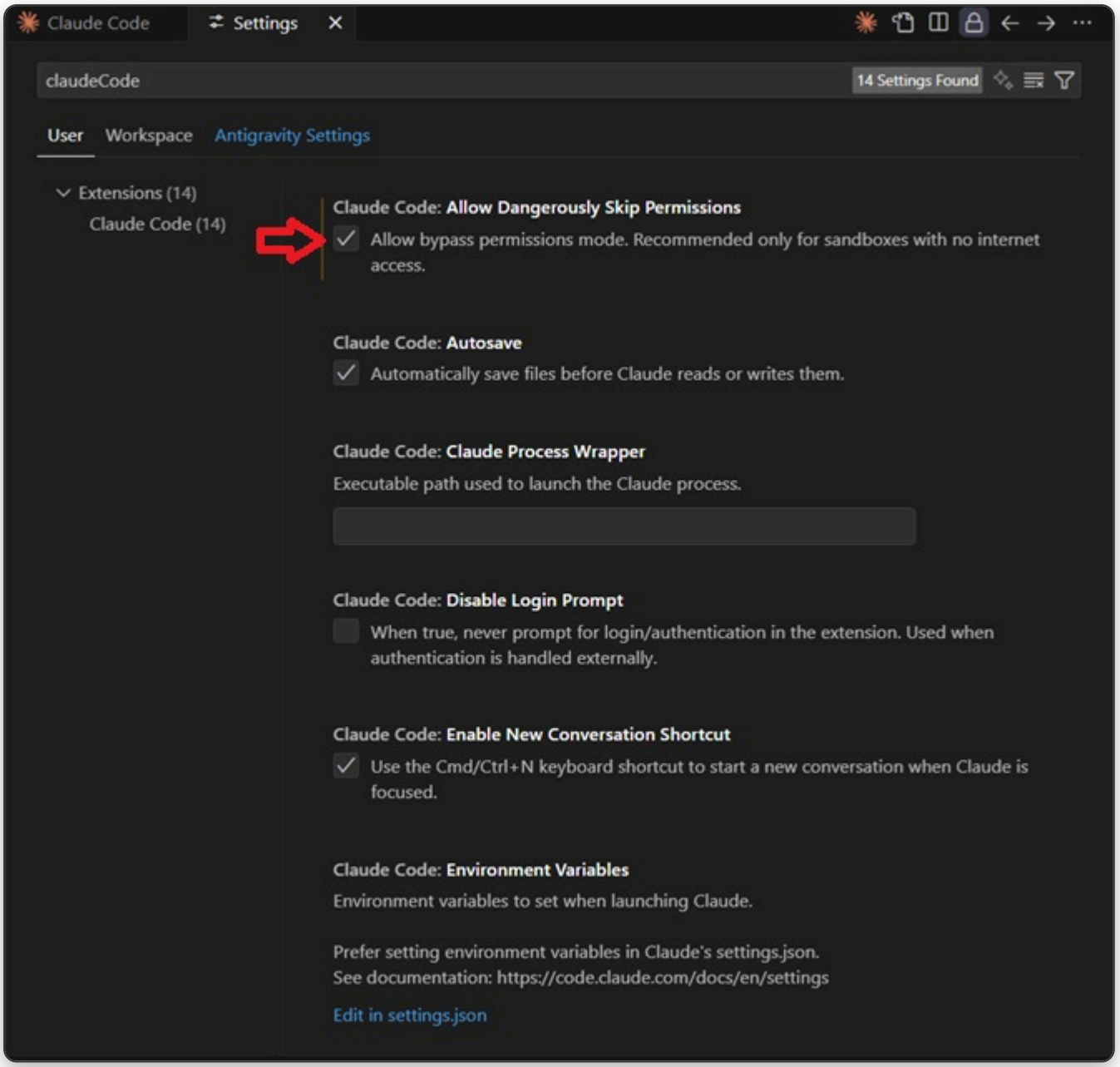

*Claude Code dugmesi (daire icinde) editorun sag ustunde gorunur. Tiklayin ve Claude Code sohbetini acin.*

Claude Code, Anthropic hesabinizla (Pro aboneliginiz olan hesap) giris yapmanizi isteyecektir. Giris yaptiktan sonra yapilandirmaniz gerekiyor. Claude Code sohbet panelinde, panelin altindaki **/** dugmesine tiklayin. Cesitli komutlari iceren bir menu gorunecektir. **Settings** bolumune kaydin ve Claude Code eklenti yapilandirmasini acmak icin "**General config...**" secenegine tiklayin.




*"/" dugmesine tiklayin, sonra Settings'e kaydin ve yapilandirmayi acmak icin "General config..." secin.*

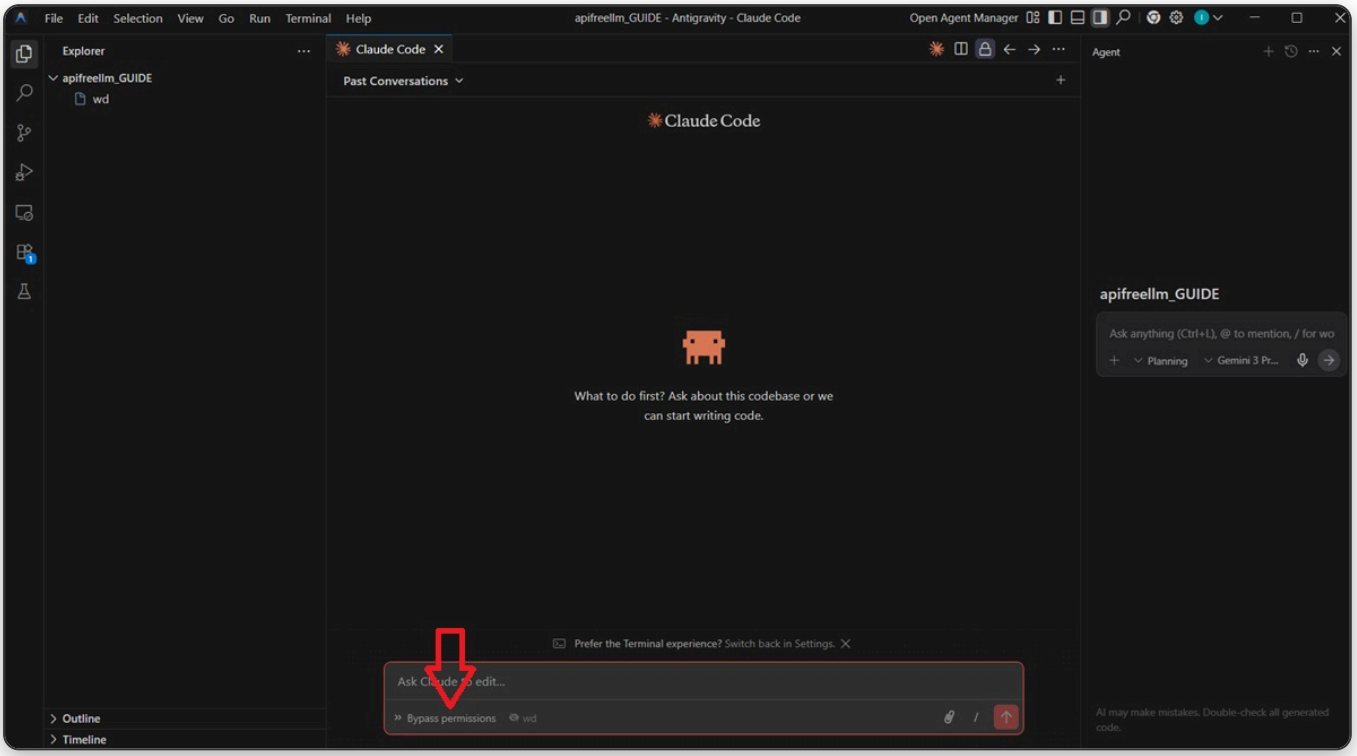
Acilan yapilandirma panelinde, **"Allow Dangerously Skip Permissions"** adli secenegi arayin. Bu degistiriciyi etkinlestirin. Etkinlestirildikten sonra, izin modunuz olarak **"Bypass permissions"** secebileceksiniz; bu, Claude Code'un tamamen otonom olarak calismasini saglar — dosyalari okuma, kod yazma, terminal komutlari calistirma ve her adimda onay istemeden degisiklik yapma.




*Claude Code ayarlarinda "Allow Dangerously Skip Permissions" degistiricisinietkinlestirin, sonra "Bypass permissions" secin.*

> **Onemli:** Bypass permissions etkinken, Claude Code sisteminizde manuel onay olmadan islemler yurutecektir. Bu, akici bir gelistirme is akisi icin zorunludur, ancak sorumluluk gerektirir. Calistirmasini istediginiz kod hakkinda dikkatli olun ve asla guvenilmeyen depolara veya suphe uyandiran URL'lere yonlendirmeyin. Yapay zeka ajanlari, prompt enjeksiyonu yoluyla istismar edilebilir — dosyalarda veya web sitelerinde gizlenen ve ajani zararli komutlar calistirmaya kandiran kotuniyetli icerik. Claude Code'un ne yaptigini her zaman inceleyin, ozellikle harici kaynaklarla calisirken.

Ayrica, yeni sohbetler icin **"Bypass permissions"in varsayilan secim olmasi** ayarini etkinlestirmenizi oneririz, boylece her yeni konusma baslattiginizda manuel olarak secmeniz gerekmez. Simdi Claude Code'u kapatin ve yeniden acin (Claude Code simge dugmesine tekrar tiklayarak). Bundan sonra, Claude Code sohbet panelinin altindaki izin secici dugmesinde **"Bypass permissions"** secenegini goreceksiniz. Etkinlestirmek icin uzerine tiklayin.




*Ekran goruntusunde gosterilen dugmeden "Bypass permissions" secin.*

Bypass permissions aktifken, Claude Code komutlari calistirir, dosyalar olusturur, kodu duzenler ve terminal islemlerini tamamen kendi basina yapar — her adimda onayinizi istemeden. Bu, **gelistirme is akisinin %100'unu otomatiklestirmenizi** saglayan seydir: ne olusturmak istediginizi anlatirsiniz ve Claude Code bastan sona otonom olarak insa eder. Artik her dosya degisikligi veya komut yurutmesinde "Kabul Et"e tiklamak yok. Siz talimatlari verirsiniz ve ajan gerisini halleder.

## Kullaniminizi akillica yonetme

Basindan bilmeniz gereken bir sey: Claude Code ile her etkilesim **token** tuketir ve aboneliginizin bir kullanim siniri vardir. Iyi haber su ki, ne kadar kullandiginizi tam olarak izleyebilirsiniz. Claude Code sohbet panelinde, **/** dugmesine tiklayin — mevcut komutlar arasinda **mevcut kullaniminizi** bulacaksiniz; ne kadar token tukettiginizi ve ne kadar kapasitenizin kaldigini gosterir.

Tum modeller ayni oranda token tuketmez. **Claude Opus 4.6** en akilli ve yetenekli modeldir, ancak etkilesim basina en fazla token tuketen de odur. **Sonnet** veya **Haiku** gibi daha kucuk modeller daha az gucludur ancak daha yuksek kullanim sinirlarina sahiptir ve onemli olcude daha az token tuketir. Onerimiz: derin akil yurutme, coklu dosya degisiklikleri veya mimari kararlar gerektiren **karmasik gorevler icin Opus** kullanin — zekasinin gercek bir fark yarattigi yer burasidur. Hizli duzeltmeler, kucuk duzenlemeler veya basit sorular gibi daha basit gorevler icin, Opus tokenlerinizi gercekten onemli oldugu zamanlar icin korumak amaciyla daha hafif bir modele gecin.

Claude tokenlerinizi korumak icin baska bir strateji daha var: Claude'un zeka seviyesini gerektirmeyen sorular icin **Antigravity'nin yerlesik ajanini kullanin**. Antigravity birden fazla modeli

destekler, ancak bu hizli sorular icin **Gemini** kullanmanizi oneririz — Antigravity bir Google urunu oldugu icin, Gemini en yuksek kullanim sinirina sahiptir ve platformda mevcut en hizli modeldir. Hizli bir CSS hatirlatmasina mi ihtiyaciniz var? Bir Git komutunun sozdizimini mi ogrenmek istiyorsunuz? Bir kutuphanenin nasil calistigini mi merak ediyorsunuz? Claude Code yerine Antigravity'ye sorun. Bu sekilde, Claude tokenlerinizi en buyuk etkiyi yarattigi agir gelistirme islerine saklarsiniz.

Onceki ekran goruntusunde gordugunuz gibi, **Claude Code sohbetini solda** ve **Antigravity'nin ajan sohbetini sagda** tutmanizi oneririz. Bu yan yana duzenleme, her iki ajana da her zaman aninda erisim saglar.

> Akilli is akisi: **Insa icin Opus, hizli gorevler icin daha hafif modeller, genel sorular icin Antigravity.** Bu sekilde Claude aboneliginizin degerini en ust duzeye cikarirsiniz. Claude kullanim siniriniz azaliyorsa, daha basit sorular icin her zaman yaninizdaki Antigravity'nin Gemini ajanini kullanabileceginizi unutmayin — Claude tokenlerinizi gercekten ihtiyac duyan gelistirme gorevleri icin saklayin.

### Temel yapilandirma

Claude Code'u nasil kurarsaniz kurun, bu yapilandirma adimlari sonuclarinizi onemli olcude iyilestirecektir:

- **CLAUDE.md dosyasi kullanin** — Proje kokunuze bir `CLAUDE.md` dosyasi yerlestirin. Bu dosya, her oturumun basinda Claude Code tarafindan otomatik olarak okunur. Proje yapisini, kodlama kurallarini, teknoloji yiginini ve ajanin uyulmasi gereken kurallari tanimlamak icin kullanin. Yapay zeka gelistiriciniz icin bir oryantasyon belgesi olarak dusunun.
- **Projenizi duzenli tutun** — Yapay zeka ajanlari temiz, iyi yapilandirilmis kod tabanlariyla cok daha iyi calisir. Kodunuz dagilmissa, ajan dagitik cikti uretecektir. Iyi klasor yapisi, acik adlandirma kurallari ve tutarli desenler buyuk fark yaratir.
- **Surum kontrolu kullanin** — Her zaman Git baslatilmis olarak calisin. Bu size bir guvenlik agi saglar. Ajan bir hata yaparsa, aninda geri donebilirsiniz. Ayrica ajanin sizin icin commit'ler olusturmasina olanak tanir, bu da neyin degistigini ve neden degistigini takip etmek icin sasirtici derecede faydalidir.

## Temel Araclar: Git & GitHub CLI

Herhangi bir sey olusturmaya baslamadan once, sisteminize kurulmasi gereken iki arac var: **Git** ve **GitHub CLI (gh)**. Bunlar her modern gelistirme is akisi icin temeldir ve yapay zeka ajanlarinizin kod depolarinizi otonom olarak yonetmesini saglayan seydir.

### Git ve GitHub CLI neden onemli

**Git**, projenizdeki her degisikligi izleyen surum kontrol sistemidir. Guvenlik aginizdir: Claude Code bir hata yaparsa, aninda geri donebilirsiniz. Ayrica ajanin commit'ler olusturmasina, dallari yonetmesine ve projenizin evriminin temiz bir gecmisini tutmasina olanak tanir — hepsi otomatik olarak.

**GitHub CLI (gh)**, terminalden dogrudan GitHub'a erisim saglayan bir komut satiri aracidir. Tam otomasyonun anahtari budur: `gh` kurulup dogrulandiginda, Claude Code depolar olusturabilir, kod gonderebilir, pull request'leri yonetebilir, depo ayarlarini yapilandirabilir, dagitim icin GitHub Actions kurabilir, gizli anahtarlar ekleyebilir ve cok daha fazlasini yapabilir — hepsi terminalinden, tarayicida GitHub'i acmaniza gerek kalmadan.

### Git ve GitHub CLI kurulumu

Bu araclari kurmanin en basit yolu **Claude Code veya Antigravity'den bunu yapmasini istemektir**. Ajaniniza sunu soyleyin: *"Sistemime Git ve GitHub CLI kur."* Ajan isletim sisteminizi algilayacak ve uygun kurulum komutlarini calistiracaktir. Windows'ta genellikle `winget` kullanir veya yukleme programlarini indirir; macOS'ta `brew` kullanir; Linux'ta ise `apt` veya sisteminizin paket yoneticisini kullanir.

Manuel olarak kurmak istiyorsaniz, Git'i resmi web sitesinden ve GitHub CLI'yi GitHub surum sayfasindan indirebilirsiniz. Ancak yapay zekanin halletmesine izin vermek daha hizlidir ve yaygin kurulum hatalarindan kacinir.

### GitHub CLI kimlik dogrulamasi

Kurulumdan sonra, `gh`'nin GitHub hesabiniza erisebilmesi icin giris yapmaniz gerekir. Yine, ajaniniza sormaniz yeterli: *"GitHub CLI'ya giris yap."* Ajan `gh auth login` komutunu calistiracak ve sizi kimlik dogrulama akisinda yonlendirecektir; bu genellikle bir tarayici baglantisi acmayi ve bir kod girmeyi icerir. Dogrulandiktan sonra, ajan GitHub depolariniza tam erisime sahip olur.

> **Bu oyun degistirici.** `gh` dogrulandiginda, Claude Code'a sunu soyleyebilirsiniz: *"my-app adinda yeni bir ozel depo olustur, projeyi baslat ve kodu gonder."* Veya daha sonra: *"Her main'e push'ta sunucuma dagitim yapan bir GitHub Action kur."* Ajan her seyi halleder — dosya olusturma, gizli anahtarlari yapilandirma, is akislarini kurma — siz editorunuzden hic ayrilmadan. Gercek gelistirme otomasyonu boyle gorunur.

# 3. Ilk Projeniz: Sifirdan Yayina

> *Ortaminiz hazir. Claude Code acik, Antigravity calisiyor, Git ve GitHub CLI yapilandirilmis durumda. Gercek bir sey insa etmenin zamani geldi. Bu noktadan itibaren, kurs kurulumdan stratejiye geciyor — yapay zeka ajanlariyla insa ederken size gercek bir avantaj saglayacak pratik teknikler ve icgoruler.*

## Asla Sifirdan Baslamayin

Bu, tum kursun en onemli tavsiyesidir: **asla sifirdan insa etmeyin.**

Claude Opus inanilmaz derecede gelismis ve akilli bir model olsa da, hata yapacaktir. Her yapay zeka modeli yapar. Proje yapinizi yanlis anlayabilir, guncel olmayan bir kutuphane kullanabilir, tutarsiz bir dosya duzeni olusturabilir veya proje buyudukce birbirine tam uymayan kod uretebilir. Bos bir klasorden baslamak, yapay zekanin hicbir referans noktasi olmadan yuzlerce karar vermesi gerektigini gosterir — ve bu kararlarin bazilari kacilmaz olarak yanlis olacaktir.

Gercek su: **insa etmek istediginiz seyin %99.9'u** zaten bir sekilde mevcut. Ister bir e-ticaret magazasi, ister bir SaaS kontrol paneli, ister bir portfolyo sitesi, ister bir sosyal medya uygulamasi veya bir rezervasyon platformu olsun — biri zaten benzer bir sey insa etmis. Ve bu projelerin cogu acik kaynakli, GitHub'da sablon olarak mevcut, klonlanip ozellestirilmeye hazir.

Is akisiniz her zaman ayni sekilde baslamalidir: **once bir sablon arayin.** GitHub'a gidin ve insa etmek istediginiz seye uyan acik kaynak projeleri arayin. Vizyonunuza yakin birini bulun, proje klasorunuze klonlayin ve sonra Claude Code'u ona yonlendirin. Artik sifirdan insa etmek yerine, ajan mevcut, calisan bir kod tabanini degistiriyor, gelistiriyor ve ozellestiriyor. Kalite ve hizdaki fark muazzamdir.

> **Sablonlar hile degil — stratejidir.** Profesyonel gelistiriciler her zaman standart sablonlar ve baslangic kitleri kullanir. Birinin urununu kopyalamiyorsunuz. Yapisal bir temel kullaniyorsunuz ve bunun uzerine kendi benzersiz urunuunuzu insa ediyorsunuz. Yapay zeka, her seyi sifirdan icat etmek yerine mevcut desenleri takip ettiginde cok daha iyi performans gosterir.

## Dogru Teknolojiyi Secmek

Sablon aramaniz sonucsuz kalirsa ve gercekten yeni bir proje baslatmaniz gerekiyorsa, sectiginiz teknoloji fark yaratabilir — ozellikle yapay zeka ajanlariyla calisirken. Bununla birlikte, tek bir zorunlu secim yoktur. En iyi yigin, sizi calisan bir urune en hizli ulastiran yigindir.

Claude Code, tum LLM'ler gibi, temelde **web tabanli kod** yazmada daha iyidir: HTML, CSS, JavaScript ve TypeScript. Bu internetin dilidir ve bu modellerin en cok uzerinde egitildigi seydir. Projeniz web teknolojilerine ne kadar yakin kalirsa, yapay zeka o kadar iyi performans gosterir.

**Web uygulamalari** icin, bulabilirseniz **Next.js** mukemmel bir tercihdir. SEO icin kritik olan yerlesik sunucu tarafi olusturma ile modern bir React frameworkudur. Claude Code, Next.js projeleriyle son derece iyi calisir: dosya tabanli yonlendirmeyi, API rotalarini, sunucu bilesenlerini ve tum ekosistemi anlar. GitHub'da hemen hemen her tur uygulama icin cok sayida Next.js sablonu bulabilirsiniz.

**Masaustu uygulamalari** (Windows, macOS veya Linux icin calistirilabilir dosyalar) icin **Electron** harika bir secenektir. Electron, HTML, CSS ve JavaScript kullanarak masaustu uygulamalar olusturmanizi saglar — Claude'un ustun oldugu ayni web teknolojileri. Kullanici arayuzu aslinda yerel bir pencere icinde olusturulan bir web sayfasi oldugu icin, yapay zeka bir web sitesi olusturmakla ayni kolaylikta guzel, islevsel masaustu uygulamalari olusturabilir.

Ancak iste onemli nüans: **eski bir teknolojideki iyi insa edilmis bir sablon, modern bir teknolojiyle sifirdan baslamaktan neredeyse her zaman daha iyidir.** Tam olarak ihtiyaciniz olan, eksiksiz, iyi yapilandirilmis, savastan gecmis, tum ozellikleri zaten uygulanmis bir PHP, jQuery veya Laravel projesi bulabilirsiniz. Bu durumda, onu kullanin. Yapay zeka ajanlari herhangi bir teknolojiyle calisabilir ve saglam, hazir bir temelden kazandiginiz zaman, daha yeni bir frameworkun teorik

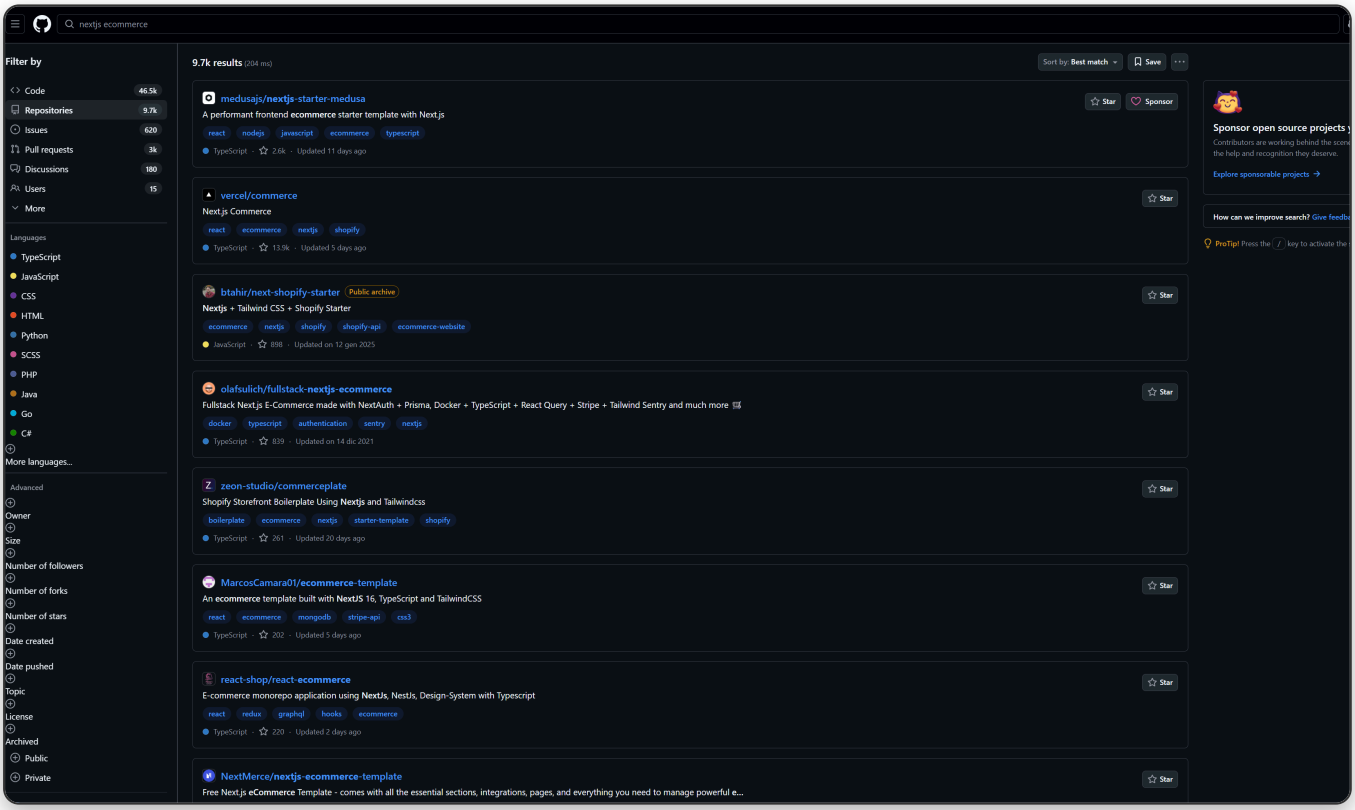
avantajlarindan cok daha degerlidir. Claude Code; PHP, Python, Ruby veya baska herhangi bir kod tabanini anlayabilir ve degistirebilir.

Oncelik basit: **mevcut en iyi sablonu bulun.** Benzer kalitede iki sablon bulursaniz ve biri Next.js, digeri PHP kullaniyorsa, Next.js ile gidin. Ancak PHP sablonu daha eksiksiz, daha fazla ozellige sahip ve daha iyi bakimliysa — tereddut etmeden onu secin. Baslangic noktasinin kalitesi ve eksiksizligi, yigin modernliginden daha onemlidir.

> **Temel kural:** sizi hedefinize en cok yaklastiran her seyi kullanin. Mevcut oldugunda modern web teknolojilerini (Next.js, Electron, React Native) hedefleyin, ancak harika bir sablonu sadece eski bir yigin kullandigi icin asla reddetmeyin. Saglam bir temelden kazanilan zaman her zaman yigin saflagindan daha degerlidir.

### Pratik ornek: sablon bulma

Diyelim ki bir e-ticaret magazasi olusturmak istiyorsunuz. Claude Code'a *"Bana sifirdan bir e-ticaret sitesi olustur"* demek yerine, GitHub'i acin ve **"nextjs ecommerce"** gibi bir sey arayin. Urun listeleri, alisveris sepetleri, odeme akislari ve odeme entegrasyonu zaten insa edilmis onlarca hazir proje bulacaksiniz.




*"nextjs ecommerce" icin hizli bir GitHub aramasi zaten birkac umut verici sablon gosteriyor.*

Akilda tutulmasi gereken onemli bir sey: GitHub'daki bircok sablon **freemium projelerdir** — cekirdek acik kaynakli ve ucretsizdir, ancak bazi ozellikler veya premium surumler odeme gerektirir. Bir sablona karar vermeden once her zaman deponun README'sini ve lisansini kontrol edin. Ihtiyaciniz olmayan ucretli abonelikler veya gizli maliyetler gerektiren her seyden kacinin.

Arama sonuclarina baktigimizda, **fullstack-nextjs-ecommerce**'i gorebiliriz — ve mukemmel bir baslangic noktasidir. Nedenini analiz edelim. Proje, TypeScript ile Next.js kullaniyor, bu da yapay zeka destekli gelistirme icin tam olarak istedigimiz sey. Ancak gercekten one cikan, zaten entegre olan seyler: **odemeler icin Stripe**, veritabani olarak **Prisma ile PostgreSQL** ve **kimlik dogrulama icin NextAuth**. Bunlar, herhangi bir web uygulamasinin en kritik — ve en fazla hataya acik — uc parcasidir.

**Stripe'in zaten entegre olmasi** ozellikle degerlidir. Odeme isleme, webhook'lar, oturum yonetimi, hata isleme ve dogru yapilmasi sasirtici derecede zor olabilen uc durumlar icerir. Odemelerde bir seyler ters gittiginde, zarari goren musterilerinizdir — basarisiz odemeler, tekrarlanan odemeler veya bozuk odeme akislari, guveni yiken ve size gercek para kaybettiren turden hatalardır. Zaten calisan bir Stripe entegrasyonu olan bir sablonla baslamak sizi bu bas agrilarindan kurtarir.

Proje ayrica veritabani olarak **PostgreSQL** kullaniyor, bu da saglam bir tercih. Postgres guvenilir, iyi belgelendirilmis, MySQL gibi alternatiflere kiyasla biraz daha iyi guvenlik varsayilanlari sunar ve Amazon AWS, Hetzner veya benzer saglayicilarda kolayca barindiribilir — sunucu kurulumu ve dagitimi bir sonraki bolumde ele alacagiz. **NextAuth**'un zaten baglanmis olmasi, kullanici kimlik dogrulamasinin kutudan cikar cikmaz halledildigi anlamina gelir; sifirdan insa etmeniz gerekmeyen baska bir karmasik parca daha.

Dogru sablonla baslamanin gucu budur: odemeler, veritabani ve kimlik dogrulama kurmak icin gunler (veya haftalar) harcamak yerine — hepsi yanlis yapilmasi kolay seyler — bu kritik sistemlerin zaten calisti bir projeyle baslarsiniz. Daha sonra tamamen urunu vizyonunuza gore ozellestirmeye odaklanabilirsiniz: tasarimi degistirme, urunlerinizi ekleme, is mantgini degistirme ve magazanizi benzersiz kilan ozellikleri insa etme.

Sablonunuzu bulduktan sonra, bir sonraki adim basit: **indirin ve calisma alani klasorunuze yerlestirin.** O klasoru Claude Code (veya Antigravity) ile acin ve calismaya baslayin. Claude Code'a sablonu dogrudan degistirmesini soyleyebilirsiniz — markayi degistirin, yeni sayfalar ekleyin, duzeni yeniden isleyin, yeni ozellikler entegre edin — veya sablonu kendi projeniz icin bir **referans** olarak kullanabilirsiniz. Biraz farkli bir sey insa ediyor olsaniz bile, sablonu ayni calisma alaninda tutmak, Claude Code'un ona bakabilecegi, kod desenlerini inceleyebilecegi ve yazdiklari icin temel olarak kullanabilecegi anlamina gelir.

Bu cok onemli bir nokta: **Claude Code'a her zaman referans olacak bir sey verin.** Ajanin bakabilecegi saglam, calisan bir kod tabani oldugunda, onemli olcude daha iyi kod yazar. Ayni desenleri takip eder, ayni kurallari kullanir ve tutarli ve guvenilir cikti uretir. Referans alacak hicbir seyi olmadan her seyi sifirdan uretmesi gerektiginde, hatalar o zaman olur — tutarsiz dosya yapilari, yanlis kutuphane surumleri, birbirine uymayan kod. Sablon, yapay zekayi temellendirilmis tutan ve yuksek kaliteli, tutarli sonuclar ureten bir capa gorevi gorur.

**Sablonda odeme veya veritabani yoksa ne olacak?** O da sorun degil. Bu kurs, herhangi bir projeye odeme entegre etmek icin dogrudan Claude Code'a verebileceginiz hazir Stripe entegrasyon dosyalari icerir. Veritabani icin, PostgreSQL veya sablonun zaten kullandigi veritabanini oneririz — guclu bir neden olmadikca sablonun secimlerine karsi cikmayin. Ayni sey kimlik dogrulama icin de gecerli: sablon NextAuth, Clerk veya baska bir kimlik dogrulama sistemi kullaniyorsa, onunla calisin. Hedef, zaten insa edilmis olani kullanmak, her seyi degistirmek degil.

# 4. Koddan Uretime

*Ürününüz insa edildi. Simdi odeme kabul etmesi, bir sunucuda calismasi ve dunya capinda kullanicilar icin hizli ve guvenli olmasi gerekiyor. Bu bolum, projenizi yerel koddan canli, uretim ortamina hazir bir isletmeye donusturen temel hizmetleri kapsiyor.*

**Bu bolum hakkinda bir not.** Burada isleri kisa ve oz tutuyoruz. Amacimiz size **ne** yapmaniz gerektigini ve **neden** yapmaniz gerektigini soylemek — zamaninizi harcayan uzun egitimler yazmak degil. Herhangi bir LLM (Claude, Gemini, ChatGPT) ayrintilari aciklayabilir, her adimda size yol gosterebilir ve sorularinizi statik bir sayfanin yapabileceginden cok daha iyi yanltlayabilir. Bir sey net degilse, ajaniniza sorun: *"Bir kurs takip ediyorum ve [X yapmaml] gerektigi yaziyor. Bu ne anlama geliyor ve bana adim adim gosterir misin?"* Bu daha hizli, daha kisisel ve her zaman gunceldir.

## Stripe ile Odemeler

Ürününüz bir sey satiyorsa, bir odeme islemcisine ihtiyaciniz var. **Stripe** sektorun standartlaridir: guvenilir, iyi belgelendirilmis ve ne kadar yaygin kullanildigi nedeniyle hemen hemen her yapay zeka ajani tarafindan desteklenir.

Yapmaniz gerekenler sunlar:

- **Stripe hesabi olusturun** stripe.com'da. **API anahtarlari** (on yuz icin yayinlanabilir anahtar, arka yuz icin gizli anahtar) ve hem **test modu** (gelistirme icin sahte odemeler) hem de **canli mod** (gercek para) erisimi alacaksiniz.
- **Webhook'lari kurun** Stripe kontrol panelinde. Webhook'lar, sunucunuzdaki URL'lerdir ve Stripe bir sey oldugunda onlari cagirir — bir odeme basarili olur, bir abonelik yenilenir, bir ucret basarisiz olur. Uygulamaniz bir aboneligi ne zaman etkinlestirecegini, bir siparisi ne zaman onaylayacagini veya bir basarisizligi ne zaman ele alacagini bu sekilde bilir. Hem **yerel test** (Stripe'in olaylari

localhost'unuza ileten bir CLI araci var) hem de **uretim** (canli sunucunuza yonlendirme) icin webhook'lara ihtiyaciniz var. Her seyi canli hale getirmeden once yerel olarak calistirin.

- **Stripe'i projenize entegre edin.** Sablonunuzda zaten Stripe varsa, sadece API anahtarlarinizi girin. Yoksa, bu kurs hazir Stripe entegrasyon dosyalari icerir — bunlari calisma alaniniza birakin ve Claude Code'a entegre etmesini soyleyin. Stripe, tek seferlik odemeler ve yinelenen abonelikleri destekler; her ikisi de sizin icin kart dogrulamayi, 3D Secure ve PCI uyumlulugun halleden barindirilan odeme sayfalari kullanir.

Temel bir kural: **odemeleri her zaman webhook'lar araciligiyla sunucu tarafinda dogrulayin**, asla on yuze guvenmeyin. Webhook, Stripe'in sunucunuza dogrudan paranin gercekten el degistirdigini soylemesindir — tek guvenilir dogruluk kaynagi budur.

## Barindirma: AWS, Hetzner ve Otesi

Uygulamanizin bir yerde yasamasi gerekiyor. Iyi haber: **AWS kaydoldugunuzda bir Ucretsiz Katman sunar — tam bir yil boyunca**, tamamen ucretsiz bir **t2.micro sunucu** ve bir **micro veritabani** alirsiniz. Fikrinizi dogrularken ve kullanici elde etmeye baslarken ilk projenizi barindirmak icin yeterlidir.

Bir AWS hesabi nasil kurulacagini veya Ucretsiz Katman nasil kullanilacagini bilmiyor musunuz? Sadece Claude veya Antigravity'ye sorun — her adimda size yol gosterecekler.

Ucretsiz katmani astiginizda, sunucu boyutlandirma hakkinda bilmeniz gerekenler:

- **t3.small** cogu uygulama icin ideal noktadir — iyi islemci, yeterli RAM ve makul fiyatlandirma (AWS'de ayda ~16$). "t" ornek neslini ifade eder; eski nesiller (t2, vb.) bazen daha az performans icin daha fazla maliyete neden olabilir, bu yuzden mevcut en son nesle bagli kalin.
- **Hetzner** *onemli olcude* daha ucuz bir Avrupa alternatifidir — t3.small'a benzer performansi yaklasik **ayda 4$**'a alabilirsiniz. Bu, benzer ozellikler icin AWS'den yaklasik 4 kat daha ucuzdur.
- **Her uygulamanin sunucuya ihtiyaci yoktur.** Projeniz statik bir site veya bir JAMstack uygulamasiysa, ozel bir sunucuya hic ihtiyaciniz olmayabilir. Vercel, Netlify veya hatta Cloudflare Pages gibi platformlar onu ucretsiz veya neredeyse ucretsiz barindirabilir. Her zaman LLM'nize sorun: *"Benim uygulamam icin en iyi barindirma nedir?"*

Onerimiz: ilk yiliniz icin **AWS Ucretsiz Katmani ile baslayin**, sonra Hetzner veya baska bir saglayicinin butceniz ve kitle konumunuz icin daha mantikli olup olmadigini degerlendirin. Kullanicilarinizin cogu Avrupa'daysa, Hetzner'in Almanya sunuculari size daha dusuk gecikme sunar. Kitleniz kuresel veya ABD merkezliyse, AWS bölgeleri daha uygun olabilir.

> **SSH anahtarlari ve yapay zeka sunucu yonetimi.** Claude Code'un sunucunuza uzaktan baglanmasini ve yonetmesini saglamak icin bir **SSH anahtari** kurmaniz gerekecek. Claude'dan bir tane olusturmasini ve sunucunuzda yapilandirmasini isteyin. Baglandiktan sonra, Claude kod dagitabilir, hizmetleri yonetebilir, sorunlari giderebilir — hepsi terminalinizden. Sunucu yonetimi gorevleri icin en iyi sonuclar icin **Opus** kullanmanizi oneririz, ancak token tasarrufu istiyorsaniz Sonnet de calisir (biraz daha yuksek hata olasiligi ile).

## Cloudflare: Performans ve Koruma

Uygulamaniz bir sunucuda calistiktan sonra, onu korumak ve hizlandirmak icin onune oturacak bir seye ihtiyaciniz var. Bu **Cloudflare**'dir. Iyi haber: **ucretsiz plan, web sitelerinin buyuk cogunlugu icin fazlasiyla yeterlidir**. Ucretli bir plana ihtiyaciniz yok.

Ancak once, bir **alan adina** ihtiyaciniz olacak. Dogrudan **Cloudflare** veya **Namecheap**'ten satin almanizi oneririz — her ikisi de guvenilir ve makul fiyatlidir. Alan adinizi aldiktan sonra, **DNS**'i AWS veya Hetzner sunucunuzun IP adresine yonlendirmek icin yapilandirmaniz gerekecek. LLM'nize sorun: *"[Cloudflare/Namecheap]'ten bir alan adi aldim. [IP adresiniz] adresindeki sunucuma isaret etmek icin DNS'i nasil kurabilirim?"* Size adim adim gosterecektir.

Cloudflare neden bu kadar onemli? Sitenizi **DDoS saldirilarina**, cesitli **guvenlik istismarlarindan** ve yaygin web guvenlk acklklarindan korur. Ayrica **kuresel bir onbellek** saglar, yani iceriginiizn kullanicilariniza yakin sunuculardan sunulmasini saglar ve sitenizi dunya capinda hizlandirir. Cloudflare gibi bir hizmet olmadan, sitenizi ciddi risklere maruz brakiyorsunuz — ozellikle simdi, yapay zeka caginda, herhangi birinin guvenlik aciklarini bulmak, yapilandirma hatalarinl istismar etmek veya hatta zayif korunan web sitelerinden veri calmak icin yapay zeka araclarini kullanabildigi bir donemde. Bunu atlamayin.

> **Hizli ipucu: Flexible SSL modu.** Cloudflare'i kurarken, **Flexible** SSL modunu secebilirsiniz. Bu, Cloudflare ile sunucunuz arasindaki baglantinin duz HTTP kullandigi, ancak Cloudflare ile son kullanicilariniz arasindaki baglantinin HTTPS oldugu anlamina gelir — boylece ziyaretciler guvenli kilit simgesini gorur. Bu, sunucunuzda SSL sertifikalari yapilandirma zahmetinden kurtarir, hizla baslamak icin harika. Hassas verileri isleeyen uretim uygulamalari icin, sonunda **Full** moda (uctan uca sifrelenmis) gecmelisiniz. Ancak ilk testleriniz ve lansmanlriniz icin Flexible gayet iyi ve cok fazla kurulum zamani tasarrufu saglar. Hangi modun projenize uygun oldugnndan emin degilseniz LLM'nize farklari aciklamasini isteyin.

Cloudflare sadece korumadan cok daha fazlasini sunar. Ucretsiz plan, bircok gelistiricinin bilmedigi guclu ozellikler icerir:

- **Cloudflare Pages** — frontend uygulamalari icin ucretsiz statik hosting (React, Next.js static export, Vue, vb.). GitHub'a push edersiniz, Cloudflare otomatik olarak derler ve deploy eder. Sifir yapilandirma, sifir maliyet. Landing page'ler, portfolyolar ve JAMstack uygulamalari icin mukemmel.
- **Edge Functions (Workers)** — ucretsiz planda, kullanicilariniza yakin edge'de sunucusuz kod calistirin. API rotalari, yonlendirmeler, A/B testleri ve ozel bir sunucuya ihtiyac duymadan hafif backend mantigi icin harika.
- **CDN & Caching** — statik varliklariniz (gorseller, CSS, JS) global olarak onbelleğe alinir, sitenizi dunyanin her yerinden son derece hizli hale getirir.

Kilit soru su: **uygulamanizin gercekten ozel bir sunucuya ihtiyaci var mi, yoksa Cloudflare bunu ucretsiz halledebilir mi?** Claude'a sorun: *"[Uygulamanizi tanimlayin] olusturuyorum. AWS/Hetzner'de ozel bir sunucuya ihtiyacim var mi, yoksa Cloudflare Pages ve Workers'i ucretsiz*

*kullanabilir miyim?"* Claude sizin ozel durumunuzu analiz edecek ve en iyi secenegi soyleyecek. Kac projenin tamamen Cloudflare'in ucretsiz katmaninda calisabilecegine sasirabilirsiniz.

## API Anahtarlari: Her Seyi Otomatiklestirin

Cogu egitimin atladigi, oyunun kurallarini degistiren bir ipucu: **hosting saglayicilariniz icin API anahtarlari olusturun** ve bunlari Claude'a verin. Bu, Claude'un altyapinizi dogrudan terminalden yonetmesini saglar — dashboard'larda tiklama yok, kopyala-yapistir yok, manuel is yok.

- **Cloudflare API Anahtari** — Cloudflare dashboard'unuza gidin → My Profile → API Tokens → bir token olusturun. Bununla Claude otomatik olarak DNS kayitlarini yapilandirabilir, Pages projeleri olusturabilir, Workers'i yonetebilir, SSL ayarlarini guncelleyebilir ve cok daha fazlasini yapabilir. Claude'a soyleyin: *"Iste Cloudflare API tokenim. Domainim icin sunucu IP'me isaret eden DNS'i kur."* Saniyeler icinde tamamlandi.
- **AWS Access Keys** — AWS IAM'a gidin → bir access key olusturun. Bununla Claude EC2 instance'larini yonetebilir, security group'lari yapilandirabilir, RDS veritabanlari olusturabilir, S3 bucket'larini yonetebilir ve tum AWS altyapinizi programatik olarak idare edebilir. Claude'a soyleyin: *"Iste AWS kimlik bilgilerim. us-east-1'de bir t3.small EC2 instance'i baslat ve bir web uygulamasi icin security group'lari yapilandir."*
- **Hetzner API Token** — Hetzner Cloud Console'unuza gidin → proje → Security → API Tokens → bir tane olusturun. Claude daha sonra sunucular olusturabilir, guvenlik duvarlari yapilandirabilir, snapshot'lari yonetebilir ve tum Hetzner altyapinizi idare edebilir. Claude'a soyleyin: *"Iste Hetzner API tokenim. Nurnberg'de Ubuntu ile bir CX22 sunucusu olustur."*

> **API anahtarlari hakkinda guvenlik notu.** Bu API anahtarlari gucludur — onlara sifre gibi davranin. **Asla Git'e commit etmeyin**, herkese acik paylasmayin ve guvenmedifginiz sohbet arayuzlerine yapistirmayin. Onlari bir `.env` dosyasinda saklayin veya dogrudan terminal oturumunuzda Claude'a iletin. Bir anahtar ele gecirilirse, hemen saglayicinin dashboard'undan iptal edin ve yenisini olusturun. Ayrica, API tokenlari olustururken **her zaman en az yetki ilkesini kullanin**: yalnizca gercekten gerekli olan izinleri verin, tam yonetici erisimi degil.

Bu API anahtarlarinin Claude ile kombinasyonu inanilmaz derecede gucludur. Kelimenin tam anlamiyla soyleyebilirsiniz: *"Bir Next.js uygulamam var. Cloudflare Pages'e deploy et, ozel domaini kur ve DNS'i yapilandir — iste API anahtarlarim."* Ve Claude her seyi otomatik olarak yapacak. Veya: *"AWS'de bir EC2 instance'i olustur, Node.js ve PM2 kur, nginx yapilandir, Cloudflare DNS'i kur ve uygulalami deploy et."* Tam altyapi kurulumu saatler degil dakikalar icinde.

## GitHub Actions ile CI/CD

Bolum 2'de Git ve GitHub CLI kurdugumuzda hatirliyorsunuz? Iste tum bunlarin karsligini aldigi yer burasi. `gh` dogrulandiginda, Claude Code'a **projenizi ozel bir GitHub deposuna gondermesini**, **GitHub Actions** kurmasini, gerekli tum **gizli anahtarlari** (sunucunuzun SSH anahtari, ortam

degiskenleri, vb.) yapilandirmasini ve otomatik dagitim hatti olusturmasini soyleyebilirsiniz — hepsi terminalinden, GitHub'in arayuzune dokunmanize gerek kalmadan.

Fikir basit: GitHub Actions yapilandirildiktan sonra, **Claude Code deponuza her kod gonderdiginde, otomatik olarak sunucunuza dagitilir.** Manuel SSH yok, dosya kopyalama yok, sunucuda kendiniz komut calistirma yok. Claude gonderir, GitHub Actions alir, SSH uzerinden AWS veya Hetzner sunucunuza baglanir ve her seyi dagitir. Tamamen otomatik.

Claude'a sunu soyleyin: *"Bu projeyi GitHub'da yeni bir ozel depoya gonder, her main'e push'ta sunucuma dagitim yapan bir GitHub Action kur. Iste sunucu IP'm ve SSH anahtarim."* Claude bunu nasil yapacagini zaten biliyor — is akisi dosyasini olusturacak, SSH anahtarini ve sunucu IP'sini GitHub gizli anahtarlari olarak ekleyecek ve tum hatti yapilandiracak. GitHub CLI'yi daha once tam olarak bu yuzden kurduk.

> **Dinamik IP tuzagi.** Statik IP'ler genellikle hem AWS'de hem de Hetzner'de ekstra maliyet getirir, bu yuzden paradan tasarruf etmek icin buyuk olasilikla **dinamik IP** kullanacaksiniz. Bu, sunucunuzu her durdurdugunuzda ve yeniden baslattiginizda IP'nin degisecegi anlamina gelir. Bu oldugunda, **iki yerde** guncellemeniz gerekecek: Cloudflare DNS kaydi ve `SERVER_IP` GitHub gizli anahtari. Claude'a sunucu IP'sini GitHub Actions'ta bir `SERVER_IP` gizli anahtari olarak saklamasini soyleyin, boylece degistiginde yalnizca bir degiskeni guncellemeniz yeterli. Ayrica Claude'a bir dagitim basarisiz olursa IP'yi kontrol etmenizi ve guncellemenizi hatirlatmasini soyleyin — degisen bir IP neredeyse her zaman nedendir.

# 5. Pazarlama ve SEO Taktikleri

> *Ürününüz yayinda. Simdi dunyanin onun var oldugunu bilmesi gerekiyor. Bagimsiz urunler icin en etkili pazarlama organik olandir — ucretsiz, yaratici ve dogru yapildiginda sasirtici derecede guclu. Bu bolum, kullandigimiz taktikleri kapsiyor.*

## Organik Buyume Stratejileri

Durust olalim: **en iyi pazarlama, pazarlama gibi gorunmez.** Internet reklamlarla doymus durumda ve insanlar tanitim gibi hissettiren her seyi gormezden gelme refleksi gelistirdi. Aslinda ise yarayan **gizli pazarlamadir** — reklam yerine gercek bilgi, soru veya oneri gibi gorunen icerik. Insanlar dogal olarak merakli ve onerilerde bulunmaya bayilir. Bunu kullanin.

**Reddit** bunun icin en guclu platformlardan biridir. Devasa, Google tarafindan yuksek oranda endekslenmis ve hedef kitlenizin zaten takildigi nis topluluklarla dolu. Ancak iste anahtar: **asla agresif pazarlama yapmayin.** *"Harika yeni siteme goz atin!"* yazmayin — bu oy dusurulecek, kaldirilacak veya banlanacaktir. Bunun yerine, su tarzi bir sey yazin:

- *"[bilinen rakip] veya [siteniz]'e benzer siteler onerebilir misiniz? Alternatif ariyorum."*

- *"[urun kategoriniz] deneyen var mi? [sitenizi] buldum ama diger secenekleri merak ediyorum."*
- Ilgili subreddit'lerde sorulari yanitlayin ve urunuunuzu gercekten yardimci oldugu yerlerde dogal olarak bahsedin.

Basarili bagimsiz pazarlamanin cogunlugu Reddit'te boyle calisir. Yalan soylemiyorsunuz — urunuunuzu gercek bir konusma icinde bir kesif olarak cerceveliyorsunuz. Insanlar merak ettikleri icin tiklar, kendilerine bir sey satildigi icin degil. Ve iste bir bonus: **Reddit gonderileri Google tarafindan endekslenir**, bu yuzden iyi yerlestirilmis bir konu size aylar hatta yillar boyunca organik arama trafigi getirebilir.

Ayrica gecici olarak artirmak icin kucuk bir butceyle bir **Reddit gonderisini sponsorlu yapabilirsiniz**. Bu, onu arama sonuclarinda yukari tasir, Google'in daha hizli endekslemesini saglar ve baslangic gorunurlugu verir. LLM'nize Reddit gonderi tanitiminin nasil calistigini ve hangi butcenin mantikli oldugunu sorun.

> **Huni hilesi.** Ana sayfanizda veya giris sayfanizda, **ana urunuunuzden bagimsiz olarak insanlari ceken** bir sey ekleyin — ucretsiz bir arac, trend bir konu, faydali bilgi veya su anda populer olan bir sey. Bu bir **huni** gorevi gorur: insanlar ucretsiz/ilginc icerik icin gelir, urunuunuzu kesfeder ve bir yuzdeleri donusum saglar. Sattiginiz seye yonlendiren gercek deger sunan bir yem olarak dusunun.

## SEO, Icerik ve Dagitim

Herhangi bir pazarlamaya baslamadan once, **analitik ve izlemenizi** kurun. Iki seye ihtiyaciniz var:

- **Google Analytics** — takip kodunu sitenize ekleyin (Claude'a entegre etmesini soyleyin). Ayrica trafiginizi telefonunuzdan istediginiz zaman kontrol edebileceginiz bir **mobil uygulama** da var. Bu size toplam ziyaretleri, kullanici davranisini, trafik kaynaklarini ve donusum verilerini gosterir.
- **Google Search Console** — bunu kurun ve Google Analytics'e baglayin. Search Console ozellikle **organik Google trafiginizi** takip eder: hangi arama sorgularinin insanlari sitenize getirdigini, arama sonuclarinda ne siklikla gorundugunuzu ve tiklanma oranlarinizi. Her iki araci nasil kuracaginizi ve baglayacaginizi LLM'nize sorun.

Butceniz varsa, **Google Ads** size bir baslangic ivmesi verebilir. Kisa sureligine reklam vermek, Google'in algoritmasiyla guven olusturmaya yardimci olur ve organik SEO'nuz buyurken erken trafik saglar. Ancak maliyetler hizla birikir, bu yuzden bunu kisa vadeli bir hizlandirici olarak gorun, uzun vadeli bir strateji olarak degil. LLM'niz Google Ads kurulumunu ve butcelemeyi ayrintili olarak aciklayabilir.

SEO'nun kendisi icin, temellerle baslayin. Tarayicinizin **DevTools**'unu (F12) acin, **Lighthouse**'a gidin ve bir rapor olusturun. Bu size Performans, Erisilebilirlik, En Iyi Uygulamalar ve **SEO** icin puanlar verir. Size duzelttmenizi soylediginii duzelttin — ve evet, duzeltmeleri Claude Code'a yaptirabilirsiniz.

Yapilmasi gereken temel SEO aksiyonlari:

- **Sayfalariniza ceviriler ekleyin**. Coklu dil destegi erisminizi onemli olcude arttirir. Claude Code'a projeniz icin uluslararasilastirma kurmasini isteyin.
- **Uygun meta etiketleri ekleyin** — baslik, aciklama, sosyal paylasim icin Open Graph etiketleri, yapilandirilmis veri. Bunlar sitenizin Google arama sonuclarinda nasil gorunecegini dogrudan etkiler.
- **Bir site haritasi olusturup gonderin.** Guncel bir site haritasi olusturun ve Google Search Console'a yukleyin. Bu, Google'a sitenizde tam olarak hangi sayfalarin var oldugunu soyler ve daha hizli endekslenmelerine yardimci olur.

> **SEO sabir gerektirir.** Bir gecede organik trafik beklemeyin — Google'in sayfalarinizi duzgun sekilde endekslemesi ve siralamasi haftalar veya aylar alir. Ancak basladiginda, **bir kurus harcamadan gelen ucretsiz trafiktir**. Google Ads uzerinden yuzlerce euro odeyeceginiz tiklamalar, organik arama yoluyla ucretsiz gelecektir. Bu arada, baslangic gorunurlugu ve geri baglantilar olusturmak icin Reddit gibi sosyal platformlarda calisin. SEO uzun vadeli bir oyundur, ancak yapabileceginiz en degerli pazarlama yatirimidir.

### Yukselen trend: yapay zeka guduml trafik

Cogu kisinin henuz dikkat etmedigi yeni ve hizla buyuyen bir trafik kaynagi var: **LLM'lerin sitenizi onermesi.** ChatGPT, Gemini, Claude ve diger yapay zeka asistanlari giderek daha fazla arama motoru olarak kullaniliyor. Birisi *"[urun kategoriniz] icin iyi bir site nedir?"* diye sordugunsa, bu modeller belirli web sitelerini onerebilir ve onerirler — ve bu gercek trafik saglar. Kendi ziyaretlerimizin onemli bir bolumu ChatGPT ve diger LLM'lerden gelir.

Bundan yararlanmak icin, **Cloudflare kontrol panelinize** gidin ve **yapay zeka tarayicilarini engelleyen secenegi devre disi biraktiginizdan** emin olun. Bircok site varsayilan olarak yapay zeka botlarini engeller, bu da LLM'lerin icerikleriniz hakkinda bilgi edinmesini onler. Yapay zeka tarayicilarinin sitenize erismesine izin vererek, bu modellerin sayfalarinizi endekslemesine, ne sundugunuzu anlamasina ve gelecekte potansiyel olarak sizi kullanicilara onermesine olanak tanirsiniz. Bu aslinda **yapay zeka cagi icin ucretsiz SEO**'dur — ve etkisi muazzam olabilir.

> **Google'in otesini dusunun.** Geleneksel SEO, Google aramasini hedefler. Ancak yapay zeka guduml oneriler buyuk bir trafik kaynagi haline geliyor — ve bu trend sadece hizlaniyor. Sitenizin yapay zeka tarayicilarina erisilebilir oldugundan, net ve aciklayici icerge sahip oldugundan ve LLM'lerin ne sundugunuzu kolayca anlayabilecegi sekilde iyi yapilandirilmis oldugundan emin olun. Simdi yapay zeka kesfine kendini konumlandiran siteler, bu kanal buyudukce buyuk bir avantaja sahip olacak.

# Tesekkurler!

Bu kursu satin aldiginiz ve zamaninizi ve paranizi bize emanet ettiginiz icin tesekkur ederiz. Gercekten degerli buldunuz ve gercek bir sey insa etmenize yardimci oldugunu umuyoruz.

Sizden haber almak isteriz. **Discord toplulugmuza katlin** apifreellm.com adresinden — takildgimiz, guncellemeleri paylastigimiz ve birbirimize yardimci oldugumuz yer orasi.

Bir dakikaniz varsa, **bize bir yorum yazin** ve ne dusundugunuzu soyeleyin. En faydali buldugununuz sey neydi? Ne eksikti? Ne daha iyi olabilirdi? Geri bildiriminizle gercekten ilgileniyoruz — bu kurs yasayan bir projedir ve **sizin** gercekten ihtiyac duydugunuz seylere gore gelistirmeye devam etmek istiyoruz.

Discord'da gorusmek uzere. Simdi gidin ve harika bir sey insa edin.