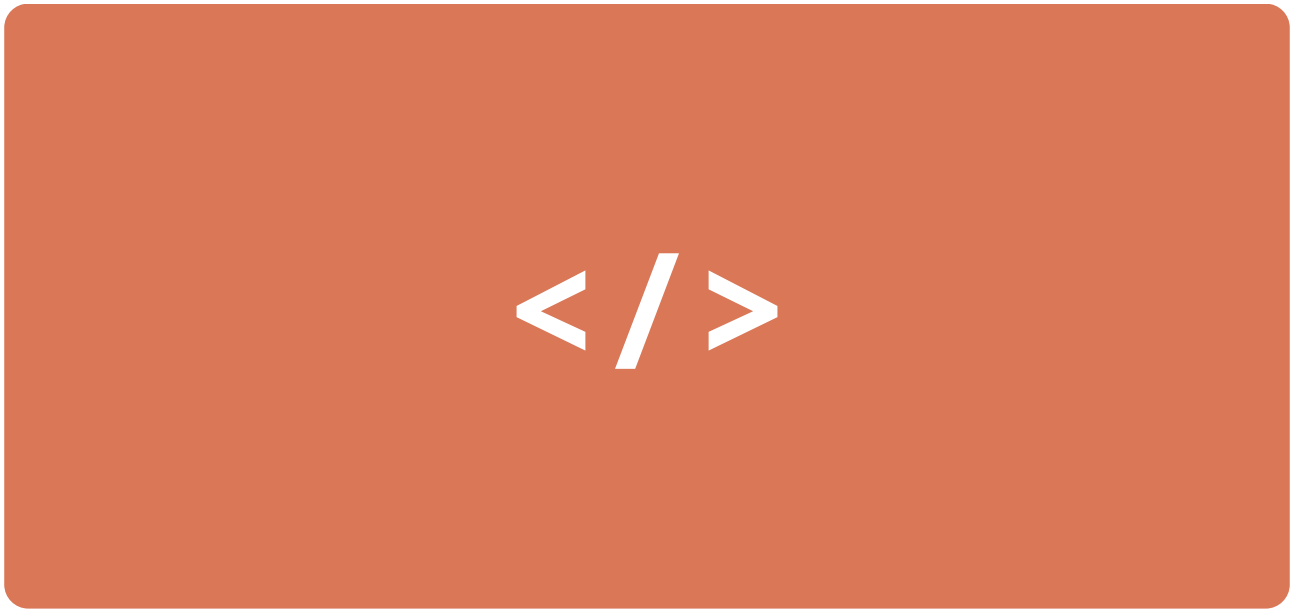
JJ.

CLAUDE CODE

Hur team använder Claude Code



Interna team transformerar sina arbetsflöden med Claude Code, vilket gör det möjligt för utvecklare och icke-teknisk personal att ta sig an komplexa projekt, automatisera uppgifter och överbrygga kompetensluckor som tidigare begränsade deras produktivitet.

Genom intervjuer med våra egna 'power users' har vi samlat insikter om hur olika avdelningar utnyttjar verktyget, dess inverkan på deras arbete, och tips för andra organisationer.

Innehåll

Claude Code för datainfrastruktur 3

Claude Code för produktutveckling 5

Claude Code för security engineering 7

Claude Code för inference 9

Claude Code för data science & visualisering 11

Claude Code för api knowledge 13

Claude Code för growth marketing 15

Claude Code för product design 17

Claude Code för rl engineering 19

Claude Code för juridik (legal) 21

Claude Code för Datainfrastruktur

Datainfrastruktur-teamet organiserar all affärsdata för team över hela företaget. De använder Claude Code för att automatisera rutinmässiga datauppgifter, felsöka komplexa infrastrukturproblem och skapa dokumenterade arbetsflöden så att både tekniska och icke-tekniska medlemmar kan hantera data självständigt.

Huvudsakliga användningsområden

Kubernetes-felsökning med skärmdumpar

När Kubernetes-kluster gick ner och inte schemalade nya poddar använde teamet Claude Code för att diagnostisera problemet. De matade in skärmdumpar av instrumentpaneler, vilket guidade dem genom moln-UI:t tills de hittade en varning om att IP-adresserna var slut. Claude Code gav sedan de exakta kommandona för att skapa en ny IP-pool, utan att behöva blanda in nätverksspecialister.

Klartext-arbetsflöden för ekonomiteamet

Teamet visade ekonomiavdelningen hur man skriver textfiler som beskriver dataflöden och sedan laddar dem i Claude Code för automatiserad exekvering. Medarbetare utan kodkunskap kunde beskriva steg som 'fråga denna dashboard, hämta info, kör dessa frågor, skapa Excel-output', och verktyget utförde hela arbetsflödet.

Navigering i kodbasen för nyanställda

När nya data scientists börjar, hänvisas de till Claude Code för att navigera i den massiva kodbasen. Verktyget läser dokumentationsfiler, identifierar relevanta filer för specifika uppgifter och förklarar beroenden i datapipelines, vilket ersätter traditionella datakataloger.

Parallell hantering av uppgifter

Vid långvariga datauppgifter öppnar de flera instanser av Claude Code i olika repositories. Varje instans behåller sin kontext, så när de byter tillbaka efter timmar eller dagar minns verktyget exakt vad som gjordes.

Claude Code för Datainfrastruktur

Teamets påverkan

Löste problem utan specialistkompetens

Löste Kubernetes-klusterproblem som normalt kräver nätverksteamet, genom att diagnostisera och fixa fel på minuter.

Accelererad onboarding

Nya analytiker kan snabbt förstå komplexa system och bidra meningsfullt utan omfattande handledning.

Möjliggjorde cross-team self-service

Ekonomiteam utan kodningserfarenhet kan nu exekvera komplexa dataarbetsflöden självständigt.

Bästa tipsen från teamet

Skriv detaljerade dokumentationsfiler

Ju bättre du dokumenterar arbetsflöden och verktyg i filer som Claude kan läsa, desto bättre presterar den. Detta gör att den briljerar på rutinuppgifter.

Använd MCP-servrar istället för CLI för känslig data

Rekommendationen är att använda MCP-servrar snarare än CLI för bättre säkerhetskontroll över vad Claude Code kan komma åt, särskilt vid hantering av känslig data.

Dela sessioner med teamet

Teamet höll sessioner där medlemmar demonstrerade sina arbetsflöden. Detta spred 'best practices' som andra inte hade upptäckt på egen hand.

Claude Code för Produktutveckling

Teamet som utvecklar Claude Code använder sin egen produkt för att bygga uppdateringar, vilket expanderar produktens enterprise-funktioner och agentiska loop-funktionalitet.

Huvudsakliga användningsområden

Snabb prototyping med auto-accept

Ingenjörer använder 'auto-accept mode' (shift+tab) för snabb prototyping. De ger Claude abstrakta problem, låter den arbeta autonomt, och granskar sedan en 80% färdig lösning innan de tar över för slutjusteringar.

Synkron kodning för kärnfunktioner

För kritiska funktioner som rör affärslogik arbetar teamet synkront med Claude Code. De övervakar processen i realtid för att säkerställa kodkvalitet och arkitektur, medan Claude hanterar det repetitiva kodandet.

Bygga Vim-mode

Ett av de mest framgångsrika asynkrona projekten var implementeringen av Vim key-bindings. Claude byggde hela funktionen (trots att det inte var en prioritet), och cirka 70% av den slutliga implementeringen kom från Claudes autonoma arbete.

Testgenerering och buggfixar

De använder verktyget för att skriva omfattande tester efter implementering av funktioner och för att hantera enkla buggfixar som identifierats i Pull Request-reviews.

Claude Code för Produktutveckling

Teamets påverkan

Snabbare funktion implementering

Implementerade komplexa funktioner som Vim-mode där 70% av koden skrevs autonomt.

Förbättrad utvecklingshastighet

Kan snabbt prototypa funktioner och iterera på idéer utan att fastna i implementationsdetaljer.

Ökad kodkvalitet via automatiska tester

Claude genererar omfattande tester och hanterar rutinmässiga buggfixar.

Bästa tipsen från teamet

Skapa självgående loopar

Sätt upp Claude att verifiera sitt eget arbete genom att köra builds, tester och linting automatiskt. Detta fångar misstag tidigt.

Utveckla intuition för uppgiftsklassificering

Lär dig skilja på uppgifter som fungerar bra asynkront (perifera funktioner, prototyping) kontra de som kräver synkron övervakning (kärnaffärslogik).

Formulera tydliga, detaljerade prompts

När komponenter har liknande namn eller funktioner, var extremt specifik i dina förfrågningar för att undvika ändringar i fel delar av kodbasen.

Claude Code för Security Engineering

Säkerhetsteamet fokuserar på att säkra mjukvaruutvecklingens livscykel och supply chain. De använder Claude Code omfattande för att skriva och debugga kod.

Huvudsakliga användningsområden

Komplex infrastruktur-felsökning

Vid incidenter matar de in stack traces och dokumentation för att spåra flöden genom kodbasen. Detta minskar tiden avsevärt för att förstå problem som normalt skulle ta 10-15 minuter av manuell kodläsning – nu tar det cirka 5 minuter.

Terraform kodgranskning och analys

För infrastrukturändringar kopierar de Terraform-planer till Claude Code och frågar 'vad kommer detta göra? kommer jag ångra detta?'. Detta skapar snabbare feedback-loopar.

Syntes av dokumentation och runbooks

De låter verktyget läsa flera dokumentationskällor för att skapa markdown-runbooks och felsökningsguider.

Kontextbyte och onboarding

När de bidrar till existerande projekt använder de verktyget för att skriva, granska och exekvera specifikationer, vilket möjliggör meningsfulla bidrag inom dagar istället för veckor.

Claude Code för Security Engineering

Teamets påverkan

Minskad tid för incidentlösning

Infrastrukturfelsökning som normalt tar 10-15 minuter tar nu ca 5 minuter.

Förbättrad säkerhetsgranskning

Terraform-kodgranskningar sker mycket snabbare, vilket eliminerar flaskhalsar för utvecklare.

Bättre dokumentationsflöde

Syntetiserade felsökningsguider från flera källor skapar effektivare processer.

Bästa tipsen från teamet

Använd custom slash commands

Security engineering använder egna kortkommandon i stor utsträckning för att effektivisera specifika arbetsflöden.

Låt Claude tala först

Istället för att ställa riktade frågor, ber de nu Claude att 'commit your work as you go' och arbeta autonomt med periodiska avstämningar.

Utnyttja för dokumentation

Utöver kodning är verktyget utmärkt på att syntetisera dokumentation och skapa strukturerad output för Slack eller Google Docs.

Claude Code för Inference

Inference-teamet hanterar minnessystemet som lagrar information. Teammedlemmar, särskilt de som är nya inom machine learning, använder Claude Code för att överbrygga kunskapsluckor.

Huvudsakliga användningsområden

Förståelse av kodbas och onboarding

Teamet förlitar sig på verktyget för att snabbt förstå arkitekturen i komplexa kodbaser. De får svar på sekunder om vilka filer som anropar specifika funktioner istället för att leta manuellt.

Generering av enhetstester med edge cases

Efter att ha skrivit kärnfunktionalitet ber de Claude skriva omfattande unit tests. Claude inkluderar automatiskt missade edge cases, vilket sparar mental energi.

Förklaring av ML-koncept

Utan bakgrund inom machine learning använder teammedlemmar Claude för att förstå modelspecifika funktioner. Det som tog en timme att Googla tar nu 10-20 minuter.

Översättning av kod mellan språk

Vid testning i olika språk (t.ex. Rust) förklarar de logiken och låter Claude skriva koden, vilket eliminerar behovet att lära sig nya språk enbart för testning.

Claude Code för Inference

Teamets påverkan

Accelererad inlärning av ML-koncept

Research-tid minskad med 80%.

Snabbare navigering i kodbasen

Kan hitta relevanta filer och förstå systemarkitektur på sekunder.

Eliminering av språkbarriärer

Kan implementera funktionalitet i språk som Rust utan att behöva behärska dem fullt ut.

Bästa tipsen från teamet

Testa Knowledge Base-funktionen först

Prova att ställa frågor för att se om Claude svarar snabbare än Google. Om det är snabbare och mer exakt är det ett värdefullt tidsbesparande verktyg.

Börja med kodgenerering

Ge specifika instruktioner, låt den skriva logiken, och verifiera sedan. Detta bygger tillit till verktyget.

Använd för att skriva tester

Att låta Claude skriva unit tests lättar på trycket i det dagliga utvecklingsarbetet.

Claude Code för Data Science & Visualisering

Data Science och ML Engineering-team behöver sofistikerade visualiseringsverktyg. Claude Code gör det möjligt för dem att bygga analysdashboards av produktionskvalitet utan att vara fullstack-utvecklare.

Huvudsakliga användningsområden

Bygga JavaScript/Typescript dashboards

Trots begränsad kunskap i JS/TS bygger teamet hela React-applikationer för att visualisera RL-modellprestanda. De ger Claude kontroll att skriva fullständiga applikationer (t.ex. 5000 rader TypeScript) från grunden.

Hantera repetitiv refactoring

Vid merge conflicts eller refactoring som är för komplex för editormacron men för liten för ett stort projekt, används Claude som en 'enarmad bandit' – committa status, låt den jobba 30 min, acceptera eller börja om.

Skapa bestående analysverktyg

Istället för 'slit-och-släng' Jupyter notebooks bygger de nu återanvändbara React-dashboards.

Zero-dependency delegering

För uppgifter i helt okända kodbaser delegerar de hela implementationen till Claude Code.

Claude Code för Data Science & Visualisering

Teamets påverkan

2-4x tidsbesparing

Rutinmässig refactoring går mycket snabbare.

Byggde komplexa appar i okända språk

Skapade 5000-rader TypeScript-appar med minimal erfarenhet.

Gick från engångsverktyg till bestående verktyg

Ersatte Jupyter notebooks med återanvändbara dashboards.

Bästa tipsen från teamet

Behandla det som en enarmad bandit

Spara din status, låt Claude jobba i 30 minuter. Att börja om från början har ofta högre framgångsfrekvens än att försöka bråka med korrigeringar.

Avbryt för enkelhet

Tveka inte att stoppa Claude och fråga 'varför gör du så här? försök något enklare'. Modellen tenderar mot komplexa lösningar men svarar bra på uppmaningar om förenkling.

Claude Code för API Knowledge

API-teamet arbetar med funktioner som PDF-stöd och citations. Att arbeta över stora, komplexa kodbaser innebär ständig kontakt med okänd kod. Claude Code fungerar som en guide.

Huvudsakliga användningsområden

Planering av första steget

Teamet använder Claude Code som sitt 'första stopp' för att identifiera vilka filer som ska undersökas för buggfixar eller analys.

Självständig debugging

Teamet har nu självförtroende att ta sig an buggar i okända delar av kodbasen istället för att be andra om hjälp. De kan fråga 'Tror du att du kan fixa denna bugg?' och få omedelbara framsteg.

Eliminering av overhead vid kontextbyte

Istället för att kopiera kodsnuttar till en webbläsare kan de ställa frågor direkt i terminalen.

Claude Code för API Knowledge

Teamets påverkan

Ökat självförtroende i okända områden

Medlemmar kan självständigt debugga incidenter i okända kodbaser.

Snabbare onboarding vid rotation

Ingenjörer som byter team kan snabbt navigera nya miljöer.

Ökad utvecklarnöjdhet

Minskad friktion i dagliga arbetsflöden.

Bästa tipsen från teamet

Se det som en iterativ partner

Förvänta dig inte att Claude löser allt direkt. Närma dig det som en kollaboratör du itererar med.

Börja med minimal information

Börja med det minsta du behöver och låt Claude guida dig, snarare än att front-loada med massiva förklaringar.

Claude Code för Growth Marketing

Growth Marketing fokuserar på betald sök, sociala medier och SEO. Som ett icke-tekniskt team på en person använder de Claude Code för att automatisera uppgifter och skapa agentiska arbetsflöden.

Huvudsakliga användningsområden

Automatiserad generering av Google Ads

Teamet byggde ett arbetsflöde som bearbetar CSV-filer med hundratals annonser, identifierar underpresterande varianter och genererar nya som möter strikta teckenbegränsningar.

Figma-plugin för massproduktion

De utvecklade ett Figma-plugin som identifierar frames och programmatiskt genererar upp till 100 annonsvarianter genom att byta rubriker och beskrivningar. Detta möjliggör 10x output.

Meta Ads MCP-server

De skapade en MCP-server integrerad med Meta Ads API för att fråga om kampanjprestanda direkt i Claude Desktop.

Avancerad prompt engineering med minne

Implementerade ett rudimentärt minnessystem som loggar hypoteser och experiment över tid.

Claude Code för Growth Marketing

Teamets påverkan

Dramatisk tidsbesparing

Skapande av annonstexter minskade från 2 timmar till 15 minuter.

10x ökning i kreativ output

Kan testa betydligt fler varianter över kanaler.

Opererar som ett större team

En person kan hantera uppgifter som traditionellt kräver dedikerade ingenjörsresurser.

Bästa tipsen från teamet

Identifiera API-aktiverade repetitiva uppgifter

Leta efter arbetsflöden med verktyg som har API:er (annonsplattformar, designverktyg). Dessa är perfekta kandidater.

Bryt ner komplexa flöden

Skapa separata agenter för specifika uppgifter (t.ex. en rubrik-agent och en beskrivnings-agent) istället för en som gör allt.

Claude Code för Product Design

Designteamet specialiserar sig på att bygga AI-produkter. Även icke-utvecklare kan använda Claude Code för att överbrygga gapet mellan design och ingenjörskonst.

Huvudsakliga användningsområden

Front-end polish och state management

Istället för omfattande dokumentation implementerar de nu visuella justeringar (typsnitt, färger) direkt. Ingenjörer noterar att designers nu gör 'stora state management-ändringar'.

Snabb interaktiv prototyping

Genom att klistra in mockups i Claude Code genererar de fullt fungerande prototyper som ingenjörer omedelbart kan förstå.

Upptäckt av edge cases

Använder verktyget för att kartlägga feltillstånd och logiska flöden, vilket identifierar problem under designfasen snarare än senare.

Komplexa textändringar

För uppgifter som att ta bort 'research preview'-meddelanden i hela kodbasen, använde de Claude Code för att hitta alla instanser och implementera uppdateringar.

Claude Code för Product Design

Teamets påverkan

Transformerat kärnarbetsflöde

Claude Code är nu ett primärt designverktyg, öppet 80% av tiden.

2-3x snabbare exekvering

Visuella ändringar som krävde fram-och-tillbaka implementeras nu direkt.

Veckor till timmar i cykeltid

Komplexa projekt som lanseringsmeddelanden görs nu på två 30-minuters samtal.

Bästa tipsen från teamet

Få hjälp med setup

Ta hjälp av ingenjörer för initial setup av repositories och rättigheter.

Använd custom memory files

Skapa instruktioner som berättar för Claude att du är en designer med lite koderfarenhet som behöver detaljerade förklaringar.

Klistra in bilder

Använd Command+V för att klistra in skärmdumpar direkt. Det är ovärderligt för att omvandla statiska mockups till interaktiva prototyper.

Claude Code för RL Engineering

RL Engineering-teamet (Reinforcement Learning) fokuserar på effektiv sampling. De använder verktyget för att skriva små till medelstora funktioner och debugga.

Huvudsakliga användningsområden

Funktionsutveckling med övervakad autonomi

Teamet låter Claude skriva det mesta av koden för mindre funktioner medan de tillhandahåller tillsyn.

Testgenerering och code review

Efter egna ändringar ber de Claude lägga till tester. Detta automatiserade flöde sparar tid på kvalitetssäkring.

Debugging och felundersökning

Används för att debugga fel med blandade resultat – ibland identifieras problem omedelbart, andra gånger krävs mer styrning.

Kubernetes driftvägledning

Ställer ofta frågor om Kubernetes-operationer som annars skulle kräva omfattande Googling.

Claude Code för RL Engineering

Teamets påverkan

Möjliggjorde experimentellt arbetssätt

Använder en 'try and rollback'-metodik där de ofta committar checkpoints för att testa Claudes lösningar riskfritt.

Dokumentationsacceleration

Claude lägger automatiskt till hjälpsamma kommentarer.

Hastighetsökning med begränsningar

Kan implementera små PRs snabbt, men kräver ofta manuell intervention eller guidning ca 2/3 av gångerna.

Bästa tipsen från teamet

Anpassa din dokumentationsfil

Lägg till instruktioner för att förhindra att Claude upprepar misstag, som att säga åt den att använda rätt sökvägar för tester.

Använd ett checkpoint-tungt flöde

Committa ofta så att du enkelt kan rulla tillbaka när experiment inte fungerar.

Prova one-shot först, samarbeta sen

Ge en snabb prompt och låt den försöka en full implementering först. Om det funkar har du sparat tid.

Claude Code för Juridik (Legal)

Det juridiska teamet upptäckte potentialen genom experiment. De använder det för att automatisera arbetsflöden och skapa prototyper.

Huvudsakliga användningsområden

Anpassade tillgänglighetslösningar

Teammedlemmar byggde kommunikationsassistenter för familjemedlemmar med talsvårigheter. På en timme skapade de en app med text-till-tal.

Automatisering av arbetsflöden

Skapade prototyper av 'telefonträd'-system för att hjälpa kollegor hitta rätt jurist.

Team-koordineringsverktyg

Chefer byggde G Suite-applikationer som automatiserar veckouppdateringar och spårar granskningsstatus.

Snabb prototyping för validering

Bygger snabbt funktionella prototyper för att visa ämnesexperter och validera idéer.

Claude Code för Juridik (Legal)

Teamets påverkan

Visuellt orienterat arbetssätt

Använder skärmdumpar för att visa vad de vill ha, itererar sedan baserat på visuell feedback.

Prototyp-driven innovation

Övervinner rädslan för att dela 'leksaksprototyper', vilket inspirerar andra.

Medvetenhet om säkerhet och compliance

Som produktjurister identifierar de omedelbart säkerhetsimplikationer av djupa integrationer.

Bästa tipsen från teamet

Planera i chatten, bygg i koden

Använd konversationsgränssnittet för att planera idén, flytta sedan till Claude Code för implementering.

Arbeta inkrementellt

Be Claude sakta ner och ta ett steg i taget.

Dela prototyper trots imperfektion

Dela ofärdiga verk för att tända gnistan för innovation hos andra.