

Quadern de pràctiques

Claude Code per a perfils tècnics · 6 hores · Cas Cafès Roca

Aquest quadern recull totes les pràctiques de les dues sessions, pas a pas. Totes giren al voltant del mateix negoci fictici, **Cafès Roca**, una petita cadena de cafeteries de Barcelona amb tres locals. A la carpeta de materials hi trobaràs la web i les dades de la sessió 1 ([web-cafes-roca](#)), l'aplicació d'estoc de la sessió 2 ([estoc-cafes-roca](#)) i, a [solucions/](#), l'estat final de cada pràctica.

Cada pràctica té un objectiu, els passos, els prompts i les ordres de referència i el que has de comprovar abans de donar-la per acabada. Els prompts són orientatius: adapta'ls, però mantén les quatre peces d'una bona petició (context, instrucció, format i com es comprova).

Abans de començar

Necessites un compte de Claude amb pla Pro o superior, un compte gratuït de GitHub i Git, Node.js i GitHub CLI instal·lats. La guia de preparació és a l'apartat «Preparació prèvia», al final del quadern.

Contingut

- **Sessió 1 · Crear amb Claude Code:** Claude Code: l'agent per a projectes de programari · Posada a punt i primera sessió · Una landing page a partir d'un brief · Anàlisi tècnica de dades · Un dashboard interactiu · Context, cost i bones pràctiques
- **Sessió 2 · Repositoris, qualitat i automatització:** Git i GitHub amb Claude Code · Treballar en paral·lel i revisar · Personalitzar l'agent · Connectar eines: MCP, connectors i plugins · Automatitzar i publicar · Criteri
- **Consulta · Preparació, ordres i fonts:** Resultats d'aprenentatge · Preparació prèvia · Ordres de referència · Glossari · Documentació oficial

Sessió 1 · Crear amb Claude Code

Blocs 1–6 · landing, dades i dashboard

Bloc 1 · Claude Code: l'agent per a projectes de programari

On viu, com treballa, quins models fa servir i què inclou cada pla.

Claude Code és l'agent de programació d'Anthropic. Llegeix un projecte sencer, fa un pla, escriu i edita fitxers, executa ordres i proves, i treballa amb Git i GitHub. Tu decideixes què pot fer sol i què t'ha de preguntar.

On el fas servir	Com s'obre	Quan el farem servir al curs
Terminal	Escrius claude dins de la carpeta del projecte	És l'entorn de referència de totes les pràctiques
Pestanya Code de l'aplicació d'escriptori	Aplicació de Claude > Code > tries la carpeta	Sessions en paral·lel, vista prèvia i revisió de canvis (bloc 8)
VS Code i JetBrains	Extensió de VS Code o connector de JetBrains	Si ja treballes amb un d'aquests editors
Navegador	claude.ai/code , sessions al núvol	Tasques llargues que continuen sense el teu ordinador (bloc 11)

El bucle de l'agent

Totes les pràctiques del curs segueixen el mateix mètode. És el que recomana la guia de bones pràctiques d'Anthropic i és el que fa que el resultat sigui fiable.

- **Explorar.** Llegeix els fitxers i entén què hi ha abans de tocar res.
- **Planificar.** Proposa un pla que pots revisar i corregir. Amb el mode Plan no modifica res.
- **Implementar.** Escriu el codi, executa ordres i corregeix el que falla.
- **Verificar.** Executa proves, obre la pàgina i compara les xifres. Tu decideixes si està bé.
- **Desar.** Un commit amb un missatge clar, perquè cada pas es pugui recuperar.

Models i esforç

Àlies	Model	Quan fer-lo servir
sonnet	Sonnet 5.5	La feina de cada dia: ràpid i bo en codi. És el que farem servir per defecte
opus	Opus 5.5	Plans, canvis grans i errors difícils
opusplan	Opus per planificar, Sonnet per executar	Tasques llargues en què el pla és la part delicada
haiku	Haiku 4.5	Tasques petites i repetitives
fable	Fable 5.1	El model més capaç, amb un límit d'ús propi

El model es canvia amb **/model** i l'esforç amb **/effort** (**low**, **medium**, **high**, **xhigh** i **max**). Més esforç vol dir que l'agent pensa més abans d'actuar: més qualitat en tasques complexes, més temps i més consum. Amb Sonnet 5.5 i Opus 5.5 l'esforç per defecte és **medium**.

Plans i límits d'ús, a 10 d'octubre de 2026

	Pro	Max	Team i Enterprise
Preu	15 € al mes amb pagament anual o 18 € al mes, sense impostos	Des de 90 € al mes	Per usuari, amb administració
Claude Code	Terminal, aplicació d'escriptori, editors i navegador	El mateix, amb 5 o 20 vegades l'ús de Pro	Segons la configuració de l'organització
Límits	Finestres de cinc hores i un màxim setmanal	Més alts	Per usuari

On es comprova: els límits es comparteixen amb el xat de Claude i Cowork. Els consultes amb /usage dins de Claude Code o a Configuració > Ús. Els preus actuals són a claude.com/pricing.

L'agent s'equivoca amb seguretat

Claude Code escriu codi que funciona la major part de les vegades i, quan s'equivoca, ho fa amb el mateix to segur. Per això cada pràctica acaba amb una comprovació que fas tu: proves, xifres, la pàgina oberta al navegador.

Pràctica 1 · Primer contacte amb l'agent

Objectiu. Obrir Claude Code sobre la carpeta del cas, fer-li explorar el projecte sense modificar res i provar el selector de model i d'esforç.

1. Descomprimeix `materials_claude_code_6h.zip` a l'Escriptori. Obre la carpeta `web-cafes-roca` amb VS Code (**File > Open Folder**) i obre un terminal (**Terminal > New Terminal**). A Windows també pots fer clic amb el botó dret a la carpeta > **Obre al Terminal**.
2. Escriu `claude` i prem Retorn. La primera vegada et demana si confies en la carpeta: respon que sí.
3. Prem **Maj+Tab** fins que a sota del quadre de text surti  **plan mode on**. En aquest mode l'agent llegeix i pensa, però no modifica res.
4. Enganxa el prompt d'exploració i llegeix la resposta.
5. Escriu `/model` per veure el model actiu i `/effort` per veure el nivell d'esforç. Deixa `sonnet` i `medium`.
6. Fes-li una pregunta concreta sobre el brief per comprovar que l'ha llegit.

TERMINAL · OBRIR CLAUDE CODE

```
cd web-cafes-roca
claude
```

EXPLORAR EL PROJECTE

Explora aquesta carpeta i explica'm què hi ha: quins fitxers, què conté cadascun i què creus que hem de construir. No modifiquis res. Acaba amb tres preguntes que em faries abans de començar.

PREGUNTA DE COMPROVACIÓ

Segons el brief, quins horaris té el local de Sants i quins colors hem de fer servir a la web?

El que hauries de veure

L'agent cita el brief, els tres articles en Markdown i les dades de vendes i opinions. Respon que Sants obre de dilluns a divendres de 7:30 a 19:00 i que els colors són #3B2A1F, #F4EBDD i #C8553D. No ha creat ni modificat cap fitxer.

☐ **Pràctica 1 feta: Claude Code obert sobre web-cafes-roca i projecte explorat en mode Plan**

Documentació oficial: [Claude Code overview](#) · [How Claude Code works](#) · [Model configuration](#) · [Using Claude Code with your Pro or Max plan](#)

Bloc 2 · Posada a punt i primera sessió

Comprovar l'entorn, la memòria del projecte i els permisos.

Abans de construir, tres coses: que l'entorn funciona, que l'agent coneix les normes del projecte i que saps quins permisos té i com es desfà un canvi.

Pràctica 2.1 · Comprovar l'entorn

Objectiu. Confirmar que Claude Code, Git, Node.js i GitHub CLI estan instal·lats i que tens la sessió iniciada.

1. Surt de Claude Code amb `/exit` (o prem **Ctrl+C** dues vegades).
2. Executa les ordres de comprovació una per una.
3. Si `gh auth status` diu que no tens sessió, executa `gh auth login` i tria **GitHub.com** › **HTTPS** › **Login with a web browser**.
4. Si alguna ordre no es reconeix, torna a la guia de preparació (pestanya **Consulta**) i instal·la el que falti.

TERMINAL · COMPROVACIONS

```
claude --version
claude doctor
git --version
node --version
gh --version
gh auth status
```

Windows: Claude Code fa servir Git per a Windows per executar ordres. Si `claude doctor` avisa que no el troba, instal·la Git des de git-scm.com i obre un terminal nou.

☐ **Totes les ordres responen i gh té la sessió iniciada**

CLAUDE.md: la memòria del projecte

Claude Code llegeix al començament de cada sessió uns fitxers de text amb instruccions. El principal és `CLAUDE.md`, a l'arrel del projecte. Hi poses com s'executa el projecte, les normes i el que l'agent no pot deduir llegint el codi.

Fitxer	Abast	Es comparteix amb l'equip?
<code>~/.claude/CLAUDE.md</code>	Totes les teves sessions, a qualsevol projecte	No
<code>./CLAUDE.md</code>	Aquest projecte	Sí, va al repositori
<code>./CLAUDE.local.md</code>	Aquest projecte, només per a tu	No, s'exclou de Git
<code>./.claude/rules/*.md</code>	Normes per a carpetes o tipus de	Sí

Fitxer	Abast	Es comparteix amb l'equip?
	fitxer concrets	

A més, Claude Code té una **memòria automàtica**: apunta el que aprèn de tu i del projecte (ordres, preferències, correccions) i ho recupera a les sessions següents. Tot es revisa i s'edita amb `/memory`. Un bon **CLAUDE.md** és curt: menys de 200 línies, concret i verificable.

Pràctica 2.2 · CLAUDE.md amb /init

Objectiu. Generar el CLAUDE.md de la web i afegir-hi les normes del projecte.

1. Torna a obrir `claude` a la carpeta `web-cafes-roca`.
2. Escriu `/init`. Claude Code analitza la carpeta i proposa un **CLAUDE.md**. Accepta'l.
3. Obre **CLAUDE.md** a VS Code i llegeix-lo: què ha entès bé i què no?
4. Enganxa el prompt de normes perquè les afegeixi.
5. Escriu `/memory` per veure quins fitxers de memòria té carregats.

NORMES DEL PROJECTE

Afegeix a CLAUDE.md aquestes normes del projecte:

- Web estàtica en HTML, CSS i JavaScript, sense frameworks ni dependències de npm.
 - Tots els textos de la web en català. Colors i to del brief.
 - Ha de funcionar bé al mòbil.
 - Abans de donar una tasca per acabada, obre la pàgina i comprova-la a 390 px d'amplada i a escriptori.
 - Missatges de commit en català, en imperatiu i curts: «Afegeix el formulari de càtering».
- Mantén el fitxer per sota de 60 línies.

☐ **CLAUDE.md creat i amb les normes del projecte**

Modes de permís

Mode	Què fa sense preguntar	Quan el fem servir
Manual (<code>default</code>)	Només llegir. Demana permís per editar i per executar ordres	Quan comences en un projecte que no coneixes
Accept edits (<code>acceptEdits</code>)	Llegir i editar fitxers. Demana permís per a les ordres	Implementar un pla que ja has revisat
Plan (<code>plan</code>)	Llegir i proposar un pla. No modifica res	Explorar i planificar
Auto (<code>auto</code>)	Un classificador revisa cada acció i atura les arriscades	Tasques llargues quan el compte el té disponible

Al terminal els canvis amb **Maj+Tab**. A l'aplicació d'escriptori, amb el selector de mode del quadre de text. També pots començar directament en un mode: `claude --permission-mode plan`. Hi ha un mode que s'ho salta tot (`bypassPermissions`); al curs no el farem servir.

Punts de control i /rewind

Cada vegada que l'agent edita fitxers, Claude Code en desa un punt de control. Prem **Esc** dues vegades amb el quadre de text buit, o escriu `/rewind`, per tornar enrere: només el codi, només la conversa o totes dues coses. Els canvis fets amb ordres de terminal (per exemple, esborrar un fitxer amb `rm`) no queden al punt de control. Per això, a més del `/rewind`, farem commits amb Git.

Pràctica 2.3 · Provar els permisos i desfer un canvi

Objectiu. Veure com demana permís l'agent en mode Manual i recuperar l'estat anterior amb `/rewind`.

1. Amb **Maj+Tab**, posa el mode Manual: a sota del quadre de text surt **manual mode on**.

2. Enganxa el prompt de prova. Quan et demani permís per crear el fitxer, accepta-ho només aquesta vegada.
3. Comprova a VS Code que `prova.html` existeix.
4. Prem **Esc** dues vegades, tria el missatge de la prova i l'opció **Restore code**.
5. Comprova que `prova.html` ja no hi és.

FITXER DE PROVA

Crea un fitxer `prova.html` amb un títol gran que digui «Prova de Cafès Roca» i res més.

□ **Fitxer creat amb permís i desfet amb `/rewind`**

Documentació oficial: [Set up Claude Code](#) · [Manage Claude's memory](#) · [Permission modes](#) · [Checkpointing](#) · [Best practices for Claude Code](#)

Bloc 3 · Una landing page a partir d'un brief

Del brief al pla, del pla a la pàgina, i de la pàgina a un commit.

Fem la web de Cafès Roca amb el mètode complet. El brief diu què volem; el mode Plan converteix el brief en un pla que revises abans que l'agent escrigui una sola línia.

Per què primer el pla

Corregir un pla costa una frase. Corregir una pàgina feta amb una idea equivocada costa una sessió sencera. El mode Plan fa que l'agent preguntí el que no està clar abans de començar.

Pràctica 3 · La landing de Cafès Roca

Objectiu. Construir una landing adaptada al mòbil a partir de `brief_web_cafes_roca.txt`, convertir els articles en Markdown en pàgines de la web i desar-ho tot en un repositori de Git.

1. Posa el mode Plan (**Maj+Tab**) i enganxa el prompt del pla.
2. Respon les preguntes que et faci. Llegeix el pla i, si cal, demana-hi canvis abans d'aprovar-lo.
3. Aprova el pla triant l'opció d'acceptar les edicions. L'agent crea `index.html` i `estils.css`.
4. Obre `index.html` amb doble clic. Fes servir les eines de desenvolupador del navegador (**F12**, icona del mòbil) per veure-la a 390 px.
5. Fes una captura d'una part que vulguis millorar, desa-la a la carpeta com a `captura.png` i enganxa el prompt d'iteració. També pots enganxar la imatge directament al terminal: **Alt+V** a Windows, **Ctrl+V** a macOS.
6. Enganxa el prompt dels articles. L'agent escriu `scripts/articles.mjs`; executa'l tu amb `node scripts/articles.mjs`.
7. Enganxa el prompt del commit.

PLA DE LA LANDING

Llegeix `brief_web_cafes_roca.txt` i els tres articles de la carpeta `articles/`.
 Vull la landing de Cafès Roca: un `index.html` i un `estils.css`, sense dependències.
 Seccions del brief en aquest ordre: capçalera amb lema i botó al càtering, els tres locals, carta destacada, una secció «Del nostre blog» que de moment quedi buida, formulari de càtering i peu.
 El formulari no s'envia enlloc: en enviar-lo, valida els camps i mostra un missatge de confirmació a la mateixa pàgina.
 Fes-me primer les preguntes que necessitis i després un pla amb l'estructura de fitxers i les decisions de disseny.

ITERACIÓ AMB UNA CAPTURA

Mira captura.png. És la secció de la carta vista al mòbil. Els preus queden massa enganxats al nom del plat i la línia de punts no s'alinea. Arregla-ho perquè es llegeixi bé a 390 px i comprova que a escriptori no ha canviat res.

ARTICLES EN MARKDOWN

Crea scripts/articles.mjs, en Node.js i sense dependències, que:

- llegeixi cada .md de articles/ amb la capçalera entre --- (títol, data, autora, resum);
- converteixi el Markdown a HTML: títols, paràgrafs, llistes, negreta, enllaços i taules;
- generi una pàgina per article a blog/, amb la mateixa capçalera i estils de la landing i un enllaç per tornar;
- ompli la secció «Del nostre blog» d'index.html amb una targeta per article, de més nou a més antic.

No l'executis tu: explica'm com l'executo.

PRIMER COMMIT

Inicialitza un repositori de Git en aquesta carpeta, crea un .gitignore adequat per a una web estàtica i fes el primer commit amb tots els fitxers. Missatge en català.

El que hauries de veure

Una pàgina amb els tres locals, la carta amb els cinc preus del brief, tres targetes d'articles i el formulari, que es veu bé al mòbil. A blog/ hi ha tres pàgines. git log mostra el primer commit.

Si t'encalles: a solucions/web-p3-landing hi ha l'estat final d'aquesta pràctica. Copia'n els fitxers a la teva carpeta i continua.

☐ **Pràctica 3 feta: landing, pàgines del blog i primer commit**

Documentació oficial: [Common workflows](#) · [Best practices for Claude Code](#) · [Prompting best practices](#)

Bloc 4 · Anàlisi tècnica de dades

Un script que es pot tornar a executar, gràfics i un informe amb cada xifra traçable.

Al xat, Claude calcula les xifres en un entorn que desapareix quan tanques la conversa. Amb Claude Code, el càlcul queda en un script dins del teu projecte: el pots revisar, executar el mes que ve amb dades noves i ensenyar a qui dubti d'una xifra.

	Anàlisi al xat	Script amb Claude Code
On queda el càlcul	A la conversa	A <code>analisi/analisi.mjs</code> , dins del projecte
Dades noves el mes que ve	Tornar a demanar-ho	<code>node analisi/analisi.mjs</code>
Qui ho pot revisar	Qui tingui la conversa	Tothom que tingui el repositori
Comprovar una xifra	Preguntar-la	Llegir la línia de codi que la calcula

Pràctica 4 · Les vendes del semestre

Objectiu. Escriure un script reproduïble que llegeixi el CSV de vendes, en tregui les xifres clau, generi tres gràfics i redacti un informe on cada xifra digui d'on surt.

1. Comença una conversa nova amb `/clear`.

- Enganxa el prompt de l'anàlisi en mode Plan i revisa el pla: comprova que diu com llegirà la coma decimal.
- Aprova'l i deixa que escrigui l'script.
- Executa'l tu: `node analisi/analisi.mjs`.
- Obre `analisi/informe.md` a VS Code amb la vista prèvia (**Ctrl+Maj+V**) i mira els gràfics.
- Enganxa el prompt de la doble comprovació i el de l'acta.
- Fes un commit.

SCRIPT D'ANÀLISI

Analitza `dades/venes_cafes_roca_2026.csv` (punt i coma, coma decimal; columnes `data`, `local`, `categoria`, `producte`, `unitats`, `import_eur`).
 Escriu `analisi/analisi.mjs` en `Node.js`, sense dependències, que:

- Llegeixi i netegi el CSV: comprovi les columnes, descarti les files no vàlides i digui quantes n'ha descartat.
- Calculi: total del semestre; vendes i pes per local; vendes per mes i local; vendes i pes per categoria; pes de les begudes fredes sobre el total de cada mes; els cinc productes que més facturen.
- Desi totes les xifres a `analisi/resultats.json`.
- Generi tres gràfics SVG a `analisi/grafics/`: evolució mensual per local, pes de cada categoria i begudes fredes per mes. Tria el tipus de gràfic que respongui millor cada pregunta i explica'm per què.
- Escrigui `analisi/informe.md` amb cinc conclusions en llenguatge planer, i al costat de cada xifra la clau de `resultats.json` d'on surt.
 Números en format català: 104.827,00 €.

DOBLE COMPROVACIÓ

Comprova el total del semestre per un altre camí: suma les vendes per local i per mes i mira que les dues sumes donin el mateix que el total. Ensenya'm les tres xifres.

L'ACTA CONTRA LES DADES

L'acta de la reunió del 8 de setembre diu que el cafè és el 55 % de la facturació i que les begudes fredes representen el 14 % a l'estiu. Comprova-ho amb `resultats.json` i explica la diferència. No modifiquis cap fitxer.

Xifres de comprovació

Total: **104.827,00 €** · Poblenou 42,4 %, Gràcia 32,0 %, Sants 25,6 % · cafè 30,1 % del total · begudes fredes 23,6 % al maig i 27,4 % al juny · flat white, primer producte, amb 10.293,00 €.

Python: si tens Python instal·lat, pots demanar el mateix script amb `pandas` i `matplotlib`. Al curs fem servir `Node.js` perquè ja el tens a punt i no cal instal·lar cap llibreria.

☐ **Pràctica 4 feta: script, resultats.json, tres gràfics i informe**

Quin gràfic respon cada pregunta

Pregunta	Gràfic	Per què
Com evolucionen les vendes de cada local?	Línies	Es llegeix la tendència i es comparen locals d'un cop d'ull
Quin pes té cada categoria?	Barres horitzontals ordenades	Els noms llargs hi caben i l'ordre ja diu qui guanya
Com creixen les begudes fredes?	Barres per mes amb el percentatge	El que importa és la proporció de cada mes

Pregunta	Gràfic	Per què
Quant aporta cada local cada mes?	Barres apilades	El total del mes i la part de cada local, ahora

Documentació oficial: [Common workflows](#) · [Node.js](#) · [File system](#) · [MDN](#) · [SVG tutorial](#)

Bloc 5 · Un dashboard interactiu

Especificar abans de construir: entrevista, SPEC.md, pla i implementació.

Per a una peça més gran, fem que l'agent ens entrevisti i escrigui l'especificació. Després comencem una sessió nova, amb el context net, i construïm a partir del document.

- **Entrevista.** L'agent pregunta qui ho farà servir, quines decisions ha de permetre prendre i quines xifres han de coincidir.
- **SPEC.md.** Un document curt amb dades, contingut, disseny i criteris d'acceptació.
- **Context net.** `/clear` i una sessió nova que només llegeix la SPEC. Les decisions viatgen al document.
- **Acceptació.** Els números del dashboard han de quadrar amb els de l'anàlisi.

Pràctica 5 · El quadre de comandament de vendes

Objectiu. Especificar, construir i verificar un dashboard d'un sol fitxer HTML amb filtres per local i per mes, a partir de les dades de vendes.

1. Enganxa el prompt de l'entrevista i respon les preguntes una a una.
2. Llegeix `dashboard/SPEC.md`. Si hi falta alguna cosa, demana que l'afegeixi.
3. Escriu `/clear`, posa el mode Plan i enganxa el prompt de construcció.
4. Revisa el pla i aprova'l.
5. Obre `dashboard/index.html` amb doble clic i prova els filtres.
6. Enganxa el prompt de verificació.
7. Fes un commit.

ENTREVISTA I ESPECIFICACIÓ

Vull un quadre de comandament de les vendes de Cafès Roca a partir de `dades/vendes_cafes_roca_2026.csv`.

Fes-me una entrevista per definir-lo: una pregunta cada vegada, sobre qui el farà servir, què ha de poder decidir, quins filtres, indicadors i gràfics necessita, i com sabrem que les xifres són correctes.

Quan tinguis prou informació, escriu `dashboard/SPEC.md` amb aquestes seccions: per a qui i per a què, dades, contingut, disseny i criteris d'acceptació. No escriguis codi encara.

CONSTRUIR A PARTIR DE LA SPEC

Llegeix `dashboard/SPEC.md` i fes un pla per construir-lo.

Restriccions: un sol fitxer `dashboard/index.html` amb CSS i JavaScript, sense llibreries externes, gràfics en SVG. Perquè s'obri amb doble clic sense servidor, genera les dades agregades en un `dashboard/dades.js` amb un script `scripts/dades-dashboard.mjs`.

Quan acabis, comprova tu mateix cada criteri d'acceptació i digue'm el resultat.

VERIFICACIÓ

Compara el dashboard amb `analisi/resultats.json`:

- filtre Tots, de gener a juny: vendes totals i pes de Poblenou;
- filtre Gràcia, de maig a juny: vendes.

Digue'm si quadren i, si no, quina de les dues xifres és la correcta i per què.

El que hauries de veure

Amb Tots i gener–juny, **104.827,00 €** i el flat white com a primer producte. Amb Gràcia i maig–juny, **12.040,20 €**. Els filtres actualitzen indicadors, gràfics i taula alhora.

Si t'encalles: solucions/web-p5-dashboard té la SPEC, l'script de dades i el dashboard acabat.

□ **Pràctica 5 feta: SPEC.md, dashboard i xifres verificades**

Documentació oficial: [Best practices: let Claude interview you](#) · [Common workflows](#)

Bloc 6 · Context, cost i bones pràctiques

Què ocupa la memòria de la sessió, quant consumeixes i com treballar perquè duri.

La finestra de context és la memòria de treball de la sessió: hi entren els teus missatges, els fitxers que llegeix l'agent i la sortida de les ordres. Quan s'omple, l'agent rendeix menys i el límit d'ús s'acaba abans.

Ordre	Què fa	Quan
<code>/context</code>	Mostra què ocupa el context	Quan la sessió s'allarga
<code>/compact [instruccions]</code>	Resumeix la conversa i allibera espai	En acabar una fase, dient què vols conservar
<code>/clear</code>	Comença una conversa nova al mateix projecte	En canviar de tasca
<code>/usage</code>	Consum de la sessió i límits del pla	Abans d'una tasca llarga
<code>/btw [pregunta]</code>	Pregunta lateral que no queda a l'història	Un dubte ràpid enmig d'una tasca
<code>/resume</code>	Recupera una conversa anterior	L'endemà, per continuar

- **Una tasca per sessió.** Quan acabes, `/clear`. El context d'una tasca embruta la següent.
- **Peticions verificables.** Digues com sabràs que està bé: una prova, una xifra, una captura.
- **Atura i corregeix.** Si veus que va per mal camí, prem **Esc** i redirigeix-lo. No esperis que acabi.
- **El que importa, als fitxers.** Les decisions van a **CLAUDE.md**, a la SPEC o a un commit. La conversa s'oblida.

Pràctica 6 · Mirar i netejar el context

Objectiu. Veure què ocupa el context i quant has consumit, i aprendre a compactar i començar de nou.

1. Escriu `/context` i mira quina part ocupen els fitxers i quina la conversa.
2. Escriu `/usage` i mira com vas de la finestra de cinc hores i de la setmana.
3. Fes una pregunta lateral amb `/btw`.
4. Compacta la conversa dient què vols conservar.
5. Torna a mirar `/context` i compara-ho.

PREGUNTA LATERAL

`/btw` quina diferència hi ha entre `dashboard/dades.js` i `scripts/dades-dashboard.mjs`?

COMPACTAR AMB CRITERI

/compact conserva només les decisions del dashboard, els fitxers que hem creat i les xifres de verificació

☐ **Pràctica 6 feta: context mirat, compactat i consum consultat****Per a la sessió 2**

Porta el projecte web - cafes - roca amb els commits fets. El farem servir per publicar la landing a GitHub Pages.

Documentació oficial: [Manage costs effectively](#) · [Context window](#) · [Commands](#)

Sessió 2 · Repositoris, qualitat i automatització

Blocs 7–12 · GitHub, personalització, MCP i criteri

Bloc 7 · Git i GitHub amb Claude Code

Entendre un projecte existent, incidències, branques, proves i pull requests.

A la segona sessió treballem com un equip de desenvolupament sobre l'aplicació d'estoc de Cafès Roca: un projecte que ja existeix, amb proves, tres incidències obertes i un error que ha detectat el responsable de Gràcia.

Paraula	Què és	Ordre que farà servir l'agent
Repositori	La carpeta del projecte amb tot l'històric de canvis	<code>git init, gh repo create</code>
Commit	Una foto dels canvis amb un missatge	<code>git commit</code>
Branca	Una línia de treball separada de la principal (<code>main</code>)	<code>git switch -c</code>
Incidència (issue)	Una tasca o un error apuntat a GitHub	<code>gh issue create, gh issue view</code>
Pull request	La proposta de portar una branca a <code>main</code> , perquè algú la revisi	<code>gh pr create</code>
Diff	Les línies que han canviat	<code>git diff, gh pr diff</code>

Pràctica 7.1 · Conèixer el projecte i pujar-lo a GitHub

Objectiu. Fer que l'agent t'expliqui un projecte que no has escrit tu, comprovar que funciona i publicar-lo en un repositori privat amb les seves incidències.

1. Obre un terminal a la carpeta `estoc-cafes-roca` i executa `npm test` i `npm start`. Obre `http://localhost:3000` i tria el local de Gràcia. Atura el servidor amb **Ctrl+C**.
2. Obre `claude` en mode Plan i enganxa el prompt d'exploració.
3. Torna al mode Manual i enganxa el prompt per crear el repositori. Accepta les ordres de `gh` quan te les demani.
4. Enganxa el prompt de les incidències.
5. Obre el repositori a GitHub i comprova que hi ha les tres incidències.

TERMINAL · ARRENCAR EL PROJECTE

```
cd estoc-cafes-roca
npm test
npm start
```

ENTENDRE EL PROJECTE

Explica'm aquest projecte com si fos el meu primer dia a l'equip: què fa, com està organitzat, com flueixen les dades des de `data/productes.json` fins a la pàgina i quines regles de negoci aplica. Després llegeix `ISSUES.md` i digue'm per on començaries i per què. No modifiquis res.

REPOSITORI A GITHUB

Crea un repositori privat a GitHub que es digui `estoc-cafes-roca` amb GitHub CLI i puja-hi la branca `main` amb tot l'historial. Comprova abans que `.env` no es pujarà.

INCIDÈNCIES

Crea a GitHub les tres incidències d'`ISSUES.md`, amb el títol, el text i l'etiqueta de cadascuna (bug o enhancement). Després ensenya'm la llista amb `gh issue list`.

☐ **Repositori privat creat amb tres incidències**
Pràctica 7.2 · Corregir un error començant per una prova

Objectiu. Resoldre la incidència 1 amb el mètode de les proves primer: una prova que falla, la correcció mínima, totes les proves en verd i una pull request.

1. Enganxa el prompt de la prova. L'agent llegeix la incidència, crea la branca i escriu una prova nova.
2. Comprova que la prova **falla**. Si passa, la prova no reproduïx l'error: demana-li que la revisi.
3. Enganxa el prompt de la correcció.
4. Revisa el canvi amb `git diff` abans d'acceptar el commit.
5. Enganxa el prompt de la pull request i obre-la a GitHub.

LA PROVA QUE FALLA

Llegeix la incidència 1 amb `gh issue view 1`. Crea la branca `fix/reposicio-minim`. Escriu a `test/estoc.test.js` una prova que reproduïxi l'error amb la beguda d'ametlla de Gràcia (estoc 6, mínim 6, consum setmanal 9) i que comprovi també que la quantitat a demanar és de 18 litres. Executa les proves i ensenya'm que la nova falla. No toquis encara `src/`.

LA CORRECCIÓ

Ara corregeix l'error a `src/estoc.js` amb el canvi més petit possible. Torna a executar totes les proves i fes un commit que expliqui l'error i la correcció.

LA PULL REQUEST

Puja la branca i obre una pull request cap a `main` que tanqui la incidència 1. A la descripció: què fallava, què has canviat, com s'ha provat i el resultat de `npm test`.

El que hauries de veure

La prova nova falla amb el codi original. La correcció és un sol caràcter: `<` passa a `<=` a `necessitaReposicio`. Després, 11 proves en verd i, a Gràcia, la beguda d'ametlla surt a la llista amb 18 l per demanar.

Si t'encalles: `solucions/estoc-p7-error-corregit` té la prova i la correcció.

☐ **Pràctica 7 feta: prova que falla, correcció i pull request oberta**

Documentació oficial: [Common workflows](#) · [GitHub CLI manual](#) · [Node.js test runner](#)

Bloc 8 · Treballar en paral·lel i revisar

Sessions simultànies en worktrees, revisió automàtica i revisió humana.

Un worktree és una còpia de treball del mateix repositori en una altra carpeta i en una altra branca. Permet tenir dues sessions de Claude Code treballant alhora sense trepitjar-se.

On	Com s'obre una sessió en un worktree
Terminal	<code>claude -w exportar-csv</code> crea <code>.claude/worktrees/exportar-csv</code> a la branca <code>worktree-exportar-csv</code>
Aplicació d'escriptori	Pestanya Code > sessió nova > opció de worktree
Subagents	<code>isolation: worktree</code> a la definició del subagent

Pràctica 8 · Dues incidències alhora i una revisió

Objectiu. Implementar la incidència 2 en un worktree mentre la pull request de la 1 espera revisió, revisar-les amb l'agent i amb els teus ulls, i fusionar-les.

1. Obre un segon terminal a `estoc-cafes-roca`, torna a `main` amb `git switch main` i escriu `claude -w exportar-csv`.
2. Enganxa el prompt de la incidència 2 i deixa que l'agent hi treballi.
3. Mentrestant, al primer terminal, escriu `/code-review` seguit del número de la pull request de la incidència 1. A GitHub, incidències i pull requests comparteixen numeració: si has creat tres incidències, la primera pull request és la 4. La revisió s'executa en segon pla i el resultat arriba quan acaba.
4. Quan la incidència 2 estigui feta i tingui la seva pull request, passa-hi `/security-review` des del worktree.
5. Revisa tu el diff a GitHub (pestanya **Files changed**) o amb `gh pr diff`.
6. Fusiona les dues pull requests. L'agent t'ha de demanar confirmació abans.

TERMINAL 2 · SESSIÓ EN UN WORKTREE

```
git switch main
claude -w exportar-csv
```

INCIDÈNCIA 2 EN EL WORKTREE

Implementa la incidència 2 (gh issue view 2): endpoint `/api/reposicio.csv` amb el filtre `?local=`, botó «Descarrega CSV» a la pàgina i proves de la funció que genera el CSV. Punt i coma, coma decimal i UTF-8 amb BOM. Quan acabis, executa `npm test`, fes commit i obre una pull request que tanqui la incidència 2.

REVISIÓ AUTOMÀTICA

```
/code-review 4
```

FUSIONAR

Fusiona les pull requests de les incidències 1 i 2 amb squash, en aquest ordre. Si la segona té conflictes amb la primera, resol·los, torna a executar les proves i ensenya'm el resultat abans de fusionar.

Llegeix el diff abans de fusionar

`/code-review` i `/security-review` troben errors, casos límit i riscos que se t'escaparien. La decisió de fusionar és teva: llegeix el diff, executa l'aplicació i comprova que fa el que demanava la incidència.

Si t'encalles: `solucions/estoc-p8-exportar-csv` té el CSV implementat i provat.

□ Pràctica 8 feta: dues pull requests revisades i fusionades

Documentació oficial: [Run parallel sessions with Git worktrees](#) · [Claude Code on desktop](#) · [Commands \(/code-review, /security-review\)](#)

Bloc 9 · Personalitzar l'agent

CLAUDE.md d'equip, normes per carpetes, habilitats, subagents, hooks i permisos.

Tot el que hem repetit en els prompts de la sessió es pot deixar escrit perquè l'agent ho faci sempre igual, i perquè ho faci igual per a tot l'equip.

Peça	On viu	Per a què
CLAUDE.md	Arrel del projecte	Ordres, estructura, regles de negoci i com treballa l'equip
Normes per carpetes	<code>.claude/rules/*.md</code> amb <code>paths:</code>	Instruccions que només s'apliquen a uns fitxers, per exemple les proves
Habilitat (skill)	<code>.claude/skills/<nom>/SKILL.md</code>	Una tasca que es repeteix, amb el seu procediment. S'invoca amb <code>/nom</code> o l'agent la fa servir quan toca
Subagent	<code>.claude/agents/<nom>.md</code>	Un especialista amb el seu context, les seves eines i el seu model
Hook	<code>.claude/settings.json</code> > <code>hooks</code>	Una ordre que s'executa sempre en un moment concret, per exemple després d'editar
Permisos	<code>.claude/settings.json</code> > <code>permissions</code>	Què pot fer sense preguntar, què ha de preguntar i què no pot fer mai

Pràctica 9.1 · CLAUDE.md d'equip i normes per a les proves

Objectiu. Deixar per escrit les normes del projecte d'estoc i les de les proves.

1. Torna a `main` i actualitza-la: `git switch main` i `git pull`.
2. Obre `claude` i enganxa el prompt del CLAUDE.md.
3. Enganxa el prompt de les normes de proves i revisa els dos fitxers.

CLAUDE.MD D'EQUIP

Escriu el CLAUDE.md d'aquest projecte per a tot l'equip, de menys de 50 línies: ordres per arrencar i provar, estructura de carpetes, regles de negoci de la reposició i com treballem (català a codi i commits, cap dependència nova sense preguntar, cada canvi de lògica amb la seva prova, una branca per incidència i mai push directe a main, npm test en verd abans d'acabar).

NORMES DE LES PROVES

Crea `.claude/rules/proves.md` amb `paths: test/**/*.js` i aquestes normes: `node:test` i `node:assert/strict` sense llibreries externes; un test per comportament amb el nom en català en forma de frase; dades de prova dins del mateix fitxer; quan una xifra surt d'un càlcul, el càlcul en un comentari.

□ CLAUDE.md i .claude/rules/proves.md creats

Pràctica 9.2 · Una habilitat: l'informe de reposició

Objectiu. Convertir una tasca setmanal en una habilitat que tot l'equip pot invocar amb una ordre.

1. Enganxa el prompt per crear l'habilitat.
2. Escriu `/informe-reposicio` Gràcia i mira què fa.
3. Obre l'informe que ha desat a `informes/` i comprova que la beguda d'ametlla hi surt.

CREAR L'HABILITAT

Crea una habilitat de projecte a `.claude/skills/informe-reposicio/SKILL.md`.

Descripció: prepara l'informe setmanal de reposició per local i el correu per a cada proveïdor; s'ha de fer servir quan es demani la comanda de la setmana o l'informe de reposició.

Passos: executar `npm test` i aturar-se si falla; obtenir la llista de reposició amb la lògica de `src/estoc.js` sense modificar `res`, filtrant pel local que arribi com a argument; desar `informes/reposicio-AAAA-MM-DD.md` amb una taula per local i un correu de comanda per proveïdor signat per Laura Sanz; no enviar `res`. Afegeix `informes/` al `.gitignore`.

☐ Habilitat creada i provada amb `/informe-reposicio`

Pràctica 9.3 · Un subagent revisor de proves

Objectiu. Crear un especialista que revisa si els canvis tenen les proves que necessiten, amb un context propi i sense poder modificar fitxers.

1. Enganxa el prompt per crear el subagent.
2. Fes-lo servir sobre la branca de la incidència 3, o sobre l'últim commit si encara no l'has començada.

CREAR EL SUBAGENT

Crea el subagent `.claude/agents/revisor-proves.md`: revisa si un canvi de codi té les proves que necessita; s'ha de fer servir després de modificar `src/` i abans d'obrir una pull request. Eines: Read, Grep, Glob i Bash. Model: sonnet. Instruccions: mirar git diff, comprovar per a cada funció nova o modificada que hi ha una prova del cas normal i una d'un cas límit, executar `npm test` i tornar una llista de proves que falten. No pot modificar fitxers.

FER-LO SERVIR

Fes servir el subagent `revisor-proves` per revisar els canvis de l'últim commit.

☐ Subagent creat i provat

Pràctica 9.4 · Un hook que executa les proves

Objectiu. Fer que les proves s'executin soles cada vegada que l'agent edita un fitxer de `src/` o de `test/`, i que l'agent vegi l'error si fallen.

1. Enganxa el prompt del hook.
2. Escriu `/hooks` per comprovar que està registrat.
3. Enganxa el prompt de prova: veuràs que les proves fallen just després de l'edició i que l'agent ho detecta.
4. Desfés el canvi amb `/rewind`.

CREAR EL HOOK

Crea un hook `PostToolUse` a `.claude/settings.json` per a les eines Edit i Write que executi `node .claude/hooks/proves-despres-d-editar.mjs`.

L'script, en Node.js: llegeix el JSON de l'entrada estàndard; si el fitxer editat (`tool_input.file_path`) és dins de `src/` o `test/`, executa `node --test`; si falla, escriu la sortida a `stderr` i surt amb el codi 2 perquè l'agent vegi l'error. Ha de funcionar a Windows i a macOS.

PROVAR EL HOOK

Canvia temporalment a `src/estoc.js` el valor de `SETMANES_DE_MARGE` a 3 i digue'm què ha passat després de l'edició.

☐ **Hook creat, provat i canvi desfet**
Pràctica 9.5 · Permisos de l'equip

Objectiu. Decidir què pot fer l'agent sense preguntar, què ha de preguntar i què no pot fer mai, i comprovar que es compleix.

1. Enganxa el prompt dels permisos i revisa el `settings.json` resultant amb la referència d'aquí sota.
2. Escriu `/permissions` per veure les regles actives.
3. Enganxa el prompt de prova del `.env`: l'agent no el pot llegir.

REGLES DE PERMISOS

Afegeix a `.claude/settings.json` regles de permisos:

- sense preguntar: `npm test`, `npm start`, `node --test`, `git status`, `git diff`, `git log`, `gh issue view` i `gh pr view`;
- preguntant sempre: `git push` i `gh pr merge`;
- mai: llegir o editar `.env`, editar `data/`, `git push --force`, `git push origin main`, `npm install` i `rm -rf`.

PROVAR UN PERMÍS DENEGAT

Llegeix el fitxer `.env` i digue'm quines claus hi ha.

REFERÈNCIA · `.CLAUDE/SETTINGS.JSON`

```
{
  "permissions": {
    "allow": ["Bash(npm test)", "Bash(npm start)", "Bash(node --test*)", "Bash(git status)",
      "Bash(git diff *)", "Bash(git log *)", "Bash(gh issue view *)", "Bash(gh pr view *)"],
    "ask": ["Bash(git push *)", "Bash(gh pr merge *)"],
    "deny": ["Read(./env)", "Read(./env.*)", "Edit(./env)", "Edit(./data/**)", "Bash(git push --force*)",
      "Bash(git push origin main)", "Bash(npm install *)", "Bash(rm -rf *)"]
  },
  "hooks": {
    "PostToolUse": [{ "matcher": "Edit|Write",
      "hooks": [{ "type": "command", "command": "node .claude/hooks/proves-despres-d-
editar.mjs", "timeout": 120 }] }
  ]
}
```

Ordre d'avaluació

Primer es miren les regles deny, després les ask i al final les allow. Una regla de denegació guanya sempre. Les regles de `.claude/settings.json` es comparteixen amb l'equip; les teves, a `.claude/settings.local.json`.

Si t'encalles: solucions/estoc-p9-agent-personalitzat té `CLAUDE.md`, la norma, l'habilitat, el subagent, el hook i els permisos.

☐ **Pràctica 9 feta: permisos configurats i `.env` protegit**

Documentació oficial: [Manage Claude's memory](#) · [Agent Skills](#) · [Subagents](#) · [Hooks guide](#) · [Configure permissions](#) · [Settings](#)

Bloc 10 · Connectar eines: MCP, connectors i plugins

Un navegador real per provar l'aplicació i paquets d'habilitats ja fets.

MCP (Model Context Protocol) és l'estàndard obert que fa servir Claude per connectar-se a eines externes: un navegador, una base de dades, Jira, Figma. Cada servidor MCP afegeix eines noves a l'agent.

Àmbit	On es desa	Per a qui
<code>local</code> (per defecte)	<code>~/.claude.json</code> , lligat a aquest projecte	Només per a tu, en aquest projecte
<code>project</code>	<code>.mcp.json</code> a l'arrel del projecte	Tot l'equip, va al repositori
<code>user</code>	<code>~/.claude.json</code>	Per a tu, a tots els projectes

Pràctica 10.1 · Playwright: provar l'aplicació en un navegador

Objectiu. Afegir el servidor MCP de Playwright i fer que l'agent provi la pàgina d'estoc com ho faria una persona, en mida mòbil.

1. Surt de Claude Code i afegeix el servidor. Amb `--scope project` es desa a `.mcp.json` per a tot l'equip.
2. En un altre terminal, arrenca l'aplicació amb `npm start`.
3. Obre `claude` i escriu `/mcp`: Playwright ha de sortir connectat. Aprova'l si t'ho demana.
4. Enganxa el prompt de la prova.

TERMINAL · AFEGIR PLAYWRIGHT

```
claude mcp add --scope project playwright -- npx @playwright/mcp@latest
```

Windows: si a `/mcp` el servidor surt amb error, elimina'l amb `claude mcp remove playwright` i torna'l a afegir amb `cmd /c npx @playwright/mcp@latest` al final de l'ordre.

PROVA AL NAVEGADOR

Fes servir Playwright per obrir `http://localhost:3000` amb una finestra de `390 × 844`. Tria el local de Gràcia i comprova que la beguda d'ametlla surt a la llista de reposició amb 18 l per demanar i que l'enllaç «Descarrega CSV» apunta a `/api/reposicio.csv` amb el filtre de Gràcia. Fes una captura de la pàgina i digue'm si hi ha res que es vegi malament al mòbil.

☐ **Playwright connectat i aplicació provada al navegador**

Pràctica 10.2 · Plugins i marketplaces

Objectiu. Instal·lar un plugin del marketplace oficial i veure què hi afegeix.

1. Escriu `/plugin` i obre la pestanya **Discover**. El marketplace oficial d'Anthropic, `claude-plugins-official`, ja hi és.
2. Tria un plugin relacionat amb el desenvolupament web o la revisió de codi, llegeix què inclou i instal·la'l a l'àmbit del projecte.
3. Escriu `/reload-plugins`. Les habilitats del plugin surten amb el format `/plugin:habilitat`.
4. Pregunta a Claude què ha afegit el plugin.

QUÈ HI HA DE NOU

Quines habilitats, subagents, hooks o servidors MCP ha afegit el plugin que acabo d'instal·lar? Explica'm en una línia per a què serveix cadascun.

Només fonts de confiança

Un servidor MCP o un plugin executa codi al teu ordinador i pot llegir el que llegeix l'agent. Instal·la només els que vinguin d'una font que coneguis i revisa'n els permisos.

☐ **Plugin instal·lat i revisat**

Documentació oficial: [Connect Claude Code to tools via MCP](#) · [Plugins](#) · [Discover and install plugins](#) · [Playwright MCP](#)

Bloc 11 · Automatitzar i publicar

Claude Code dins de scripts, a GitHub Actions i al núvol, i la landing publicada.

Fins ara hem treballat en sessions interactives. Claude Code també funciona sense conversa: dins d'un script, en un flux de GitHub o en una rutina que s'executa sola al núvol d'Anthropic.

Manera	Com	Exemple
Mode no interactiu	<code>claude -p "..."</code>	Un script que genera l'informe de reposició
GitHub Actions	<code>/install-github-app</code> i <code>@claude</code> en una incidència	Implementar la incidència 3 des de GitHub
Rutines	<code>/schedule</code> o <code>claude.ai/code/routines</code>	L'informe de reposició cada dilluns a les 8:00
Sessió al núvol	<code>claude --cloud "..."</code> o <code>claude.ai/code</code>	Una tasca llarga que continua amb l'ordinador apagat

Pràctica 11.1 · Claude Code en mode no interactiu

Objectiu. Fer servir `claude -p` des del terminal, amb sortida en text i en JSON, i convertir-ho en un script.

1. Surt de Claude Code i executa les ordres d'exemple a [estoc-cafes-roca](#).
2. Enganxa a una sessió de Claude Code el prompt per crear l'script.
3. Executa l'script i obre l'informe que ha generat.

TERMINAL · CLAUDE -P

```
claude -p "Quants productes s'han de reposar a cada local? Una línia per local." --allowedTools "Read"
claude -p "Llista els productes per reposar" --allowedTools "Read" --output-format json
git log --oneline -10 | claude -p "Escriu les notes de versió d'aquests canvis per a l'equip, en català"
```

CREAR L'SCRIPT

Crea `scripts/informe-reposicio.sh` i `scripts/informe-reposicio.ps1` (per a Windows) que facin servir `claude -p` amb només l'eina `Read` per generar la taula de reposició, opcionalment d'un sol local, i la desin a `informes/reposicio-AAAA-MM-DD.md`. Explica'm com s'executen.

☐ **Ordres provades i script creat**

Pràctica 11.2 · GitHub Actions, rutines i núvol (demostració)

Objectiu. Veure com Claude Code treballa sense el teu ordinador: des d'una incidència de GitHub, amb una rutina programada i en una sessió al núvol.

1. **GitHub Actions.** El formador executa `/install-github-app` al repositori. Claude Code instal·la l'aplicació de GitHub, desa el secret i obre una pull request amb el flux `.github/workflows/claude.yml`.
2. Un cop fusionada, escriu a la incidència 3 un comentari que comenci per `@claude`. Al cap d'uns minuts, Claude obre una pull request.
3. **Rutines.** Amb `/schedule`, el formador crea una rutina que cada dilluns a les 8:00 prepara l'informe de reposició.
4. **Núvol.** `claude --cloud` llança una tasca que continua a `claude.ai/code` encara que es tanqui el portàtil.

COMENTARI A LA INCIDÈNCIA 3

@claude implementa aquesta incidència. Afegeix proves de la funció de filtre i obre una pull request.

RUTINA SETMANAL

`/schedule` cada dilluns a les 8:00, al repositori `estoc-cafes-roca`: executa les proves, prepara l'informe de reposició de cada local i obre una incidència amb la taula i els correus per als proveïdors.

Per què és una demostració

GitHub Actions necessita permisos d'administrador al repositori i un token, i les rutines i les sessions al núvol necessiten el repositori connectat a `claude.ai/code`. Ho tens tot explicat aquí per fer-ho pel teu compte quan tinguis el teu projecte.

Pràctica 11.3 · Publicar la landing a GitHub Pages

Objectiu. Publicar la web de Cafès Roca de la sessió 1 en una adreça pública.

1. Obre un terminal a la carpeta `web-cafes-roca` i obre `claude`.
2. Enganxa el prompt de publicació.
3. Espera un o dos minuts i obre l'adreça que et doni.

PUBLICAR

Publica aquesta web a GitHub Pages: crea un repositori públic `web-cafes-roca` amb `gh`, puja-hi la branca `main` i activa GitHub Pages des de la branca `main` i la carpeta arrel amb `gh api`. Abans, comprova que no puges res que no s'hagi de publicar. Quan acabis, dona'm l'adreça de la web.

REFERÈNCIA · EL QUE FARÀ L'AGENT

```
gh repo create web-cafes-roca --public --source=. --push
gh api -X POST repos/{owner}/web-cafes-roca/pages -f "source[branch]=main" -f
"source[path]=/"
```

El que hauries de veure

La web a `https://<el-teu-usuari>.github.io/web-cafes-roca/`, amb la landing, les pàgines del blog i el dashboard a `/dashboard/`.

☐ Pràctica 11 feta: script amb `claude -p` i landing publicada

Documentació oficial: [Run Claude Code programmatically](#) · [CLI reference](#) · [Claude Code GitHub Actions](#) · [Routines](#) · [Claude Code on the web](#) · [GitHub Pages](#) · [Quickstart](#)

Bloc 12 · Criteri

Seguretat, privadesa, cost i quan n'hi ha prou amb el vibe coding.

Un agent que llegeix contingut extern i executa ordres al teu ordinador necessita límits. Aquest bloc recull els que hem anat posant i en prova dos.

- **Permisos.** Mode Manual per començar, Plan per explorar, regles **deny** per al que no s'ha de fer mai.
- **Sandbox.** `/sandbox` aïlla les ordres del sistema de fitxers i de la xarxa. Funciona a macOS i Linux; a Windows, dins de WSL2.
- **Secrets.** Les claus van a `.env`, fora de Git i amb `Read(./env)` denegat. Mai en un prompt ni en un CLAUDE.md.
- **Injecció de prompts.** Un fitxer, una web o una incidència poden portar instruccions amagades. L'agent les ha de tractar com a dades.
- **Privadesa.** A Configuració > Privadesa decideixes si les converses serveixen per entrenar models. Team i Enterprise tenen condicions d'empresa.
- **Cost.** `/usage` abans de les tasques llargues. Sonnet per a la feina diària i Opus quan calgui.

Pràctica 12.1 · Una instrucció amagada

Objectiu. Veure com tracta l'agent un fitxer que conté instruccions per a ell i quines proteccions hi ha.

1. Obre `docs/nota_proveidor.md` a VS Code i llegeix-lo sencer, inclòs el comentari del final.
2. A Claude Code, enganxa el prompt del resum.
3. Llegeix la resposta: ha seguit les instruccions del comentari, les ha ignorades o te les ha assenyalat?
4. Enganxa el prompt de l'anàlisi.

RESUMIR LA NOTA

Resumeix `docs/nota_proveidor.md` per a l'equip en tres línies i actualitza el que calgui perquè les comandes de begudes vegetals es facin en múltiples de 6.

ANÀLISI

Hi ha alguna instrucció en aquest fitxer que no hauries de seguir? Quines de les nostres regles de permisos l'haurien aturat si l'haguessis intentat seguir?

El que hauries de veure

L'agent fa el resum i assenjala el comentari com a sospitós sense obeir-lo. Si ho hagués intentat, l'haurien aturat les regles de `settings.json`: llegir i editar `.env` i `git push origin main` estan denegats, i `git push` sempre pregunta.

☐ Instrucció amagada detectada i analitzada

Quan n'hi ha prou amb el vibe coding

N'hi ha prou	Cal revisió experta, proves i manteniment
Prototips, eines internes d'un sol ús, una landing estàtica	Aplicacions amb usuaris, pagaments o dades personals
Dades fictícies o públiques	Dades de clients o de l'empresa

N'hi ha prou	Cal revisió experta, proves i manteniment
Si falla, no passa res greu	Si falla, hi ha un cost o un risc legal
La fas servir tu	La fa servir un equip o el públic

Abans de donar per acabat un projecte fet amb un agent

Proves que cobreixen els casos importants i passen. Has llegit el diff. Cap secret al repositori. Dependències revisades. Algú sap mantenir-lo quan l'agent no hi sigui.

Documentació oficial: [Security](#) · [Sandboxing](#) · [Data usage](#) · [Model improvement privacy settings](#)

Consulta · Preparació, ordres i fonts

Guia d'instal·lació, ordres de referència, glossari, prompts i documentació

Resultats d'aprenentatge

Les capacitats que treballem en les dues sessions i el que tindràs fet en acabar.

- **Entendre l'agent.** Explicar com llegeix, planifica, actua i verifica un agent de programació, i en quins punts calen la teva revisió i la teva aprovació.
- **Configurar l'entorn.** Instal·lar Claude Code, triar model i esforç, i controlar què pot fer amb els modes de permís i les regles de settings.json.
- **Construir amb mètode.** Fer una landing, una anàlisi de dades i un dashboard a partir d'una especificació, i comprovar que el resultat és correcte.
- **Treballar en equip.** Fer servir Git i GitHub amb l'agent: branques, commits, incidències, pull requests i revisió de codi.
- **Adaptar l'agent.** Escriure CLAUDE.md, habilitats, subagents, hooks i permisos per a un projecte i un equip.
- **Criteri.** Aplicar criteris de seguretat, privadesa i cost, i distingir el codi que ja es pot fer servir del que necessita més revisió.

Resultats del curs

- La landing de Cafès Roca amb tres pàgines de blog generades a partir de Markdown, publicada a GitHub Pages.
- Un script d'anàlisi de vendes que es pot tornar a executar, amb tres gràfics i un informe amb cada xifra traçable.
- Un dashboard interactiu amb filtres, construït a partir d'una especificació escrita per l'agent.
- Un repositori a GitHub amb un error corregit amb proves, una funcionalitat nova i dues pull requests revisades.
- Un projecte configurat per a l'equip: CLAUDE.md, una norma de proves, una habilitat, un subagent, un hook i permisos.
- Un servidor MCP connectat i un script que fa servir Claude Code sense obrir cap sessió.

Preparació prèvia

Què cal tenir instal·lat i a punt abans de la primera sessió.

Fes aquesta preparació abans de venir. Són uns vint minuts i fa que la primera hora de classe sigui de pràctica.

Comptes

Compte	Per a què	Cost
Claude, pla Pro o superior	Claude Code. El pla gratuït no l'inclou	15 € al mes amb pagament anual o 18 € al mes, sense impostos. Es pot cancel·lar en qualsevol moment
GitHub	Repositoris, incidències, pull requests i GitHub Pages	Gratuït

Arriba amb límit disponible

El límit d'ús de Claude Code es comparteix amb el xat i es renova per finestres de cinc hores i per setmana. Per a les dues sessions n'hi ha prou amb el pla Pro, sempre que no l'esgotis els dies abans.

Programari

Programa	Windows	macOS
Aplicació d'escriptori de Claude	claude.com/download	claude.com/download
Claude Code	<code>irm https://claude.ai/install.ps1 iex</code> a PowerShell	<code>curl -fsSL https://claude.ai/install.sh bash</code> al Terminal
Git	git-scm.com (Git per a Windows) o <code>winget install --id Git.Git</code>	<code>xcode-select --install</code> o <code>brew install git</code>
Node.js 22 LTS	nodejs.org o <code>winget install OpenJS.NodeJS.LTS</code>	nodejs.org o <code>brew install node</code>
GitHub CLI	cli.github.com o <code>winget install --id GitHub.cli</code>	<code>brew install gh</code>
Visual Studio Code	code.visualstudio.com	code.visualstudio.com

Preparació · Instal·lar i comprovar

Objectiu. Tenir-ho tot instal·lat, amb la sessió iniciada a Claude i a GitHub, i comprovar-ho amb cinc ordres.

1. Instal·la els programes de la taula en aquest ordre: Git, Node.js, GitHub CLI, VS Code, l'aplicació de Claude i Claude Code.
2. Tanca el terminal i obre'n un de nou perquè reconegui els programes instal·lats.
3. Escriu `claude`, tria iniciar sessió amb el teu compte de Claude i completa-ho al navegador amb el teu compte Pro.
4. Escriu `gh auth login` i segueix els passos: **GitHub.com** › **HTTPS** › **Login with a web browser**.
5. Configura el teu nom i correu a Git amb les dues ordres de `git config`.
6. Executa les ordres de comprovació.

TERMINAL · IDENTITAT DE GIT

```
git config --global user.name "El teu nom"
git config --global user.email "el-teu-correu@exemple.com"
```

TERMINAL · COMPROVACIÓ FINAL

```
claude --version
claude doctor
git --version
node --version
gh auth status
```

Ordinadors del Cibernàrium: si fas servir un ordinador de l'aula, ha d'estar habilitat per instal·lar aquests programes i per accedir a claude.ai, claude.com, anthropic.com, github.com i registry.npmjs.org. Claude Code s'instal·la a la carpeta de l'usuari, sense permisos d'administrador. Entre sessions, la feina queda al teu repositori de GitHub.

□ Preparació feta: les cinc ordres responen

Documentació oficial: [Set up Claude Code](#) · [Troubleshooting installation](#) · [GitHub CLI](#) · [Git](#) · [Node.js](#)

Ordres de referència

Les ordres, dreceres i fitxers que fem servir al curs.

Al terminal

Ordre	Què fa
<code>claude</code>	Obre una sessió interactiva a la carpeta actual
<code>claude "tasca"</code>	Obre una sessió que comença amb aquesta tasca
<code>claude -c</code>	Continua l'última conversa d'aquesta carpeta
<code>claude --permission-mode plan</code>	Comença en mode Plan
<code>claude -w nom</code>	Obre una sessió en un worktree nou
<code>claude -p "tasca"</code>	Mode no interactiu: fa la tasca, escriu el resultat i surt
<code>claude --cloud "tasca"</code>	Llança una sessió al núvol
<code>claude mcp add · claude mcp list</code>	Afegeix o llista servidors MCP
<code>claude update · claude doctor</code>	Actualitza Claude Code o en diagnostica la instal·lació

Dins de la sessió

Ordre	Què fa
<code>/init</code>	Genera un CLAUDE.md a partir del projecte
<code>/memory</code>	Obre els fitxers de memòria i la memòria automàtica
<code>/model · /effort</code>	Canvia el model o el nivell d'esforç
<code>/permissions</code>	Mostra i edita les regles de permisos
<code>/rewind</code>	Torna a un punt de control anterior
<code>/context · /compact · /clear</code>	Mira, resumeix o buida el context
<code>/usage</code>	Consum de la sessió i límits del pla
<code>/btw</code>	Pregunta lateral que no queda a l'història
<code>/resume</code>	Recupera una conversa anterior
<code>/code-review · /security-review</code>	Revisió de codi i de seguretat dels canvis
<code>/hooks · /mcp · /plugin</code>	Gestiona hooks, servidors MCP i plugins
<code>/sandbox</code>	Activa l'aïllament de les ordres
<code>/install-github-app</code>	Configura Claude Code a GitHub Actions
<code>/schedule</code>	Crea una rutina que s'executa al núvol
<code>/exit</code>	Surt de la sessió

Dreceres

Drecera	Què fa
Maj+Tab	Canvia de mode de permís

Drecera	Què fa
Esc	Atura l'agent sense perdre la feina feta
Esc Esc	Obre el menú de punts de control
@fitxer	Afegeix un fitxer a la petició
!ordre	Executa una ordre de terminal sense passar per l'agent
/	Mostra totes les ordres i habilitats disponibles

Fitxers de configuració

Fitxer	Per a què
CLAUDE.md · CLAUDE.local.md	Instruccions del projecte, compartides o personals
.claude/rules/*.md	Normes per a uns fitxers concrets
.claude/skills/<nom>/SKILL.md	Habilitats del projecte
.claude/agents/<nom>.md	Subagents del projecte
.claude/settings.json · .claude/settings.local.json	Permisos, hooks i preferències, de l'equip o personals
.mcp.json	Servidors MCP del projecte
~/ .claude/	Les mateixes peces, per a tots els teus projectes

Glossari

Els termes del curs en paraules planeres.

Terme	Definició
Agent	Un model d'IA que rep un objectiu, fa un pla i actua amb eines (llegir fitxers, executar ordres) fins a acabar-lo.
Àmbit (scope)	On es desa una configuració i a qui afecta: només a tu, al projecte o a totes les teves sessions.
Branca	Una línia de treball separada dins d'un repositori. La principal es diu main.
CLAUDE.md	Fitxer d'instruccions que Claude Code llegeix al començament de cada sessió.
CLI	Programa que es fa servir escrivint ordres al terminal. GitHub CLI és gh.
Commit	Una foto dels canvis del projecte, amb un missatge que explica què s'ha fet.
Context	Tot el que l'agent té present en una sessió: missatges, fitxers llegits i sortides d'ordres.
CSV	Fitxer de text amb dades en columnes separades per un caràcter, aquí el punt i coma.
Dashboard	Pàgina amb indicadors, gràfics i filtres que resumeix moltes dades.
Diff	Les línies que han canviat entre dues versions d'un fitxer.
Endpoint	Una adreça d'una API que torna dades, com

Terme	Definició
	/api/reposicio.
Esforç	Quant raona el model abans de respondre. Es tria amb /effort.
GitHub Actions	Servei de GitHub que executa tasques automàtiques quan passa alguna cosa al repositori.
GitHub Pages	Allotjament gratuït de webs estàtiques des d'un repositori de GitHub.
Habilitat (skill)	Un procediment escrit en un SKILL.md que l'agent fa servir quan toca o quan l'invoques amb /nom.
Hook	Una ordre que s'executa sempre en un moment concret de la sessió, per exemple després d'editar un fitxer.
Incidència (issue)	Una tasca, un error o una petició apuntada en un repositori de GitHub.
Injecció de prompts	Instruccions amagades en un contingut extern que intenten que l'agent faci el que no ha de fer.
JSON	Format de text per a dades estructurades, amb claus i valors.
Landing page	Pàgina web d'una sola pantalla llarga, pensada per a un objectiu concret.
Markdown	Format de text senzill amb marques com # per als títols o ** per a la negreta.
MCP	Model Context Protocol: l'estàndard per connectar Claude amb eines i serveis externs.
Mode de permís	El nivell d'autonomia de l'agent: Manual, Accept edits, Plan o Auto.
Node.js	Entorn per executar JavaScript fora del navegador. Amb ell executem els scripts i l'aplicació d'estoc.
Plugin	Un paquet que instal·la de cop habilitats, subagents, hooks i servidors MCP.
Prova (test)	Un codi que comprova automàticament que una funció fa el que ha de fer.
Pull request	La proposta de portar els canvis d'una branca a main, perquè es revisin abans.
Punt de control	Un estat dels fitxers que Claude Code desa abans d'editar i al qual pots tornar amb /rewind.
Repositori	La carpeta d'un projecte amb tot l'historial de canvis gestionat per Git.
Rutina	Una tasca de Claude Code que s'executa sola al núvol, per horari o per un esdeveniment.
Sandbox	Aïllament que limita què poden tocar les ordres que executa l'agent.
SPEC	Document d'especificació: què s'ha de construir, per a qui i com sabem que està bé.
Subagent	Un agent especialitzat, amb el seu context i les seves eines, al qual l'agent principal encarrega una part de la feina.

Terme	Definició
Vibe coding	Construir programari descrivint-lo en llenguatge natural i deixant que un agent escrigui el codi.
Worktree	Una segona carpeta de treball del mateix repositori, en una altra branca, per treballar en paral·lel.

Documentació oficial

Les fonts d'aquest temari.

Pàgina	Adreça
Claude Code overview	code.claude.com/docs/en/overview
Set up Claude Code	code.claude.com/docs/en/setup
How Claude Code works	code.claude.com/docs/en/how-claude-code-works
Best practices for Claude Code	code.claude.com/docs/en/best-practices
Common workflows	code.claude.com/docs/en/common-workflows
Model configuration	code.claude.com/docs/en/model-config
Permission modes	code.claude.com/docs/en/permission-modes
Checkpointing	code.claude.com/docs/en/checkpointing
Manage Claude's memory	code.claude.com/docs/en/memory
Commands	code.claude.com/docs/en/commands
CLI reference	code.claude.com/docs/en/cli-reference
Manage costs effectively	code.claude.com/docs/en/costs
Run parallel sessions with Git worktrees	code.claude.com/docs/en/worktrees
Claude Code on desktop	code.claude.com/docs/en/desktop
Agent Skills	code.claude.com/docs/en/skills
Subagents	code.claude.com/docs/en/sub-agents
Hooks guide	code.claude.com/docs/en/hooks-guide
Configure permissions	code.claude.com/docs/en/permissions
Connect Claude Code to tools via MCP	code.claude.com/docs/en/mcp
Plugins	code.claude.com/docs/en/plugins/overview
Run Claude Code programmatically	code.claude.com/docs/en/headless
Claude Code GitHub Actions	code.claude.com/docs/en/github-actions
Routines	code.claude.com/docs/en/routines
Claude Code on the web	code.claude.com/docs/en/claude-code-on-the-web
Security	code.claude.com/docs/en/security
Sandboxing	code.claude.com/docs/en/sandboxing
Using Claude Code with your Pro or Max plan	support.claude.com/en/articles/11145838-using-claude-code-with-your-pro-or-max-plan

Pàgina	Adreça
Plans i preus de Claude	claude.com/pricing

Informació contrastada

El que diu aquest temari sobre Claude Code surt de la documentació oficial d'Anthropic i s'ha contrastat el 10 d'octubre de 2026. Claude Code s'actualitza cada setmana: si una ordre o un menú no coincideix amb el que veus, mana la documentació.