GUÍA PASO A PASO

Un CLAUDE.md corto para tu agente

Dos plantillas de CLAUDE.md cortas, una para Next.js y otra para un tema de Shopify, con qué poner y qué dejar fuera. Después, cómo comprobar que Claude Code las cargó y cuándo compactar o limpiar la sesión para que el agente trabaje con el contexto justo.

PARA	Claude Code en proyectos Next.js o temas de Shopify
PROBADO CON	shopify theme check (Shopify CLI 3.88.1) · Node 22 · Windows 11
ACTUALIZADA	30 de octubre de 2026
DOCUMENTACIÓN	code.claude.com/docs/en/memory
ARTÍCULO	ecomlabs.dev/g/contexto
AUTOR	Luis Morales · @luismoralesdev

Índice

Antes de empezar	Qué carga Claude Code al iniciar
Paso 01	Empieza con /init y recorta
Paso 02	Plantilla para Next.js
Paso 03	Plantilla para un tema de Shopify
Paso 04	Mueve lo específico a una regla
Paso 05	Comprueba que cargó
Paso 06	Durante la sesión
Final	Checklist

ANTES DE EMPEZAR

Qué carga Claude Code al iniciar

Cada sesión de Claude Code empieza con el contexto vacío. Antes de tu primer mensaje carga tus archivos `CLAUDE.md`, la memoria automática, los nombres de las herramientas MCP y las descripciones de las skills. Todo lo que pongas en un `CLAUDE.md` ocupa contexto en cada tarea, por eso conviene que sea corto.

ALCANCE	ARCHIVO	PARA QUÉ
Usuario	`~/.claude/CLAUDE.md`	Tus preferencias en todos los proyectos
Proyecto	`./CLAUDE.md` o `./.claude/CLAUDE.md`	Comandos y reglas del equipo, en git
Local	`./CLAUDE.local.md`	Tus notas de este proyecto, en `.gitignore`
Subcarpeta	`carpeta/CLAUDE.md`	Se carga cuando Claude trabaja en esa carpeta
Regla	`.claude/rules/*.md` con `paths`	Se carga con los archivos que coinciden

Los archivos no se reemplazan: se concatenan, del más general al más cercano a la carpeta donde arrancas. La documentación recomienda menos de 200 líneas por archivo.

PASO 01

Empieza con /init y recorta

En una sesión dentro de tu proyecto, corre `/init`. Claude analiza el código y escribe un primer `CLAUDE.md` con comandos y convenciones; si ya tienes uno, sugiere mejoras en lugar de reemplazarlo. Desde ahí el trabajo es quitar líneas. Para cada una, pregúntate si Claude podría saberlo leyendo el código y si de verdad iría a buscarlo. Si las dos respuestas son sí, bórrala.

SE QUEDA	SE VA
Comandos exactos de lint, tipos y tests	Árbol de carpetas
Código que engaña: un patrón viejo que no hay que copiar	Lista de dependencias
Decisiones que no están en el repo	«Escribe código limpio» y reglas genéricas
Lo que ya corregiste dos veces	Procedimientos largos (van en una skill)

PASO 02

Plantilla para Next.js

Guárdala como `CLAUDE.md` en la raíz del proyecto. Las líneas de Convenciones y Cuidado son ejemplos: cámbialas por las de tu código.

- `@AGENTS.md` importa el archivo que crea `create-next-app` en su configuración por defecto. Si existe un `CLAUDE.md`, Claude Code lee ese y no el `AGENTS.md`, salvo que lo importes. Si no tienes `AGENTS.md`, borra la

línea.

- **Los comandos** salen de los scripts que la documentación de Next.js pone en `package.json`. Antes de guardar, abre tu `package.json` y confirma que existen con esos nombres.
- **Un import no ahorra contexto:** el archivo importado se carga completo junto con el `CLAUDE.md`.

```
@AGENTS.md

# Comandos
- Servidor de desarrollo: npm run dev
- Lint: npm run lint
- Tipos: npx tsc --noEmit
- Build de producción: npm run build, solo si te lo pido.

# Convenciones
- Los precios llegan en centavos; formatéalos con lib/precio.ts.
- components/legacy/ sigue en uso, pero no copies sus patrones.

# Cuidado
- app/checkout/ y app/api/pagos/: primero un plan, nunca un cambio directo.
- No leas ni imprimas .env.local.
```

> **CUIDADO**
> «No leas ni imprimas .env.local» es una instrucción, no un candado. Claude Code trata el `CLAUDE.md` como contexto, no como configuración obligatoria. Si necesitas que sea imposible, bloquéalo con un hook o con permisos.

PASO 03

Plantilla para un tema de Shopify

Guárdala como `CLAUDE.md` en la raíz del tema, junto a `layout/` y `sections/`. Cambia `tu-tienda.myshopify.com` por tu tienda.

```
# Comandos
- Vista previa: shopify theme dev --store tu-tienda.myshopify.com
- Revisar el tema: shopify theme check

# Convenciones
- Los textos visibles van en locales/ y se leen con el filtro t, nunca
  escritos directo en el Liquid.
- No renombres ni borres el id de un setting que ya existe: la tienda
  tiene valores guardados con ese id.

# Cuidado
- Nunca corras shopify theme push ni shopify theme publish: suben o
  publican en la tienda. Si hace falta, pídemelo.
- layout/theme.liquid carga la analítica: avísame antes de tocarlo.
```

- `shopify theme check` revisa el tema en tu computadora. Lo probamos en un tema mínimo: sin errores termina con código 0; con una clave de traducción que no existe en `locales/` reporta `TranslationKeyExists` y termina con código 1. Es justo el error que evita la primera convención.
- `shopify theme dev` sube el tema como tema de desarrollo a la tienda conectada y te da una vista previa local en `http://127.0.0.1:9292`. No publica nada.
- **Los ids de settings** son la llave con la que la tienda guarda lo que ya configuró en el editor de temas; por eso no se renombran.

> **CUIDADO**
>
> `shopify theme push` sube tus archivos al tema de la tienda y `shopify theme publish` cambia el tema publicado. Por eso la plantilla los prohíbe: escribe en tu `CLAUDE.md` los comandos que sí puede correr y deja estos para ti.

PASO 04

Mueve lo específico a una regla

Si el `CLAUDE.md` crece, saca lo que solo aplica a una parte del código a `.claude/rules/`. Una regla con `paths` se carga cuando Claude lee o edita un archivo que coincide. Esta, en `.claude/rules/liquid.md`, aplica solo a secciones y snippets:

```
---
paths:
  - "sections/**/*.liquid"
  - "snippets/**/*.liquid"
---
- Los textos visibles van en locales/ y se leen con el filtro t.
- Después de editar un archivo Liquid, corre shopify theme check.
```

Comprobamos que los dos patrones coinciden con `sections/hero.liquid` y `snippets/envio.liquid` y no con `layout/theme.liquid`. Una advertencia de la documentación: estas reglas se resumen junto con la conversación al compactar y vuelven a cargarse cuando Claude toca otra vez un archivo que coincide. Si una regla debe estar siempre, quítale `paths`.

PASO 05

Comprueba que cargó

1. Abre Claude Code en la raíz del proyecto.
2. Corre `/context`. Tu `CLAUDE.md` debe aparecer en la lista **Memory files**.
3. Si no aparece, corre `/memory`: lista los archivos de memoria de usuario y de proyecto y te deja abrirlos.
4. Un `CLAUDE.md` de subcarpeta no aparece en esa lista al iniciar, porque se carga cuando Claude trabaja en esa carpeta.

PASO 06

Durante la sesión

Nombra los archivos de la tarea con `@` y la ruta, di qué no tocar y cómo se ve «terminado». Cuando la sesión se alarga, decide tú qué se queda:

SITUACIÓN	COMANDO	QUÉ PASA
Misma tarea, sesión larga	`/compact` con foco	Resume la conversación; el `CLAUDE.md` de la raíz se vuelve a leer del disco
Tarea nueva, sin relación	`/clear`	Conversación nueva con el contexto vacío; la anterior se recupera con `/resume`
Corregiste lo mismo dos veces	`/clear`	Empiezas sin los intentos fallidos, con un prompt que incluya lo aprendido
Solo una parte es ruido	`/rewind`	Resume desde o hasta el mensaje que elijas

Un `/compact` con foco se escribe así:

```
/compact conserva los cambios en components/carrito/ y los comandos de prueba; descarta la exploración de app/checkout/
```

Una instrucción que diste solo en el chat puede perderse al compactar. Si importa en cada sesión, muévela al `CLAUDE.md`.

FINAL

Checklist

- [ ] Mi `CLAUDE.md` tiene menos de 200 líneas.
- [ ] Cada línea es algo que Claude no podría deducir leyendo el código.
- [ ] Los comandos existen en mi `package.json` o en mi CLI, con esos nombres.
- [ ] Si el proyecto tiene `AGENTS.md`, mi `CLAUDE.md` lo importa con `@AGENTS.md`.
- [ ] Lo que solo aplica a una carpeta está en una regla con `paths`.
- [ ] Lo que no puede fallar está en un hook o en permisos, no en una línea.
- [ ] `/context` muestra mi `CLAUDE.md` en **Memory files**.
- [ ] En cada prompt nombro los archivos con `@` y digo qué no tocar.
- [ ] Uso `/clear` entre tareas sin relación y `/compact` con foco en la misma tarea.