

GUÍA DE CONSTRUCCIÓN · CLAUDE CODE

De link a campaña

Construí, sin escribir una línea de código, el sistema que lleva una propiedad desde el link del portal hasta la campaña publicada. Quince prompts, en orden.

Pegás un link → copies → post · story · carrusel → ficha PDF
→ creativos de anuncio → reel → campaña en Meta

QUÉ ES

MANUAL DE CONSTRUCCIÓN
ASISTIDA POR IA

PARA QUIÉN

INMOBILIARIAS Y EQUIPOS
DE MARKETING

STACK

NEXT.JS 16 · REACT 19
OPENROUTER · META API

Cómo se usa esta guía

Esta guía no te explica cómo programar. Te da las instrucciones ya escritas para que se las dictes a Claude Code, que es quien programa.

Cada paso tiene un **bloque negro**. Ese bloque es texto que vos no tenés que entender: lo copiás entero y lo pegás en Claude Code. Está escrito en segunda persona porque le habla a él, no a vos.

Lo poco que está dirigido a vos es el texto blanco que rodea a cada bloque: qué vas a conseguir con ese paso y cómo comprobar que salió bien.

LA REGLA MÁS IMPORTANTE

Pegá **un solo bloque por vez** y esperá a que termine antes de pasar al siguiente. Los prompts están ordenados: cada uno da por hecho lo que construyó el anterior. Si los pegás todos juntos, cuando algo falle no vas a saber dónde, y arreglarlo cuesta más que haberlo hecho por partes.

Los pasos **00 a 07** son el núcleo: con esos ya tenés un sistema que sirve para trabajar. Los pasos **08 a 14** son módulos independientes — pedí solo los que uses.

Índice

Parte 0	Preparar la máquina	09	Creativos de anuncio
00	Fundaciones del proyecto	10	Publicar en Instagram
01	Identidad de marca	11	Campañas en Meta Ads
02	Propiedades y galería	12	Asesor de tráfico
03	Clasificación de fotos con IA	13	Editor de reels
04	Importar pegando el link	14	Exportar en ZIP
05	Copies de venta	Ref.	Modelos, límites y costos
06	Post y story	Ref.	Las nueve trampas
07	Ficha técnica en PDF	Ref.	Checklist final
08	Carrusel generado con IA		

Preparar la máquina

Tres instalaciones y una llave. Quince minutos, una sola vez.

1 • Node 22

Descargalo de nodejs.org y elegí la versión **LTS**. Es el motor sobre el que corre todo. Instalador siguiente-siguiente-terminar.

NO INSTALES LA VERSIÓN "CURRENT"

La guía fija a propósito algunas librerías en versiones que no funcionan con Node muy nuevo. Con la LTS vas seguro.

2 • Claude Code

Es quien va a escribir la aplicación. Se instala desde la terminal con un comando que encontrás en claude.com/claude-code. Funciona en Windows, Mac y Linux.

3 • La carpeta del proyecto

Creá una carpeta vacía donde quieras — por ejemplo `Documentos\propads` — y abrí Claude Code adentro. Todo lo que sigue pasa ahí.

EVITÁ ONEDRIVE, DROPBOX O GOOGLE DRIVE

Si la carpeta está sincronizada, el sincronizador toma bloqueos sobre los archivos justo cuando la app los está escribiendo, y vas a tener fallos intermitentes imposibles de rastrear. El paso 00 incluye una defensa contra esto, pero es mucho mejor no pelear la batalla.

4 • La llave de OpenRouter

Es la única llave obligatoria. Una sola cuenta te da acceso a todos los modelos de IA que usa el sistema: los textos, las imágenes y el análisis de fotos.

Entrá a openrouter.ai, creá la cuenta, andá a **Keys** y generá una. Cargá **USD 5** de saldo: alcanza para meses de uso normal.

5 • Las llaves opcionales

Cada una activa un módulo. Sin ellas la aplicación funciona igual, con ese módulo apagado. Conseguilas solo cuando llegues al paso que las usa.

SERVICIO	PASO	PARA QUÉ	COSTO
Firecrawl	04	Importar propiedades pegando el link del portal	Plan gratis alcanza
ElevenLabs	13	La voz en off de los reels	Plan gratis limitado
Upload-Post	10	Publicar el post en Instagram	Desde USD 10/mes

DÓNDE SE GUARDAN LAS LLAVES

En un archivo llamado `.env.local` dentro de la carpeta del proyecto. El paso 00 lo crea. Ese archivo **nunca** se comparte ni se sube a ningún lado: es el equivalente a tu contraseña.

El núcleo

Ocho prompts. Al terminarlos tenés un sistema que ya sirve para trabajar: cargás una propiedad y salen los copies, el post, la story y la ficha en PDF, con tu marca aplicada.

PASO 00 · BASE DE TODO

Fundaciones del proyecto

Levanta el esqueleto y, sobre todo, deja resueltas de entrada las tres cosas que más adelante son carísimas de arreglar: cómo se guardan los archivos, cómo se dibuja el texto y de dónde salen los colores.

PEGÁ ESTO EN CLAUDE CODE

PASO 00

Vas a construir conmigo una aplicación web local de marketing inmobiliario. Es para un solo usuario, corre en mi máquina y NO usa base de datos remota, ni login, ni servicios en la nube: todo persiste en el filesystem.

En este primer paso quiero SOLO las fundaciones. No construyas ninguna pantalla de negocio todavía.

— STACK — respetalo exactamente, no propongas alternativas

Next.js 16 con App Router, React 19, TypeScript en modo strict, Tailwind CSS 3.4.

Zod 3 para validar. Zustand para estado de cliente. Un único alias de imports:

@/* apuntando a src/*.

Imágenes generadas en el servidor con @napi-rs/canvas.

PDFs con puppeteer.

IA a través de OpenRouter, usando el SDK de openai apuntado a

https://openrouter.ai/api/v1.

— ARQUITECTURA — esto define todo lo que viene después

Una PROPIEDAD es el expediente central del sistema. Todos los módulos futuros

leen y escriben la MISMA carpeta:

```
data/properties/<id>/
  property.json      datos, textos y el estado de cada módulo
  photos/gallery/    galería de fotos
  outputs/           las piezas generadas
```

Reglas que valen para siempre:

- Un módulo nuevo EXTIENDE property.json con sus propios campos. Nunca crea su propio almacenamiento paralelo ni duplica datos de la propiedad.
- El código de dominio va en src/lib/, la interfaz en src/features/<módulo>/, las rutas en src/app/.
- Ningún archivo pasa de 500 líneas, ninguna función de 50.

— QUÉ QUIERO EN ESTE PASO

1. El proyecto Next.js creado y arrancando con `npm run dev`, con Tailwind configurado y los scripts dev, build, lint y typecheck.
2. Una capa de almacenamiento en src/lib/storage.ts con funciones para resolver las rutas de una propiedad, leer y escribir property.json, y listar propiedades. Genera los IDs con nanoid.

La escritura de JSON tiene que ser ATÓMICA: escribir primero en un archivo temporal y después renombrarlo sobre el definitivo, con reintentos y espera creciente, y si el rename sigue fallando, caer a copiar y borrar. Motivo: si la carpeta llega a estar sincronizada con OneDrive o Dropbox, el sincronizador toma bloqueos cortos y los renames fallan de forma intermitente con EPERM o EBUSY. Sin esto se corrompen datos.

3. Registro de fuentes para el canvas en `src/lib/canvas/fonts.ts`.

Descargá la familia Lato (licencia SIL Open Font License, uso libre) en `src/shared/assets/fonts/` y registrá los `.ttf` con `GlobalFonts.registerFromPath()` en una función que se llame antes de dibujar cualquier texto. Exportá dos constantes con la familia registrada, una regular y una negrita, y usá SIEMPRE esas constantes.

Esto es crítico: `@napi-rs/canvas` no trae fuentes propias. Si el código pide "Segoe UI" o "Arial" y esa fuente no está instalada en la máquina, no falla ni avisa – dibuja el texto en blanco absoluto y las piezas salen vacías.

En `next.config.ts` agregá `@napi-rs/canvas`, `puppeteer` y `sharp` a `serverExternalPackages`, y un `outputFileTracingIncludes` que incluya la carpeta de fuentes para las rutas de API. Si no, el bundler no rastrea los `.ttf` y desaparecen.

4. Helpers de color en `src/lib/canvas/common.ts`:

- una función que dado un color de fondo en hexadecimal calcule su luminancia y devuelva si el texto encima debe ir negro o blanco;
- una función que devuelva el color de acento leyéndolo del perfil de marca, con un valor por defecto.

Prohibido escribir un color de acento fijo en cualquier otro lugar del código.

5. Un helper de formato de fecha hecho a mano en `src/shared/lib/format.ts`.

No uses `toLocaleString` ni `Intl` en nada que se muestre en pantalla: el servidor y el navegador formatean distinto – el servidor mete un espacio invisible `U+202F` – y React lo reporta como error de hidratación.

6. Un archivo `.env.local` con la variable `OPENROUTER_API_KEY` vacía, y un `.env.local.example` igual. Agregá `.env.local` y `data/` al `.gitignore`.

7. Configuraré ESLint con flat config usando `eslint-config-next`, ignorando `.next`, `node_modules`, `data` y `public`. Comprabá que ``npm run lint`` corre de verdad.

— CÓMO VERIFICAR ANTES DE DECIRME QUE TERMINASTE

Corré `npm run typecheck`, `npm run lint` y `npm run build`, los tres sin errores.

Levantá `npm run dev` y confirmá que la página abre.

Escribí un `property.json` de prueba con la capa de almacenamiento, volvé a leerlo y mostrame el contenido.

Cuando termines, mostrame el árbol de carpetas creado y esperá mi visto bueno.

QUÉ TENÉS QUE VER AL FINAL DEL PASO 00

Una página en blanco abriendo en el navegador y, en la carpeta del proyecto, una subcarpeta `src` con `lib`, `features` y `app` adentro. Nada más. Si Claude Code se entusiasmó y te construyó pantallas, pedile que las saque: van a chocar con los pasos que siguen.

SI TE PREGUNTA POR PERMISOS

Claude Code va a pedirte autorización para crear archivos y para instalar dependencias. Aceptá. Es lo esperable en este paso.

Identidad de marca

Va antes que las propiedades a propósito. La marca no es decorativa: define el color con el que se pintan todas las piezas y el tono con el que la IA escribe cada texto. Si la dejás para el final, hay que rehacer todo lo anterior.

PEGÁ ESTO EN CLAUDE CODE

PASO 01

Agregá el módulo de MARCA. Es la identidad que se inyecta en todas las piezas y en todos los prompts del sistema.

— DATOS

Guardá el perfil en `data/brand/profile.json` con estos campos:

- nombre de la agencia
- color primario en hexadecimal
- color secundario en hexadecimal, opcional
- logo (archivo, guardado en `data/brand/logos/`)
- teléfono de WhatsApp y país, para armar el enlace de contacto
- sitio web e Instagram, opcionales
- tono de voz: cercano, profesional, lujoso o directo
- público objetivo, en una frase
- palabras que SIEMPRE debe usar y palabras que NUNCA debe usar

— PANTALLA

Una ruta `/marca` con un formulario para editar todo esto, la vista previa del logo y un selector de color que muestre el color aplicado.

Cuando suba el logo, extraé sus colores dominantes con IA de visión y ofrecelos como sugerencia para el color primario – pero que yo pueda ignorarlos y elegir el mío.

— CÓMO SE USA EN EL RESTO DEL SISTEMA

Creá en `src/lib/brand/` dos piezas que van a usar todos los módulos siguientes:

1. Una función que lea el perfil de marca, con valores por defecto sensatos si todavía no se configuró. Nada del sistema debe romperse por no tener marca.
2. Un constructor de encabezado de marca para prompts: recibe el perfil y devuelve un bloque de texto con el nombre, el tono, el público y las palabras permitidas y prohibidas, listo para anteponer a cualquier prompt de IA que genere texto.

Y el color primario tiene que quedar disponible para las piezas gráficas a través del helper de acento que creaste en el paso 00.

— VERIFICACIÓN

Configurá una marca de prueba con color verde, guardá, recargá la página y confirmá que persistió. Mostrame el contenido de `data/brand/profile.json`.

Propiedades y galería

El corazón del sistema. Todo lo que viene después lee de acá.

PEGÁ ESTO EN CLAUDE CODE

PASO 02

Agregá el alta y el listado de PROPIEDADES.

— DATOS DE UNA PROPIEDAD

Definí el esquema con Zod y derivá el tipo de TypeScript de ahí – un solo lugar donde se declara la forma del dato.

- tipo: casa, departamento, terreno u oficina
- operación: venta o alquiler
- título
- precio y moneda (ARS, USD, MXN, CLP, COP)
- expensas y su moneda, opcional
- dirección, barrio, ciudad, país
- metros cuadrados: cubiertos, semicubiertos, descubiertos y de terreno
- ambientes, dormitorios, baños, toilettes, cocheras, niveles
- antigüedad
- amenities: lista de textos libres
- descripción
- destacados: lista corta de puntos fuertes

Todos los campos numéricos son opcionales salvo el precio: los avisos reales vienen incompletos y el formulario tiene que tolerarlo.

— GALERÍA

Hasta 10 fotos por propiedad, guardadas en photos/gallery/ dentro de la carpeta de la propiedad. Cada foto se registra en property.json con su nombre de archivo y una marca opcional de "foto principal".

En la interfaz: subida por arrastrar y soltar, miniaturas, reordenar, borrar, y marcar una como principal. La foto principal es la que van a usar la portada del carrusel, el post y la ficha técnica.

Definí el tope de 10 como una constante exportada, no como un número suelto.

— PANTALLAS

- La ruta / muestra el listado de propiedades, cada una con su foto principal, título, precio y un indicador de qué piezas ya tiene generadas.
- Una ruta para crear y editar una propiedad con el formulario completo.
- Las fotos se sirven por una ruta de API que lea del filesystem. No las copies a public/: son archivos del cliente y no deben quedar expuestos.

— SEGURIDAD DE LAS RUTAS DE ARCHIVO

La ruta que sirve archivos recibe un id de propiedad y una ruta relativa. Resolvé la ruta absoluta y verificá que quede DENTRO de la carpeta de esa propiedad antes de leer nada. Si queda fuera, devolvé 404. Sin esto, una ruta con ".." deja leer cualquier archivo de mi disco.

— VERIFICACIÓN

Creá una propiedad con datos reales y seis fotos. Confirmá que aparece en el listado con su portada, que las fotos se ven, que puedo reordenarlas y que al recargar sigue todo igual. Mostrame el property.json resultante.

Clasificación de fotos con IA

Un paso corto que parece menor y no lo es: sin esto, el carrusel del paso 08 pone la foto del baño en el slide de la cocina.

PEGÁ ESTO EN CLAUDE CODE

PASO 03

Agregá clasificación automática de las fotos de la galería con IA de visión.

Cuando se suban fotos, mandalas al modelo google/gemini-2.5-flash a través de OpenRouter y pedile que etiquete cada una con UNA de estas categorías:

fachada · sala · cocina · dormitorio · bano · exterior · vista · piscina · general

Definí esas categorías como una constante y derivá el tipo de ahí.

Detalles de la llamada:

- temperatura 0.2 – esto es clasificación, no creatividad
- límite de 800 tokens de salida
- clasificá todas las fotos en UNA sola llamada, mandando las imágenes juntas y pidiendo un arreglo de resultados. Una llamada por foto multiplica el costo y el tiempo sin ninguna ventaja.
- validá la respuesta con Zod antes de guardarla, y si una categoría no está en la lista, guardá "general" en vez de romper

Guardá la categoría y la fecha de clasificación en el registro de cada foto dentro de property.json.

En la interfaz, mostrá la categoría sobre cada miniatura y dejame corregirla a mano: la IA se equivoca y yo conozco la propiedad.

Si la clasificación falla, la propiedad tiene que seguir funcionando igual, solo que sin categorías.

— VERIFICACIÓN

Subí seis fotos de ambientes distintos y mostrame qué categoría le asignó a cada una. Corregí una a mano y confirmá que la corrección persiste.

Importar pegando el link

Convierte quince minutos de tipeo por propiedad en veinte segundos. Requiere la llave de Firecrawl.

PEGÁ ESTO EN CLAUDE CODE

PASO 04

Agregá importar una propiedad pegando la URL de un portal inmobiliario (ZonaProp, Argenprop, Inmuebles24, Portal Inmobiliario, Fincaraíz).

— CÓMO

1. Bajá el contenido de la página con Firecrawl, endpoint `https://api.firecrawl.dev/v2/scrape`, con la clave en `FIRECRAWL_API_KEY`.
2. Pasale el texto obtenido a `google/gemini-2.5-flash` pidiendo extracción estructurada de TODOS los campos de la propiedad que definimos en el paso 02. Usá salida estructurada con esquema JSON, no texto libre.
3. Descargá las imágenes del aviso, quedate con las 10 primeras de mayor resolución y guardalas en la galería.

— SI FIRECRAWL FALLA

Implementá un camino alternativo que baje el HTML directo con un user-agent de navegador y se lo pase igual al modelo. Muchos portales responden bien y así el sistema no depende de un solo proveedor.

— SEGURIDAD – no te lo saltees

La URL la escribo yo, pero el sistema la va a seguir sin preguntar. Antes de hacer cualquier petición a una URL que venga de afuera:

- aceptá solo http y https
 - resolvé el nombre de dominio a su IP y rechazá si cae en rango privado, loopback, link-local o metadata de nube
 - seguí los redirects a mano, revalidando la IP en CADA salto
- Poné esto en un helper reutilizable y usalo en todas las descargas externas.

— INTERFAZ

Después de extraer, mostrame los datos en un FORMULARIO EDITABLE y esperá que yo confirme antes de crear la propiedad. El scraping falla seguido y necesito poder corregir antes de guardar.

Si no hay clave de Firecrawl configurada, ocultá el botón de importar en lugar de mostrar un error.

— VERIFICACIÓN

Importá un aviso real de un portal y mostrame qué campos extrajo, cuáles quedaron vacíos y cuántas fotos bajó.

Copies de venta

El primer módulo que usa el tono de marca del paso 01.

PEGÁ ESTO EN CLAUDE CODE

PASO 05

Agregá la generación de COPIES con IA para una propiedad.

— QUÉ GENERA, en una sola llamada

- description descripción larga de venta, para el portal
- instagram_caption pie de foto con emojis y hashtags
- email_subject asunto de correo
- email_body_html cuerpo del correo en HTML simple
- headline_post titular corto para el post cuadrado
- headline_story titular corto para la story vertical

— CÓMO

Modelo google/gemini-2.5-flash a través de OpenRouter, temperatura 0.85. Usá salida estructurada con esquema JSON estricto y los seis campos como requeridos. Validá con Zod igual, aunque venga estructurada.

Antepone al prompt el encabezado de marca del paso 01: tono, público, palabras permitidas y prohibidas.

Usá la jerga del país de la propiedad. En Argentina y Chile son "dormitorios"; en México "recámaras"; en Colombia "habitaciones". No mezcles nunca dentro de un mismo texto.

Escribí los precios con separador de miles y símbolo de moneda correctos para el país.

— MANEJO DE ERRORES

Traducí los errores de OpenRouter a mensajes que yo entienda y pueda accionar:

- 401 → la clave es inválida, revisá .env.local
- 402 → te quedaste sin saldo en OpenRouter
- 429 → demasiadas peticiones, esperá un momento

No me muestres nunca "algo salió mal" a secas.

Guardá los copies y la fecha en property.json, y si falla, guardá también el error para poder mostrarlo.

— VERIFICACIÓN

Generá copies para la propiedad de prueba y mostrámelos completos. Cambiá el tono de la marca a "lujoso", regenerá, y mostrame los dos textos uno al lado del otro para confirmar que el tono efectivamente cambió.

Post y story

Acá es donde aparece la trampa de las fuentes. El prompt la neutraliza de entrada.

PEGÁ ESTO EN CLAUDE CODE

PASO 06

Agregá la generación del POST cuadrado y la STORY vertical con @napi-rs/canvas.

— MEDIDAS

- post 1080 x 1080
- story 1080 x 1920

— QUÉ LLEVA CADA PIEZA

- la foto principal de la galería, recortada para llenar el lienzo sin deformarse
- un degradado oscuro en la parte inferior para que el texto se lea sobre cualquier foto
- el titular corto que generó el paso 05
- el precio, destacado
- los datos clave en una línea: ambientes, baños, metros – con la jerga del país
- el logo de la marca
- un código QR que abre WhatsApp con un mensaje ya escrito sobre esa propiedad

— REGLAS INNEGOCIABLES

1. Llamá a la función de registro de fuentes del paso 00 ANTES de dibujar, y usá solo las constantes de fuente registradas. Nunca nombres una fuente del sistema.
2. Todos los acentos salen del color de marca. Para cada texto sobre un fondo de color, calculá la luminancia del fondo y elegí texto negro o blanco. Ningún color escrito fijo.
3. El texto que no entra se corta con puntos suspensivos, no se desborda ni se encima. Medí el ancho real del texto con el canvas antes de dibujarlo.

— EL QR DE WHATSAPP

Normalizá el teléfono antes de armar el enlace:

- si el número empieza con "+", ya está completo, dejalo
- si empieza con "00", sacá esos dos dígitos
- si no tiene código de país, agregá el que corresponde al país de la marca, y sacá el cero inicial de larga distancia nacional
- si después de todo eso no quedan dígitos, NO dibujes el QR

Sin esta normalización, un número argentino cargado como "+54 11..." termina convertido en "+5454 11..." y el QR no lleva a ninguna parte.

— SALIDA

Guardá los PNG en outputs/ dentro de la carpeta de la propiedad, y registrá la ruta y la fecha en property.json.

— VERIFICACIÓN – hacé esto de verdad, no me digas que compila

Generá las dos piezas. ABRÍ los archivos PNG generados y confirmá que:

- el texto se ve (si sale vacío, las fuentes no quedaron registradas)
- el precio usa el color de mi marca, no un dorado ni un azul por defecto
- el QR aparece y apunta al número correcto

Mostrame las dos imágenes.

Ficha técnica en PDF

La pieza que se manda por correo o por WhatsApp a un interesado.

PEGÁ ESTO EN CLAUDE CODE

PASO 07

Agregá la generación de la FICHA TÉCNICA en PDF de una propiedad.

— CÓMO

Armá una plantilla HTML con CSS y convertila a PDF con puppeteer. No uses librerías de dibujo de PDF: con HTML el diseño es mucho más fácil de ajustar.

— CONTENIDO, en una sola página A4

- encabezado con el logo y el nombre de la agencia
- la foto principal, grande
- título, dirección y precio destacado
- tabla de datos: ambientes, dormitorios, baños, cocheras, metros cubiertos, metros de terreno, antigüedad, expensas
- la descripción generada en el paso 05
- una grilla de hasta ocho fotos más de la galería
- pie con teléfono, web e Instagram de la marca

— QUE SEA UNA SOLA PÁGINA — esto es lo que más cuesta

- la página tiene alto fijo con el contenido en columna flexible
- el bloque central absorbe el espacio sobrante y recorta lo que sobre
- la descripción se limita a ocho líneas con recorte por CSS
- al exportar, forzá explícitamente el rango de páginas a la primera

Sin lo último, una descripción larga genera una segunda página casi vacía.

— MARCA

El color de acento del encabezado, la línea del precio y el pie salen del color primario de la marca. Nada fijo.

— SALIDA

Guardá el PDF en outputs/ y registrá su ruta y fecha en property.json.

— VERIFICACIÓN

Generá la ficha y ABRÍ el PDF. Confirmá contando las páginas que tiene exactamente una. Confirmá que el color del encabezado es el de mi marca. Probá después con una descripción muy larga y confirmá que sigue siendo una sola página.

HASTA ACÁ LLEGA EL NÚCLEO

Con los pasos 00 a 07 ya tenés un sistema utilizable: cargás una propiedad, salen los textos, el post, la story y la ficha. Antes de seguir, usalo con una propiedad real de tu cartera. Los módulos que siguen son independientes entre sí — pedí solo los que vayás a usar.

Los módulos

Independientes entre sí. Cada uno agrega una capacidad completa sobre el mismo expediente de propiedad. Pedí solo los que uses, en el orden que quieras.

PASO 08 · ORGÁNICO

Carrusel generado con IA

Toma tus fotos reales y les sobreimprime texto. No inventa la propiedad: la respeta.

PEGÁ ESTO EN CLAUDE CODE

PASO 08

Agregá un CARRUSEL de Instagram de 5 a 7 slides.

— EL PLAN DEL CARRUSEL

Antes de generar nada, armá el plan eligiendo qué foto y qué texto lleva cada slide, usando las categorías del paso 03:

- portada: la foto que yo marqué como principal; si no marqué ninguna, la mejor de categoría fachada
- slides intermedios: una foto de cada ambiente distinto, sin repetir
- cierre: llamado a la acción con los datos de contacto de la marca

Los titulares de cada tipo de ambiente salen de una LISTA rotativa de varias opciones, eligiendo por posición del slide. Si es una frase fija, dos carruseles seguidos salen idénticos.

Usá la jerga del país: dormitorios en Argentina y Chile, recámaras en México, habitaciones en Colombia. Y su abreviatura corta en la línea de datos.

— LA GENERACIÓN

Modelo google/gemini-3.1-flash-image-preview vía OpenRouter, en modo imagen a imagen. El orden del contenido importa:

1. la foto real de la propiedad como primera imagen — es la base
2. el logo de la marca como segunda imagen, si hay
3. el texto del prompt al final

Tiempo máximo de espera por imagen: 90 segundos.

La foto NO se reemplaza por una generada. El modelo solo sobreimprime.

— CÓMO ESCRIBIR EL PROMPT DE CADA SLIDE — el detalle que rompe todo
Separá SIEMPRE el texto a dibujar de las especificaciones de estilo:

TEXT (render verbatim, nothing else): "Cocina integrada"

Style: bottom-left, bold sans, ~22px, 78% opacity backdrop

Y agregá una regla global que prohíba dibujar medidas, porcentajes o nombres de color como texto visible.

Si mezclás el texto y las specs en un mismo párrafo, el modelo dibuja "22px" y "78% opacity" adentro de la imagen. Es el error más caro de detectar porque la imagen se ve bien hasta que la mirás de cerca.

— EDICIÓN DE UN SLIDE SUELTO

Agregá poder regenerar UN slide sin rehacer el carrusel entero: le paso una instrucción en texto, se regenera esa imagen sola y se sobrescribe el mismo archivo. Como el nombre no cambia, agregá un parámetro con la marca de tiempo al mostrarla, para que el navegador no muestre la versión vieja de su caché.

— SALIDA

Los slides en outputs/carrusel/ dentro de la propiedad, numerados en orden, con la lista y la fecha registradas en property.json.

— VERIFICACIÓN

Generá un carrusel completo y ABRÍ los slides. Confirmá que ningún slide tiene medidas escritas encima, que la jerga corresponde al país y que dos carruseles seguidos no salen con los mismos titulares.

Creativos de anuncio

El módulo que más cambia el resultado comercial. En publicidad una pieza no alcanza: Meta necesita varias para encontrar cuál rinde.

PEGÁ ESTO EN CLAUDE CODE

PASO 09

Agregá el motor de CREATIVOS DE ANUNCIO por combinatoria.

— LOS DOS CATÁLOGOS

Definilos en UN SOLO lugar de src/lib/, porque el paso 11 los va a reutilizar.
Nunca los dupliques.

ÁNGULOS de venta (once):

inversion · urgencia · lifestyle · ubicacion · primer_hogar · seguridad_familia
amenities · social_proof · educativo · tecnico · oportunidad

AUDIENCIAS (seis):

inversionista · familia_creciendo · joven_profesional · pareja_joven
empresario · jubilado

Cada entrada lleva un identificador, una etiqueta legible y una descripción de en qué se enfoca. La descripción es la que va al prompt.

— FORMATO

Uno solo: cuadrado 1080 x 1080. Meta lo acepta en Feed, Stories, Reels, Explore y Marketplace, así que un formato cubre todas las ubicaciones sin multiplicar el costo de generación.

— EL FLUJO

Yo elijo qué ángulos y qué audiencias quiero. El sistema genera una pieza por cada cruce. Para cada creativo:

- un copy propio, escrito desde ESE ángulo para ESA audiencia
- una imagen generada a partir de la foto principal, con el color y el logo de mi marca

Los copies se generan TODOS en una sola llamada, pidiendo un arreglo – no una llamada por creativo. Usá un modelo de escritura fuerte, temperatura 0.9, con límite de salida alto (8000 tokens). Tolerá que la respuesta venga envuelta en un bloque de código al parsearla.

— CONCURRENCIA Y TOPEs – importante para el bolsillo

- generá como máximo 3 imágenes en paralelo; más que eso y OpenRouter devuelve 429
- tope duro de 60 creativos por lote, como constante exportada, para que un clic distraído no dispare el costo
- mostrame el costo estimado ANTES de empezar

— PROGRESO REAL

Guardá el lote en data/properties/<id>/creatives/<loteId>/ con un batch.json que registre, por cada creativo, su ángulo, su audiencia, su copy y su estado. Actualizá ese archivo a medida que cada pieza termina, y que la pantalla muestre los que ya están mientras el resto sigue generándose. Nada de una barra falsa.

Si el proceso falla a mitad, el lote queda marcado como fallido pero los creativos ya generados se conservan.

— EXPORTACIÓN

Un botón que descargue el lote como ZIP con:

- las imágenes nombradas por ángulo y audiencia
- un `copies.csv` con el texto de cada anuncio

El CSV tiene que abrir bien en Excel con acentos: escríbelo en UTF-8 con marca de orden de bytes al principio.

— VERIFICACIÓN

Generé un lote chico: 2 ángulos por 2 audiencias, cuatro piezas. Confirmé que los cuatro `copies` son distintos entre sí y coherentes con su ángulo. Descargué el ZIP, abrílo, y abrí el CSV en Excel para confirmar que los acentos se ven.

Publicar en Instagram

PEGÁ ESTO EN CLAUDE CODE

PASO 10

Agregá publicar en Instagram las piezas ya generadas: el post simple y el carrusel.

— CÓMO

Usá Upload-Post. Endpoint POST https://api.upload-post.com/api/upload_photos, autenticación por cabecera Authorization con el valor "Apikey" seguido de la clave. El cuerpo va como multipart con el usuario, la plataforma, las fotos en orden, y el pie de foto.

Una imagen es un post de feed; entre dos y diez es un carrusel. Diez es el tope de Instagram: definilo como constante y validalo antes de enviar.

Como la aplicación es local, los archivos se leen del disco y se adjuntan directo. No hay que subirlos a ningún lado primero.

Limitá lo publicable a post y carrusel. Stories y reels usan otros endpoints y no están cubiertos acá.

— REGLAS

1. Antes de publicar, mostrame una vista previa con la imagen exacta y el copy exacto que se van a subir, y un botón de confirmar. Nunca publiques de un solo clic: es irreversible y es mi cuenta.
2. Registrá SIEMPRE el intento en property.json – fecha, qué pieza, y si salió bien o mal, con el mensaje de error si falló. Necesito el historial completo, no solo los éxitos.
3. Si faltan las credenciales, mostrá el botón deshabilitado explicando qué falta. No lo escondas.
4. Si el servicio devuelve error, mostrame el mensaje real que devolvió.

— VERIFICACIÓN

Probá primero la validación con credenciales vacías y confirmá que avisa bien. NO publiques a mi cuenta real hasta que yo te lo pida expresamente.

Campañas en Meta Ads

El módulo que toca dinero real. El prompt lo deja todo en pausa a propósito.

PEGÁ ESTO EN CLAUDE CODE

PASO 11

Agregá la creación de CAMPAÑAS en Meta con la Graph API, sobre una propiedad existente.

— CREDENCIALES

META_ACCESS_TOKEN (token de usuario de sistema del business, con permiso ads_management), META_AD_ACCOUNT_ID, META_PAGE_ID, META_BUSINESS_ID y META_GRAPH_API_VERSION con valor por defecto v23.0.

— CLIENTE CON CONTROL DE CUOTA — construí esto ANTES que nada

Meta cobra la cuota por puntos en una ventana deslizante de 5 minutos:

- nivel desarrollo: 60 puntos, y si te pasás te bloquea 300 segundos
- nivel estándar: 9000 puntos, bloqueo de 60 segundos
- una lectura cuesta 1 punto, una escritura cuesta 3

Implementá en el cliente:

- un registro de los puntos gastados en los últimos 5 minutos
- antes de cada llamada, esperar si el gasto acumulado más el costo de esta llamada superan el cupo. Usá 50 y 8000 como topes, para dejar colchón.
- detección del nivel leyendo las cabeceras de uso que devuelve Meta
- una pausa mínima de 200 ms entre llamadas y las llamadas encoladas en serie
- reintento con espera creciente, hasta 4 veces, solo ante estos códigos de error de saturación: 4, 17, 32, 613, 80000, 80003, 80004, 80014
- los códigos 190 y 102 son de token vencido: no reintentes, avisame que hay que renovarlo

Sin esto, en desarrollo agotás la cuota en menos de un minuto y quedás bloqueado cinco.

— EL FLUJO

Yo elijo la propiedad, el objetivo (mensajes o leads), el presupuesto diario, el radio geográfico alrededor de la dirección y el rango de edad.

El sistema crea la campaña, el conjunto de anuncios y los anuncios, reutilizando los creativos del paso 09 y los MISMOS catálogos de ángulos y audiencias.

Para la segmentación, geocodificá la dirección a coordenadas y armá un radio.

Traducí las audiencias del catálogo a intereses reales buscándolos en la API de segmentación de Meta.

Si el objetivo es leads, creá también el formulario con las preguntas típicas del rubro: presupuesto, plazo de compra y si necesita crédito.

— TRES REGLAS INNEGOCIABLES

1. Creá TODO en estado pausado. Nunca actives una campaña por tu cuenta: esto gasta dinero real de mi cuenta.
2. Antes de enviar nada, mostrame el resumen completo de lo que se va a crear y pedime confirmación explícita.
3. Guardá lo creado en meta-campaigns.json dentro de la carpeta de la propiedad, con los identificadores que devolvió Meta, y un enlace directo al administrador de anuncios.

— ERRORES

Los mensajes de la Graph API son crípticos y el detalle está en subcampos. Cuando falle, mostrame el código, el subcódigo, el mensaje completo y el identificador de traza, sin resumir.

— VERIFICACIÓN

Probá primero con credenciales inválidas y confirmá que el error se entiende. Después armá una campaña de prueba y mostrame el resumen – pero NO la envíes a Meta hasta que yo te lo confirme.

Asesor de tráfico

PEGÁ ESTO EN CLAUDE CODE

PASO 12

Agregá un ASESOR conversacional que lea mi cuenta publicitaria de Meta y responda preguntas sobre rendimiento en castellano.

— CÓMO

Un chat que usa un modelo con herramientas. Las herramientas leen de la Graph API a través del cliente con control de cuota del paso 11 – nunca por fuera.

Herramientas mínimas:

- listar campañas activas con su gasto y resultados
- traer métricas de un rango de fechas: gasto, alcance, clics, costo por resultado, frecuencia
- desglosar un conjunto de anuncios por edad, género y ubicación
- comparar el rendimiento de los anuncios de una campaña entre sí

— ELECCIÓN DE MODELO SEGÚN LA PREGUNTA

No uses el modelo más caro para todo. Clasificá la pregunta entrante y elegí:

- pregunta simple, de un dato → modelo económico
- pregunta de análisis con varias métricas → modelo intermedio
- pregunta de estrategia o diagnóstico → modelo fuerte

Limitá el bucle de herramientas a 6 iteraciones para que una pregunta ambigua no se vaya de gasto.

— CÓMO TIENE QUE RESPONDER

Con números concretos y una recomendación accionable, no con generalidades.

Si un dato no lo tiene, que lo diga en vez de estimarlo.

Que cite siempre el período de los datos que está usando.

— VERIFICACIÓN

Preguntale cuánto gastaste en los últimos 7 días y qué anuncio tuvo el costo por resultado más bajo. Confirmá que los números coinciden con el administrador de anuncios de Meta.

Editor de reels

El módulo más pesado del sistema. Dejalo para el final: renderiza en tu máquina y tarda minutos.

PEGÁ ESTO EN CLAUDE CODE

PASO 13

Agregá un EDITOR que convierta el video de una propiedad en un reel vertical de 1080 x 1920. Todo el render corre local con Remotion.

— LA TUBERÍA

1. Subo un video de la propiedad. Los archivos son grandes: subí el límite de tamaño del cuerpo de las peticiones a 1 gb en la configuración de Next, o cualquier video revienta con "Body exceeded 1 MB limit".
2. Se detectan las escenas del video con IA de visión y se etiqueta cada una (exterior, social, cocina, privados, amenities, detalle, cierre), con su segundo de inicio y de fin.
3. Se corta en clips que yo puedo reordenar y recortar.
4. La IA escribe el guion de venta a partir de los datos de la propiedad y el tono de mi marca, repartido en una frase por escena, de 15 a 25 palabras cada una.
5. El guion se narra con ElevenLabs. Los números hay que deletrearlos en el texto que se manda a la voz – "tres ambientes", no "3 ambientes" – o la voz los lee mal.
6. El audio se transcribe con whisper.cpp para obtener los tiempos palabra por palabra, y de ahí salen los subtítulos sincronizados.
7. Se renderiza el reel final.

— WHISPER

Usá el instalador de whisper.cpp de Remotion, fijando la versión 1.5.5 y el modelo "medium" – el chico se equivoca demasiado en castellano. Instalalo y descargá el modelo la primera vez que se use, no al arrancar la aplicación: son cientos de megas.

El audio hay que convertirlo a WAV de 16 kHz mono antes de transcribir.

Agregá la carpeta de whisper al .gitignore.

— SUBTÍTULOS

Ofrecé tres presentaciones para elegir: blanco en negrita, karaoke amarillo, y cinematográfico abajo.

— RENDER

Códec h264, imágenes intermedias en jpeg con calidad 80, preajuste de compresión "veryfast" y concurrencia automática. Con esos valores el render tarda una fracción de lo que tarda con los de fábrica, sin diferencia visible en Instagram.

Mostrá el progreso real por porcentaje.

— ESTADO

Cada etapa guarda su resultado en property.json y se puede rehacer sola, sin volver a empezar de cero. Si cambio el guion, se rehacen la voz y los subtítulos, no el análisis de escenas.

No bloquee la interfaz mientras renderiza.

— VERIFICACIÓN

Procesá un video corto de prueba de punta a punta y ABRÍ el mp4 final. Confirmá que los subtítulos están sincronizados con la voz y que el video es vertical.

Exportar en ZIP

PEGÁ ESTO EN CLAUDE CODE

PASO 14

Agregá un botón que descargue TODAS las piezas de una propiedad en un solo ZIP.

Contenido:

- post.png y story.png
- la carpeta del carrusel con sus slides numerados en orden
- la ficha técnica en PDF
- el reel en mp4, si existe
- los lotes de creativos, cada uno en su carpeta
- un copies.txt con todos los textos generados, listos para copiar y pegar

Nombres de archivo legibles, con el título de la propiedad al frente y sin acentos ni caracteres raros que rompan en Windows o Mac.

Armá el ZIP en memoria y devuelvelo como descarga. No dejes archivos temporales tirados en el disco.

Si una pieza todavía no se generó, simplemente no la incluyas: el ZIP nunca debe fallar por una pieza faltante.

— VERIFICACIÓN

Descargá el ZIP de la propiedad de prueba, abrílo, y mostrame la lista de lo que tiene adentro.

Modelos, límites y costos

Los números concretos con los que trabaja el sistema. Útil cuando algo falla y hay que entender por qué.

Modelos de IA — todos por OpenRouter

PARA QUÉ	MODELO	AJUSTES	NOTA
Copies de venta	google/gemini-2.5-flash	temp 0.85	Salida estructurada, seis campos requeridos
Guion de reel	google/gemini-2.5-flash	temp 0.75	Una frase por escena
Clasificar fotos	google/gemini-2.5-flash	temp 0.2 · 800 tok	Todas las fotos en una sola llamada
Extraer del portal	google/gemini-2.5-flash	estructurada	Sobre el texto que trae Firecrawl
Generar imágenes	google/gemini-3.1-flash-image-preview	90 s máx.	Imagen a imagen; foto primero, logo después, texto al final
Copies de anuncios	Modelo de escritura fuerte	temp 0.9 · 8000 tok	Todos los copies del lote en una llamada

UN DETALLE QUE HACE PERDER UNA TARDE

Las cabeceras de la petición a OpenRouter tienen que ser solo ASCII. Un guion largo (—) en el título de la aplicación rompe `fetch` con un error de `ByteString` que no menciona la cabecera por ningún lado.

Límites de petición

SERVICIO	LÍMITE	CÓMO LO MANEJA EL SISTEMA
OpenRouter · imágenes	429 con más de 3 en paralelo	Concurrencia fija en 3 y tope de 60 creativos por lote
Meta · nivel desarrollo	60 puntos / 5 min bloqueo 300 s	Registro de puntos con tope propio de 50 y espera antes de llamar
Meta · nivel estándar	9000 puntos / 5 min bloqueo 60 s	Tope propio de 8000; el nivel se detecta de las cabeceras
Meta · costo por llamada	1 lectura · 3 escritura	Pausa de 200 ms entre llamadas, encoladas en serie
Meta · reintentos	4 máximo	Solo ante códigos de saturación; nunca ante token vencido
Instagram · carrusel	10 fotos	Validado antes de enviar

Galería	10 fotos	Constante exportada
Subida de video	1 gb	Límite del cuerpo de petición elevado en la configuración

Códigos de error de Meta que conviene reconocer

CÓDIGO	QUÉ SIGNIFICA	QUÉ HACER
4 · 17 · 32	Saturación	Reintentar con espera creciente
613	Límite de la cuenta publicitaria	Reintentar más espaciado
80000 a 80014	Límite por caso de uso	Reintentar; revisar si hay procesos en paralelo
190 · 102	Token vencido o inválido	No reintentar. Renovar el token

Qué cuesta usarlo

Los textos son prácticamente gratis. El costo real está en las imágenes, a unos **USD 0.04 cada una**.

OPERACIÓN	IMÁGENES	COSTO APROXIMADO
Copies de una propiedad	0	menos de USD 0.01
Post y story	0	gratis (se dibujan localmente)
Ficha técnica en PDF	0	gratis
Carrusel de 6 slides	6	~ USD 0.24
Lote de 12 creativos	12	~ USD 0.50
Una propiedad trabajada entera	~18	~ USD 0.75

Con USD 5 de saldo hacés más o menos seis propiedades completas. El editor de reels no suma costo de IA por imagen, pero sí consume caracteres de ElevenLabs.

Las nueve trampas

Todas aparecieron construyendo este sistema. Ninguna es obvia y varias no dan error: simplemente el resultado sale mal. Los prompts de esta guía ya las previenen — esto es para cuando aparezcan igual.

01 Las imágenes salen perfectas en tu máquina y sin una sola letra en la de tu cliente.

La librería de canvas no trae fuentes propias. Cuando el código pide "Segoe UI" y esa fuente no está instalada, no falla ni avisa: dibuja el texto en blanco absoluto.

QUÉ PEDIR

Que guarde un .ttf dentro del proyecto y lo registre al arrancar. Que ninguna fuente del sistema se nombre en el código.

02 Tu marca es verde pero los precios salen dorados.

Es el color que quedó escrito a mano mientras se probaba el diseño. Sobrevive a todos los cambios de marca porque nadie recuerda que está ahí.

QUÉ PEDIR

Que ningún color de acento esté fijo, y que el color del texto encima se calcule por luminancia del fondo.

03 El carrusel sale con "22px" y "78% opacity" escritos sobre la foto.

El modelo de imágenes no distingue entre el texto que querés que dibuje y las instrucciones de cómo dibujarlo. Si van juntos, dibuja las dos cosas.

QUÉ PEDIR

Texto a renderizar aislado entre comillas, especificaciones en una línea aparte, y una regla global que prohíba dibujar medidas.

04 Una fecha en pantalla tira "Hydration failed" y la página parpadea.

El formateador del servidor mete un espacio invisible distinto al del navegador. El texto se ve idéntico pero no lo es, y React lo trata como error.

QUÉ PEDIR

Nada de `toLocaleString` ni `Intl` en pantalla. Formato de fecha armado a mano.

05 Todo anda hasta que la carpeta se sincroniza con la nube.

El sincronizador toma bloqueos cortos sobre los archivos justo cuando la aplicación los está reemplazando. Falla una de cada veinte veces, siempre en distinto lugar.

QUÉ PEDIR

Escritura atómica: archivo temporal, después renombrar, con reintento y espera creciente, y copia como último recurso.

06 El QR de WhatsApp no lleva a ningún lado.

Un número cargado como "+54 11..." al que se le vuelve a anteponer el código de país queda como "+5454 11...". El QR se genera igual y se ve bien.

QUÉ PEDIR

Normalizar el teléfono antes de armar el enlace, respetando el "+" cuando ya está, y no dibujar el QR si no quedan dígitos válidos.

07 La ficha en PDF tiene una segunda página casi vacía.

Una descripción larga empuja el contenido. El PDF se genera bien, solo que con una hoja de más que arruina la impresión.

QUÉ PEDIR

Alto fijo de página, recorte de la descripción por líneas, y forzar explícitamente el rango de páginas a la primera al exportar.

08 Meta te bloquea a los treinta segundos de probar.

En nivel desarrollo la cuota es de 60 puntos cada 5 minutos y una escritura cuesta 3. Crear una campaña completa la consume entera, y el bloqueo dura 300 segundos.

QUÉ PEDIR

Control de cuota por puntos en el cliente, con tope propio por debajo del real y espera automática antes de llamar.

09 Te dice "listo, funciona" y no funciona.

Que el proyecto compile no dice nada de una aplicación cuyo resultado son imágenes y PDFs. Una foto en blanco compila perfecto.

QUÉ PEDIR

Que abra el archivo generado y te lo muestre. "Compila" no es una verificación; "acá está la imagen y se ve el texto" sí.

Checklist final

Recorrelo con una propiedad real de tu cartera, no con datos de prueba. Si los diez pasos pasan, el sistema está terminado.

01 Cargá una propiedad con seis fotos reales

Aparece en el listado con su portada y sus datos completos.

02 Revisá las categorías que asignó a las fotos

La cocina dice cocina. Corregí una a mano y confirmá que la corrección queda.

03 Cambiá el color de tu marca y regenerá el post

El precio y las etiquetas cambian de color. Si siguen dorados, es la trampa 02.

04 Abrió el post desde la carpeta, no desde la aplicación

Buscá el archivo en [data/](#) y abrílo con el visor de imágenes. El texto tiene que verse. Es la prueba de la trampa 01.

05 Escaneá el QR con el teléfono

Abre WhatsApp con tu número correcto y un mensaje ya escrito sobre esa propiedad.

06 Abrió la ficha en PDF y contó las páginas

Una sola. Probá también con una descripción larga.

07 Mirá los slides del carrusel de cerca

Ningún número de tamaño escrito encima de la foto, y la jerga es la de tu país.

08 Generá cuatro creativos y leé los copies

Los cuatro son distintos entre sí y cada uno se nota escrito desde su ángulo.

09 Abrió el CSV de creativos en Excel

Los acentos se ven bien. Si salen símbolos raros, falta la marca de orden de bytes.

10 Descargá el ZIP completo

Todas las piezas adentro, con nombres legibles. Eso es lo que le mandás al cliente.

LO ÚLTIMO

El sistema no se vuelve valioso cuando tiene los quince módulos. Se vuelve valioso el día que dejás de abrir el editor de diseño para publicar una propiedad.