Jano.

Skills para crear y probar prompts empresariales

Introducción

Con estas dos skills, Claude te arma la primera versión del prompt de un agente a partir de lo que te mandó tu cliente, con la estructura que viste en el video, y después la prueba y saca versiones nuevas. **prompt-empresarial** escribe la v1, el mensaje para pedirle al cliente los datos que faltan y los casos para probarla. **iterar-prompt** la prueba contra el modelo o en un flujo de n8n, la arregla y saca una versión nueva cada vez que el cliente responde o algo falla. En esta guía tenés cómo instalarlas, cómo pedirle cada cosa, qué revisar y el ejemplo de la veterinaria del video.

Importante

Las skills te dan una primera versión sólida, no la versión final. Revisala siempre sección por sección y probala: vos conocés al cliente y sabés qué no puede pasar nunca.

Sección 1: Qué hace cada skill

Una skill es un paquete de instrucciones que Claude sigue cuando le pedís algo de ese tema. Claude elige cuál usar según lo que le pedís:

1. **prompt-empresarial, la v1.** Le pasás el material del cliente (las respuestas de un formulario, tus notas de la llamada, una transcripción) y te devuelve el prompt, el mensaje para pedirle al cliente lo que falta, el campo del mensaje del usuario, la lista de herramientas para n8n y los casos de prueba. No te pregunta datos del negocio: lo que falta va al mensaje. También revisa un prompt que ya tenés y te dice qué le falla.
2. **iterar-prompt, la prueba.** Le manda cada caso a un modelo de OpenAI, como lo haría n8n, o directo a un flujo de n8n, y te dice cuáles pasaron y cuáles no, con la causa.
3. **iterar-prompt, la versión nueva.** Le pegás lo que respondió el cliente o lo que encontró quien probó el agente, como venga. Guarda la versión anterior, cambia solo lo que hace falta, suma los casos, la prueba entera y arregla lo que falle, hasta tres vueltas, sin pedirte permiso. Al final te da un solo resumen con lo que tenés que decidir y, si algo sigue faltando, el mensaje para el cliente.

Todo lo de un negocio queda en una carpeta con su nombre:

1. prompt.md: la versión vigente. Es solo el prompt.
2. prompt-v1.0.md, prompt-v1.1.md…: las versiones anteriores.
3. versiones.md: una línea por versión, con qué cambió y por qué.
4. casos.md: los casos de prueba.
5. pruebas/: un reporte por cada prueba.

Sección 2: La estructura que usa

Es la del video. Cada numeral marca la prioridad: # es una sección principal, ## va

adentro de una sección y ### es información de consulta.

```
# Rol
# Tareas
### Información del negocio
# Tono y estilo
# Mensaje de presentación
# Proceso de conversación
## Paso 1 · …
## Paso 2 · …
# Ejemplos
# Importante
```

1. **Rol:** quién es, de qué negocio y por qué canal atiende.
2. **Tareas:** lo que sí hace, concreto, y al final lo que no es su función.
3. **Información del negocio:** lo fijo, como la dirección y los horarios, solo con datos que dio el cliente. Lo que cambia seguido va en una base que el agente consulta con una herramienta.
4. **Tono y estilo:** cómo habla, escrito en el mismo registro que querés que use.
5. **Mensaje de presentación:** el primer mensaje exacto y qué hacer si la persona ya trae una consulta.
6. **Proceso de conversación:** los pasos, cada uno con qué preguntar, en qué variable guardarlo y cuándo avanzar. Adentro se activan las herramientas.
7. **Ejemplos:** uno correcto y uno incorrecto del comportamiento que más cuesta. Los ejemplos mandan sobre las reglas: cada uno tiene que cumplirlas todas.
8. **Importante:** al final, pocas reglas que nunca se pueden romper.

La versión de prueba. En la v1 las herramientas todavía no existen en n8n. Por eso la última regla de Importante le dice al agente que, donde usaría una, escriba una etiqueta con su nombre y los datos, por ejemplo [CONSULTAR_DISPONIBILIDAD] servicio: vacunación | fecha: 2026-10-05. Así probás la conversación sin armar nada. Cuando conectás las herramientas, borrás esa regla: esa es la v2.0.

Sección 3: Instalarlas en Claude Code

1. Descomprimí prompts-con-claude.zip. Te queda la carpeta prompts-con-claude, con las dos skills adentro (en la carpeta oculta .claude/skills), la carpeta veterinaria, con el material del video, y agente-de-prueba.json, el flujo para probar en n8n.
2. Abrí la app de Claude, entrá a la pestaña **Code** y abrí una sesión nueva en la carpeta prompts-con-claude. Si usás Claude Code en la terminal, entrá a esa carpeta y escribí claude. Claude Code necesita un plan pago de Claude.
3. Para cada cliente, creá adentro de prompts-con-claude una carpeta con el nombre del negocio y poné ahí lo que te mandó: el formulario, tus notas o la transcripción, en un archivo de texto.
4. Si las querés en todos tus proyectos, copiá las carpetas prompt-empresarial e iterar-prompt de .claude/skills a la carpeta .claude/skills de tu carpeta de usuario. Para ver las carpetas ocultas en el Finder, tocá **Cmd + Shift + punto**; en Windows, en el Explorador, Vista > Mostrar > Elementos ocultos.

Sección 4: Usarlas en Claude, en el navegador

En Claude en el navegador, las skills arman la v1, el mensaje y los casos, y sacan versiones nuevas. Las pruebas, contra el modelo o en n8n, necesitan Claude Code.

1. Las skills funcionan en los planes Free, Pro, Max, Team y Enterprise, con la ejecución de código activada. Entrá a **Settings > Capabilities** y fijate que esté prendida **Code execution and file creation**. En un plan Team o Enterprise, la activa quien administra la organización.
2. Usá prompt-empresarial.zip e iterar-prompt.zip tal como vinieron: adentro de cada uno tiene que estar la carpeta de la skill, no los archivos sueltos.
3. Andá a **Customize > Skills**, tocá **+**, después **Create skill** y **Upload a skill**, y elegí un .zip. Repetilo con el otro.
4. Abrí un chat nuevo, adjuntá el material de tu cliente y pedile la v1.

Los nombres de los menús son los de Claude en inglés. Si lo tenés en español, están en el mismo lugar con su traducción.

Sección 5: Cómo pedirle la v1

Cuanto más material le des, menos queda para el mensaje al cliente. Tres pedidos que funcionan:

> armá la v1 del prompt del agente de la veterinaria con lo que hay en veterinaria/material.md

> armá la v1 del prompt del agente de WhatsApp de la panadería. El formulario y mis notas están en panaderia/. Las herramientas en n8n van a ser "Ver productos" y "Registrar pedido".

> revisá este prompt con la estructura y decime qué le falla: [pegás el prompt]

Si tu material responde estas cinco preguntas, la v1 sale casi completa:

1. ¿Qué tiene que lograr el agente en cada conversación?
2. ¿Cuál es el peor error que podría cometer?
3. ¿Cómo habla el mejor empleado del negocio en ese canal?
4. ¿Cuáles son los mensajes que más recibe el negocio?
5. ¿Dónde termina el agente y empieza una persona del equipo?

Sección 6: Qué revisar en la v1

1. Rol arriba, Importante al final, y un proceso con pasos que dicen qué preguntar, dónde guardarlo y cuándo avanzar.
2. Ningún dato inventado: compará precios, horarios, direcciones y teléfonos con lo que te dio el cliente. Lo que falta tiene que estar en el mensaje, no en el prompt.
3. Cada herramienta con su nombre exacto entre comillas, cuándo se usa, con qué datos y qué hacer con lo que devuelve, también si no encontró nada o si dio error.
4. Los ejemplos cumplen todas las reglas.
5. En Importante: la pregunta fuera de tema (el huevo frito), no inventar, la fecha de hoy con {{ $now }}, los formatos técnicos que resuelve el agente, la regla de honestidad, las reglas del rubro y, en la v1, la regla de la versión de prueba.

6. Si algo no te cierra, no lo corrijas a mano: pedíselo a Claude con la sección y el porqué. Así queda como versión nueva.

Sección 7: Probarla

La prueba de verdad es en Telegram, primero con alguien de tu equipo y después con tu cliente. Antes, Claude la prueba: así no llega con errores que se ven en un minuto. Hay dos maneras, y las dos dejan el mismo reporte en pruebas/:

1. **Por API:** Claude manda cada caso a gpt-4.1 con tu prompt, la fecha de hoy en lugar de {{ $now }} y cada mensaje entre comillas, como lo haría n8n. Es la más rápida y no necesita nada armado.
2. **En el flujo de n8n:** Claude sube el prompt a un flujo de prueba de tu n8n y le escribe por su webhook, caso por caso, con una conversación nueva en cada uno. Es la prueba más parecida a la real: el agente de n8n, su memoria y su fecha.

En las dos, Claude lee las respuestas y te arma una tabla: caso, qué se esperaba, qué pasó, OK o FALLA. Si un caso falla, te dice la causa: falta la regla, la regla es ambigua, el modelo encontró una excepción o es un caso nuevo. Pedile arreglalo y saca una versión nueva, la prueba otra vez y repite hasta tres vueltas. Las claves nunca van en el chat: las guardás vos con un comando, en el archivo .env de la carpeta.

Por API

Necesitás una API key de OpenAI con crédito cargado (se crea en platform.openai.com, en API keys, y se ve una sola vez).

1. Pedile a Claude probá la v1 con los casos. Si todavía no tiene tu API key, te da el comando para correr vos.
2. En la terminal de la sesión, desde la carpeta prompts-con-claude, corré:

   ```
   python3 .claude/skills/iterar-prompt/scripts/probar.py conectar
   ```

   Pegá la key y tocá Enter: no se ve mientras la pegás. Tiene que decir **Listo: conexión OK**. En Windows, usá python en lugar de python3.
3. Avisale a Claude: listo, probala.

En el flujo de n8n

El flujo **Agente de prueba** tiene cuatro piezas: un Webhook que recibe cada mensaje, un AI Agent con gpt-4.1 a temperatura 0, una memoria que separa cada conversación y la respuesta del webhook. Viene sin credenciales y sin prompt: el prompt lo sube Claude.

1. En n8n, creá un workflow nuevo y, en el menú de los tres puntos, elegí **Import from File** y el archivo agente-de-prueba.json. Fijate que se llame **Agente de prueba**; si no, cambiale el nombre arriba a la izquierda.
2. Abrí el nodo **OpenAI Chat Model** y, en **Credential**, elegí tu credencial de OpenAI o creá una con tu API key.
3. El flujo viene con la zona horaria de Buenos Aires. Si tu cliente está en otra, cambiala en los ajustes del flujo, en **Settings > Timezone**.
4. Guardá y tocá **Publish**. Si no está publicado, el webhook no responde.
5. Creá una API key de n8n en **Settings > n8n API** y copiala. En n8n Cloud, la API no está disponible en la prueba gratis.
6. En la terminal de la sesión, corré:

```
python3 .claude/skills/iterar-prompt/scripts/probar.py conectar-n8n
```

   Escribí la dirección de tu n8n, como la ves en el navegador, y pegá la API key. Tiene que decir **Listo: conexión OK con tu n8n**.
7. Pedile a Claude probala en el flujo. Sube el prompt al agente, en modo Expression para que {{ $now }} se reemplace por la fecha, y corre los casos.

Mientras corre, en n8n vas a ver llegar las ejecuciones del flujo, una por mensaje.

Si dice que no tiene crédito: cargalo en tu cuenta de OpenAI, en Billing. Con gpt-4.1, una prueba de unos veinte casos cuesta menos de un dólar.

Si dice que rechazó la API key (401): creá una nueva y corré otra vez conectar o conectar-n8n.

Si dice que el flujo no está publicado o que falló por credenciales: abrilo, elegí la credencial de OpenAI en el nodo del modelo, guardá y tocá Publish.

Si querés probar con otro modelo: por API, pedíselo a Claude, por ejemplo "probala con gpt-4.1-mini". En el flujo, cambialo en el nodo del modelo.

Sección 8: Versiones nuevas

1. Cuando el cliente responde, pegale a Claude lo que mandó, como venga (el WhatsApp, un audio transcripto, tus notas), y pedile la versión nueva. Cuando un caso falla, pedile arreglalo.
2. Claude guarda la versión anterior (por ejemplo prompt-v1.0.md), toca solo las secciones que cambian y anota la versión en versiones.md.
3. Suma un caso por cada cambio, con lo que no tiene que pasar, y revisa los que ya estaban, por si el cambio les cambia el camino.
4. Prueba entera la versión nueva, no solo lo que cambió. Si algo falla, lo arregla y vuelve a probar, hasta tres vueltas. Si sigue fallando, te lo dice primero.
5. Te responde con un solo resumen: la versión, la tabla de la prueba, qué cambió, lo que tenés que decidir y, si sigue faltando algo, el mensaje para tu cliente.

Si la versión nueva sale peor, volvés a la anterior: está guardada.

Sección 9: El ejemplo del video

La carpeta veterinaria trae lo que mandó la clienta de una veterinaria inventada: las respuestas del formulario y las notas de la llamada. Para repetir la clase:

1. Pedí la v1: armá la v1 del prompt del agente de la veterinaria con lo que hay en veterinaria/material.md. El mensaje para la clienta pide, entre otras cosas, las formas de pago y si van a domicilio.
2. Probala: probá la v1 con los casos, o probala en el flujo.
3. Pedí la versión nueva con la respuesta de la clienta:

```
la clienta respondió esto, sacá la versión nueva:
- No vamos a domicilio.
- Aceptamos efectivo, débito, crédito y transferencia.
- Y algo que me olvidé: para vacunar tienen que traer la libreta de vacunas. Si no la tienen, que saquen primero una consulta clínica.
```

Los datos van a Información del negocio y la libreta, al paso del motivo: si el servicio es vacunación, el agente pregunta por la libreta y, si no la tiene, ofrece primero una consulta clínica. Claude la prueba entera, con los casos de antes y los nuevos, como el de la vacuna sin libreta.

El texto exacto que escribe Claude y lo que responde el modelo cambian de una vez a la otra. Lo que no cambia es la estructura y lo que tiene que pasar en cada caso.

Sección 10: Cuando la pases a n8n

1. Si tu n8n está conectado, pedile a Claude subilo al flujo de [nombre del flujo]: pone el prompt en el System Message del agente, en modo Expression. Si el flujo está publicado, desde ese momento responde con la versión nueva. Si preferís pegarlo a mano, pedile el prompt para copiar.
2. A mano, el prompt va en el nodo AI Agent, en **Options > System Message**, con el campo en modo **Expression**. Si queda en Fixed, {{ $now }} le llega como texto y el agente no sabe qué día es.
3. La zona horaria: n8n usa la de Nueva York si no le decís otra. Configurá la del negocio en los ajustes del workflow, en **Settings > Timezone**.
4. El mensaje de la persona va en **Prompt (User Message)**, con **Source for Prompt** en **Define below** y el texto entre comillas, como te lo da la skill.
5. Cada herramienta tiene que existir en n8n con el mismo nombre que en el prompt. Usá nombres sin tildes: n8n le pasa el nombre al modelo con guiones bajos en lugar de espacios, y una letra con tilde también se convierte en guion bajo.
6. Cuando las herramientas estén conectadas, pedile a Claude la v2.0: borra la regla de la versión de prueba. Para probarla por API, Claude le agrega esa regla a una copia del prompt, así sigue viendo en qué momento usaría cada herramienta.

Sección 11: Los comandos

Los dos primeros los corrés vos, en la terminal, desde la carpeta prompts-con-claude. Los otros los corre Claude cuando se lo pedís, pero también los podés correr vos.

```
python3 .claude/skills/iterar-prompt/scripts/probar.py conectar
python3 .claude/skills/iterar-prompt/scripts/probar.py conectar-n8n
python3 .claude/skills/iterar-prompt/scripts/probar.py veterinaria
python3 .claude/skills/iterar-prompt/scripts/probar.py subir veterinaria
python3 .claude/skills/iterar-prompt/scripts/probar.py veterinaria --flujo
```

1. conectar: guarda tu API key de OpenAI.
2. conectar-n8n: guarda la dirección y la API key de tu n8n.
3. veterinaria: prueba los casos por API.
4. subir veterinaria: sube prompt.md al flujo Agente de prueba.
5. veterinaria --flujo: prueba los casos en el flujo.

 Usalo libre en tus proyectos y con tus clientes; si lo compartís, nombrá la fuente.