Guía de desarrollador · bidding-agent

bidding-agent: guía técnica

Cómo analiza el agente los cuatro documentos de una licitación y construye la propuesta, la estimación, las consideraciones técnicas, el análisis de IA y los dos modelos de costos.

3 nov 2026 | Guía de desarrollador | Claude Code | 19 min de lectura

Descripción general

El **Agente Licitatorio** (`bidding-agent`) es el especialista en licitaciones, preventa técnica, arquitectura de soluciones y estimación de proyectos de software de Campuslands Full Service. A partir de la documentación oficial de un proceso, lo analiza de forma integral y produce **cuatro entregables**: la propuesta técnica, la estimación de horas, las consideraciones técnicas y el análisis de inteligencia artificial, además de los soportes (análisis documental, matriz de cumplimiento y los dos modelos de costos llenados y conciliados).

Esta guía explica cómo está construido: los insumos y su jerarquía, el flujo de tres fases y catorce pasos, las cinco habilidades que lo componen, las reglas duras, la verificación cruzada y cómo mantenerlo. Está pensada para quien mantiene el agente.

4 documentos oficiales de entrada por licitación	4 entregables, más los soportes y los dos modelos de costos
14 pasos en tres fases: ingesta, entregables y verificación	1 matriz de cumplimiento con una fila por requerimiento, que actúa como control de entrega

LO ESENCIAL

La propuesta debe ser **defendible ante el comité evaluador**: cada afirmación se rastrea a un documento oficial. Por eso la fuente de verdad es solo la documentación suministrada, nada se inventa, y **las aclaraciones prevalecen sobre el RFP**: cotizar un alcance que las aclaraciones ya modificaron es el error más costoso del proceso.

Qué hace y qué no hace

El agente hace	El agente no hace
Lee los cuatro documentos completos antes de escribir nada	Generar un entregable con una lectura parcial
Numera todos los requerimientos con su cita de origen	Inventar alcance, cifras, plazos, certificaciones ni capacidades
Declara `Cumple parcialmente` y `No cumple` en lugar de disimularlos	Marcar todo como «Cumple» para que la matriz «se vea bien»
Reporta con números si el plazo no es factible	Ajustar la estimación hacia abajo para que el cronograma cuadre
Llena ambos modelos de costos y los concilia	Entregar un modelo sin el otro, ni una diferencia sin explicar
Afirma por escrito si el alcance requiere IA	Dejar «no se mencionó» como conclusión
Pregunta ante ambigüedad bloqueante, con las preguntas agrupadas	Resolver por su cuenta una contradicción que cambia alcance o precio

Cómo corre: Claude Code y `CLAUDE.md`

Corre en **Claude Code** y se gobierna con `CLAUDE.md`, que define su rol, los insumos, las decisiones fundacionales, la jerarquía documental, el flujo y las reglas duras. Los procedimientos reutilizables viven en `habilidades/` como cinco skills, y el conocimiento acumulado entre licitaciones, en `wiki/`.

Decisiones fundacionales

Tema	Decisión	Por qué
Fuente de verdad	Solo la documentación suministrada	La propuesta se defiende ante el comité: cada afirmación se rastrea a un documento

Tema	Decisión	Por qué
Veracidad	Cero invención de alcance, cifras, plazos o capacidades	Un dato inventado es una descalificación o un incumplimiento contractual
Cobertura	Cero requerimientos omitidos	Todo requerimiento del RFP y de las aclaraciones aparece en la matriz con su respuesta
Trazabilidad	Cada ítem cita su origen (`RFP §4.2` , `Aclaración P-17`)	Permite auditar y responder al comité sin rehacer el análisis
Entregables	Los cuatro; ninguno es opcional	Es lo que el proceso licitatorio exige
Doble modelo	Se llenan ambos y se concilian	El oficial se radica; el de cotización calcula el precio real
Salida	Carpeta propia por licitación	Autocontenida, como en los demás agentes

Los documentos de entrada

El usuario deposita los insumos en `recursos/licitaciones/<slug-licitacion>/` :

#	Documento	Qué aporta	Obligatorio
1	RFP (*Request for Proposal*)	Alcance, requerimientos funcionales y no funcionales, plazos, criterios de evaluación y condiciones contractuales	Sí
2	Aclaraciones o preguntas y respuestas	Correcciones, precisiones y **cambios de alcance** emitidos durante el proceso	Sí
3	Modelo de costos **oficial**	Formato, líneas y reglas de costeo que exige la empresa solicitante	Sí
4	Modelo de costos de **cotización**	El modelo interno con el que Campuslands calcula el precio real (típicamente la plantilla de FullService)	Sí

Si falta alguno, el agente lo reporta y pregunta antes de construir. No supone el contenido de un documento ausente; si el usuario confirma que no existe en ese proceso, queda registrado en el análisis y en `wiki/log.md`.

La jerarquía documental

Cuando dos documentos se contradicen, este es el orden de precedencia:



Toda contradicción detectada se registra en el análisis con la cita de **ambos** documentos y la lectura adoptada. Si la contradicción **cambia materialmente el alcance o el precio**, el agente no la resuelve: se la reporta al usuario y pregunta.

Estructura del repositorio

```
bidding-agent/
  CLAUDE.md                       # rol, jerarquía, flujo y reglas duras
  habilidades/                    # una carpeta por skill, con su SKILL.md
    ingesta-documentos-licitacion/
    matriz-cumplimiento/
    estimacion-horas/
    modelos-costos/
    analisis-ia-tokens/
  recursos/                       # solo lectura
    licitaciones/<slug>/          # los 4 documentos oficiales de cada proceso
  licitaciones/
    <slug-licitacion>/            # una carpeta por proceso
      00 - Analisis Documental.md
      01 - Matriz de Cumplimiento.xlsx
      02 - Propuesta Tecnica.pdf
      03 - Estimacion de Horas.xlsx
      04 - Consideraciones Tecnicas.md
```

```
    05 - Analisis IA y Tokens.xlsx  # solo si el alcance requiere IA
    06 - Modelo de Costos Oficial.xlsx
    07 - Modelo de Costos Cotizacion.xlsx
    verificacion-cruzada.md
wiki/
  index.md · log.md             # catálogo y bitácora
  flujo-trabajo.md · perfil-usuario.md
  estructura-propuesta-tecnica.md · estimacion-horas.md
  consideraciones-tecnicas.md · costeo-ia-tokens.md
  marca-campuslands.md · plantilla-verificacion-cruzada.md
  licitaciones/<slug>.md        # ficha por licitación
```

DÓNDE VIVE EL AGENTE Y SUS RECURSOS COMPARTIDOS

El agente está en `agents/claude-agents/bidding-agent/`. Los logos, el isotipo, el favicon, el `Brief Fullservice.pdf` y el Brandbook de Campuslands ya no se copian dentro de cada agente: están en `resources/campuslands-resources/`, en la raíz del repositorio (desde la carpeta del agente: `../../../resources/campuslands-resources/`).

Las reglas de carpetas: `wiki/`, `habilidades/` y `licitaciones/` son del agente; **`recursos/` es solo lectura**. Los documentos oficiales **nunca se editan**: el modelo oficial se **copia** a `06 - Modelo de Costos Oficial.xlsx` y se trabaja sobre la copia. Un slug es minúsculas, sin espacios ni tildes y lleva el identificador del proceso cuando existe (por ejemplo, `ipa-pr10598`).

El flujo de trabajo



Fase	Paso	Qué ocurre
1 · Ingesta y análisis	1	Verifica que estén los cuatro documentos; si falta alguno, pregunta
	2	Lee los cuatro completos, en orden: RFP, aclaraciones, modelo oficial, modelo de cotización
	3	Extrae y numera todos los requerimientos con su cita
	4	Construye `00 - Analisis Documental.md`
	5	Consulta al usuario todas las ambigüedades bloqueantes
2 · Los entregables	6	Matriz de cumplimiento (`01`), **antes** de la propuesta
	7	Propuesta técnica (`02`)
	8	Estimación de horas (`03`)
	9	Consideraciones técnicas (`04`), con la conclusión explícita sobre IA
	10	Análisis de IA y tokens (`05`), solo si el paso 9 concluyó que se requiere IA
	11	Modelos de costos (`06` y `07`), llenados y conciliados
3 · Verificación y entrega	12	Verificación cruzada (`verificacion-cruzada.md`)
	13	Entrega de los archivos en `licitaciones/<slug>/`
	14	Actualiza `wiki/index.md` , la ficha de la licitación y `wiki/log.md`

Las cinco habilidades

1. `ingesta-documentos-licitacion`

Es **siempre el primer paso** de toda licitación nueva y se re-ejecuta al recibir una adenda o unas aclaraciones adicionales. No se usa para retomar una licitación cuyo análisis ya está cerrado y aprobado.

Numeración de requerimientos. Cada requerimiento lleva un identificador estable que se usa en todos los entregables, su texto resumido pero fiel, su **cita de origen** (`RFP §4.2` , `Aclaración P-17` , `Anexo Técnico 3 p.12`) y si es obligatorio o deseable.

Prefijo	Categoría
`RF-nn`	Funcional
`RNF-nn`	No funcional: rendimiento, disponibilidad, seguridad, escalabilidad, usabilidad
`TEC-nn`	Técnico: stack, integraciones, infraestructura, migración
`PLZ-nn`	De plazo: fechas, hitos y entregables intermedios
`ADM-nn`	Legal o administrativo: pólizas, certificaciones, experiencia acreditable
`ENT-nn`	De entregables: documentación, capacitación, transferencia
`OPS-nn`	De operación y soporte: SLA, mesa de ayuda, garantía, mantenimiento

El análisis documental (`00`) tiene nueve secciones: datos del proceso (con el objeto transcrito literalmente), inventario de documentos recibidos, inventario de requerimientos, plazos e hitos, criterios de evaluación y su ponderación, estructura de los dos modelos de costos, contradicciones y precedencias, supuestos con su justificación documental y ambigüedades a consultar, con su impacto en alcance, precio o plazo.

Errores a evitar: cotizar contra el RFP sin haber leído las aclaraciones; leer solo los encabezados de los modelos de costos sin mapear su estructura real de celdas y fórmulas; omitir requerimientos «administrativos» o de entregables por no ser desarrollo (también consumen esfuerzo y deben costearse); y empezar la propuesta antes de cerrar el análisis.

2. `matriz-cumplimiento`

Se ejecuta **inmediatamente después del análisis documental y antes de la propuesta técnica**: construirla primero obliga a diseñar la solución contra la lista completa de requerimientos, en lugar de descubrir al final qué quedó afuera. Es el gate de «cero requerimientos omitidos».

Columna	Contenido
ID	Identificador estable
Requerimiento	Texto resumido pero fiel
Origen	Cita exacta
Obligatorio / Deseable	Según lo declare el propio documento
Cumplimiento	`Cumple`, `Cumple parcialmente`, `No cumple` o `No aplica`
Cómo se cumple	Descripción concreta de la solución, sin lenguaje vago
Módulo / sección	Dónde se responde en la propuesta
Esfuerzo asociado	Línea de la estimación de horas, o `N/A` con razón
Observaciones	Supuestos, dependencias y riesgos

Las reglas de llenado: una fila por requerimiento **sin excepciones** (ninguna categoría se omite por «no ser desarrollo»); `Cumple parcialmente` y `No cumple` **se declaran, no se disimulan**, para que el usuario decida entre negociarlo por aclaraciones, buscar un aliado o desistir; `No aplica` siempre lleva razón escrita; «se contempla en la solución» no es cumplimiento; y todo `Cumple` tiene módulo y esfuerzo asociado.

Al cierre, un resumen de cobertura cuenta los totales por estado. Si hay **requerimientos obligatorios en `No cumple`, es un bloqueante**: se reporta al usuario antes de avanzar, porque puede significar que el proceso no es viable tal como está planteado.

3. `estimacion-horas`

Se ejecuta cuando la propuesta tiene sus módulos definidos y la matriz está llena, y se repite al re-estimar tras una adenda.

1. **Confirmar el factor horas↔días.** El entregable va en horas; el modelo interno, en días. Si el RFP define jornada, manda el RFP; si no, el factor confirmado por el usuario (típicamente 8 h = 1 día hábil). Si no está definido, **se pregunta** y el factor queda declarado en el archivo.
2. **Descomponer en ítems atómicos.** Una fila que solo describe la anterior no lleva horas: nunca se duplica el mismo esfuerzo.
3. **Estimar por especialidad.** Las mismas del modelo interno: `DB`, `UX`, `FM`, `BS`, `BM`, `QA`, `IM`, `MS`, `MM` e `IA`. Solo se llenan las que participan (no en 0); el piso es el equivalente a 0,5 días; y se estima con criterio de desarrollador real, nunca con un valor genérico.
4. **Incluir lo transversal y lo que no es desarrollo:** estructura del proyecto, UX base, implementación, estructura del componente de IA (si aplica), documentación, capacitación, migración de datos, pruebas exigidas, acompañamiento en producción y los meses de operación incluidos. **Un requerimiento sin horas asignadas es una omisión.**
5. **Respetar estimaciones de la fuente.** Si el RFP trae horas o días por actividad, esas mandan. Si el documento es internamente inconsistente (el detalle no suma lo que dice el resumen), no se elige: se reportan ambas cifras al usuario.
6. **Validar la factibilidad contra el plazo (obligatorio).**

CÓDIGO

```
Horas totales estimadas            = H
Plazo exigido por el RFP           = P días hábiles   (con cita)
Horas hábiles por persona por día  = J                (factor declarado)
Personas requeridas                = H / (P × J)
```

Se compara con el equipo propuesto. Si no cabe, **se reporta con los números** y las opciones reales (ampliar el equipo, negociar plazo o alcance por aclaraciones, o desistir): no se maquilla la estimación. Se verifica también por especialidad, porque un total que cuadra puede esconder un único arquitecto con 400 horas en 30 días.

4. `modelos-costos`

Se ejecuta después de la estimación y del análisis de IA, porque sus costos entran en ambos modelos.

- **Copiar, nunca editar el original.** El modelo oficial pasa a `06` y el de cotización a `07`; los de `recursos/` son fuentes inmutables.
- **Mapear antes de escribir.** Para cada modelo se identifica qué celdas son de entrada y cuáles de fórmula. Se documenta el mapeo la primera vez en la ficha de la licitación. **No se tocan las fórmulas**; si faltan filas, se insertan y se verifica la propagación de las fórmulas.
- **Modelo oficial (`06`).** Se respeta exactamente el formato que exige la empresa: es el que se radica y un desvío puede descalificar la propuesta. Si pide una desagregación que el modelo interno no produce (por ejemplo, costo por perfil-hora en lugar de especialidad-día), se hace

la conversión y se documenta.

- **Modelo de cotización (`07`).** Es el motor del precio real. Si la plantilla trae hojas de otros clientes o procesos, **se eliminan** antes de trabajar y la hoja se renombra con el identificador de la licitación.
- **Costos que no son desarrollo:** infraestructura, licencias de terceros, servicios recurrentes, consumo de IA (con su margen de contingencia), certificaciones y auditorías, capacitación, meses de operación y soporte, pólizas y garantías, y contingencia explícita y justificada.

Conciliación obligatoria. Se calcula `Total oficial (06) − Total de cotización (07)` y **cada componente de la diferencia queda explicado** (impuestos, margen, líneas que un modelo agrupa y el otro desagrega, redondeos). Una diferencia sin explicar es un error de llenado; si no se puede explicar, se reporta al usuario antes de entregar.

5. `analisis-ia-tokens`

Se ejecuta en **toda** licitación, porque incluso la conclusión «no se requiere IA» debe quedar escrita y fundamentada. El archivo `05` solo se genera si la conclusión es que sí se requiere.

1. **Decidir si se requiere IA**, recorriendo la tabla de señales de `wiki/costeo-ia-tokens.md` (chatbot, clasificación, extracción de documentos, resumen, búsqueda semántica o RAG, traducción, análisis de sentimiento, generación de reportes narrativos, transcripción de audio). Lo que se resuelve con reglas, validaciones, automatización de flujos, reportes tabulares o búsqueda por palabra clave **no es IA**: proponer un LLM donde basta una consulta SQL infla el costo y es indefendible.
2. **Verificar la vigencia de los precios** antes de calcular y registrar la fecha de verificación en el entregable. Si el RFP impone un proveedor, manda el RFP.
3. **Recolectar las variables de volumen**, cada una con su fuente. Si un dato no está en ningún documento, se pregunta; si el usuario indica un supuesto, se declara textualmente como supuesto.
4. **Contar tokens, no estimarlos a ojo.** Para dimensionar, un token equivale a unos 3 a 4 caracteres. Para cifras que van al costeo se usa el endpoint oficial de conteo de tokens. **Nunca** `tiktoken` ni tokenizadores de otros proveedores: subcuentan los tokens de Claude.
5. **Elegir el modelo por actividad** y justificarlo: Haiku para clasificación y alto volumen, Sonnet para la mayoría de cargas productivas y Opus para razonamiento complejo. Mezclarlos es una optimización legítima.
6. **Calcular aplicando caché y *batch*.**

CÓDIGO

```
Costo input = (input_no_cacheado × precio)
            + (input_leído_de_caché × precio × 0.1)
            + (input_escrito_a_caché × precio × 1.25)    [TTL 5 min; 2× si TTL 1 h]
Costo output = output_MTok × precio_output
```

El caché de prompt es determinante en chatbots y RAG (cerca de 90 % de ahorro en esa porción del input), y el *Batch API* da 50 % de descuento en cargas no interactivas. Ignorarlos produce un precio no competitivo.

1. **Incluir los costos de IA que no son tokens:** embeddings y base vectorial, OCR, transcripción de voz, infraestructura de inferencia si se exige modelo local, y observabilidad y evaluación.
2. **Construir `05`:** una fila por actividad, con el requerimiento que cubre, el modelo y su justificación, el volumen con su fuente, los tokens desglosados, el porcentaje cacheable y el costo mensual, en escenarios conservador, esperado y alto.

LOS PRECIOS DE IA CADUCAN

`wiki/costeo-ia-tokens.md` trae una tabla con fecha de **2026-07-29**, que incluye una tarifa promocional de un modelo vigente solo hasta el **2026-08-31**. Antes de usarla en un costeo nuevo hay que confirmar los precios vigentes en la documentación del proveedor, actualizar la tabla y anotarlo en `wiki/log.md`.

La propuesta técnica

`wiki/estructura-propuesta-tecnica.md` define su anatomía. **Si el RFP o las aclaraciones fijan una estructura obligatoria, esa manda** y la plantilla se adapta; si no, se usa esta:

Sección	Contenido
0 · Portada e identificación	Logo, objeto transcrito literalmente, identificador del proceso, entidad, fecha y datos del proponente
1 · Entendimiento del requerimiento	Resumen con citas; debe reflejar las **aclaraciones**, no solo el RFP
2 · Solución por módulos	Un bloque por módulo con sus funcionalidades y los requerimientos que cubre

Sección	Contenido
3 · Arquitectura	Diagrama, stack con justificación documental, modelo de datos, integraciones, despliegue y escalabilidad
4 · Componente de IA	Solo si aplica; remite al entregable `05`
5 · Seguridad y cumplimiento	Autenticación, cifrado, datos personales, auditoría, normas exigidas y continuidad
6 · Metodología	Marco de trabajo, control de cambios, pruebas, riesgos y gobierno
7 · Cronograma	Fases e hitos alineados con los plazos del RFP, con la cita de cada fecha
8 a 11	Equipo, operación y garantía, supuestos y exclusiones, y anexos

Todo módulo debe tener horas en la estimación y toda hora debe pertenecer a un módulo.

Las consideraciones técnicas

`wiki/consideraciones-tecnicas.md` es un checklist de **15 bloques** que se recorre completo en toda licitación: arquitectura y plataforma, infraestructura y despliegue, capacidad y rendimiento, integraciones, datos y migración, seguridad y cumplimiento, **inteligencia artificial (decisión obligatoria)**, experiencia de usuario y accesibilidad, calidad y pruebas, operación y soporte, entregables documentales, licenciamiento y terceros, gestión del proyecto, restricciones de plazo y riesgos técnicos. El entregable `04` cierra con la conclusión explícita sobre IA.

La verificación cruzada

Antes de entregar, el agente completa `verificacion-cruzada.md` a partir de `wiki/plantilla-verificacion-cruzada.md`, con **rastro auditable**, no solo la afirmación de que se hizo:

Comprobación	Qué se confirma
Cobertura	Todo requerimiento del RFP y de las aclaraciones aparece en la matriz con respuesta
Correspondencia	Las horas de `03` corresponden 1:1 con los módulos de `02`

Comprobación	Qué se confirma
Factibilidad	El cronograma cabe en el plazo con el equipo propuesto y las horas estimadas
Conciliación	Los totales de `06` y `07` cuadran y la diferencia está explicada
IA	Los costos de `05` están incluidos en `06` y `07`
Integridad de archivos	Los XLSX recalculan sin `#REF!`, `#DIV/0!` ni `#NAME?`, salvo defectos preexistentes documentados
Veracidad y reserva	Ninguna cifra, plazo o capacidad sin respaldo documental; nada de una licitación pasó a otra

Si en lugar de fórmulas vivas se entregan valores estáticos calculados fuera de Excel, esa decisión queda documentada, incluido cómo regenerar el archivo si se necesita un modelo editable.

Las reglas duras

#	Regla	Qué significa
1	Nunca inventar	Alcance, cifras, plazos, certificaciones y capacidades salen de los documentos o del brief; lo faltante se marca `[PENDIENTE: qué falta]`
2	Cero requerimientos omitidos	Sin fila en la matriz, la propuesta no está lista
3	Toda decisión se fundamenta	Con la cita del documento o la nota de que fue decisión del usuario
4	Las aclaraciones prevalecen	Cotizar contra un RFP ya modificado es el error más caro
5	El plazo es una restricción	Si no es factible, se dice con los números

#	Regla	Qué significa
6	Dos modelos, conciliados	La diferencia entre totales se explica
7	Conclusión explícita sobre IA	«No se mencionó» no es una conclusión
8	Reserva de la información	`recursos/licitaciones/` y `licitaciones/` contienen precios y estrategia: no se mezcla una licitación con otra
9	Estimaciones defendibles	Un número sin método y supuestos no se entrega
10	Nada de relleno comercial	Sin promesas no respaldadas ni métricas inventadas

Cuándo se detiene y pregunta

Un requerimiento admite más de una lectura con impacto en el precio; el RFP y las aclaraciones se contradicen en alcance, plazo o entregables; falta un dato bloqueante para costear (volumen de usuarios, transacciones, meses de operación, residencia de datos, integraciones); el plazo no parece factible; el modelo oficial pide una estructura que no encaja con el interno; o hay que decidir entre arquitecturas con diferencias materiales de costo. Agrupa las preguntas y avanza mientras tanto en todo lo que no dependa de ellas.

Comandos «!»

Los comandos son órdenes cortas del usuario (`!optimize`, `!notime`…) que el agente debe ejecutar de forma obligatoria, siempre dentro de sus reglas duras. Hay un solo catálogo para todos los agentes y de él salen las instrucciones de cada uno, las bitácoras y las tablas de las guías.

Dónde vive cada cosa

Qué	Dónde
Catálogo, regla común y qué comandos tiene cada agente	`resources/comandos/comandos.py` (la fuente única; no se edita a mano nada de lo que genera)

Qué	Dónde
Catálogo legible, matriz por agente y decisiones	`resources/comandos/catalogo.md`
Instrucciones del agente	sección «Comandos (`!`)» de `CLAUDE.md`, entre las marcas `<!-- comandos:inicio -->` y `<!-- comandos:fin -->`
Tabla de esta guía	se genera del mismo catálogo al compilar las guías

La regla común

1. Un comando es una palabra precedida de `!` al inicio de un mensaje o de una línea; el texto que sigue es su objeto.
2. Es obligatorio: el agente no pide permiso ni lo reinterpreta. Solo lo limita una **regla dura** (veracidad, marca, confidencialidad, puertas de aprobación y de verificación); entonces hace lo que la regla permite y dice qué no pudo.
3. Se combinan. Si dos modos chocan, el orden es: reglas duras > `!strict-mode` y `!no-assumptions` > `!think-deep`, `!critical-mode` y `!double-check` > `!notime`.
4. Rige la tarea en curso, no las siguientes.
5. La primera línea de la respuesta nombra los comandos recibidos.
6. Un comando desconocido no se adivina: se propone el más parecido y se pregunta. `!help` lista los disponibles.
7. Si un comando cambió un entregable o la wiki, se anota en `wiki/log.md`.

Cómo añadir o cambiar un comando

1. Edita `CMDS` (definición), `AGENTES` (a quién aplica y su nota) o `ALIAS` en `resources/comandos/comandos.py`.
2. Corre `python -X utf8 resources/comandos/comandos.py`: reescribe la sección de cada agente, el catálogo y, la primera vez, añade la entrada a su bitácora.
3. Recompila las guías (`python guides/_src/build.py`).
4. `python resources/comandos/comandos.py --verificar` falla si algo quedó desfasado.

Comandos de este agente

General

Comando	Qué hace aquí
`!analyze`	Analiza la solicitud, el problema, los documentos o la situación **antes de actuar**: qué se pide, qué hay, qué falta y qué riesgos ve. Entrega hallazgos; no ejecuta ni modifica nada. **Aquí:** Ingesta y análisis documental de los cuatro documentos (RFP, aclaraciones, modelo oficial, modelo de cotización); las aclaraciones prevalecen.
`!research`	Investiga a fondo el tema, cruza fuentes y entrega lo relevante con **cada dato citado a su fuente**. Sin fuente verificable no es un dato: se marca `[PENDIENTE]`.
`!plan`	Entrega un plan de trabajo estructurado (pasos, orden, insumos, entregables, supuestos y puntos de decisión) y **se detiene a esperar aprobación** antes de ejecutarlo. **Aquí:** Presenta cómo construirá los cuatro entregables y las dudas que bloquean; espera respuesta antes de construir.
`!execute`	Ejecuta ya lo solicitado o el siguiente paso aprobado del flujo, sin volver a proponer ni a preguntar lo ya decidido. No se salta las puertas de aprobación ni de verificación.
`!review`	Revisa el trabajo indicado y entrega oportunidades de mejora priorizadas (qué cambiaría y por qué). No lo modifica salvo que se le pida.
`!audit`	Auditoría integral de punta a punta: contradicciones entre documentos, cifras que no cuadran, reglas incumplidas y datos sin fuente. Entrega hallazgos con evidencia y severidad. **Aquí:** Verificación cruzada: propuesta, horas, costos y modelos conciliados entre sí y con el RFP.
`!validate`	Comprueba, requisito por requisito, que el resultado cumple lo establecido (reglas duras, insumos, formato). Veredicto «cumple / no cumple» por ítem, con evidencia. **Aquí:** Comprueba cobertura (ningún requerimiento sin fila), trazabilidad y que el plazo sea factible con el equipo propuesto.
`!optimize`	Optimiza lo que se esté desarrollando o ya existe (proceso, solución o resultado): mejora claridad, calidad, consistencia, peso o rendimiento **sin cambiar su alcance ni sus datos**.
`!compare`	Compara alternativas con criterios explícitos y recomienda la más conveniente, con el porqué.

Comando	Qué hace aquí
`!summarize`	Resumen claro y estructurado (lo esencial, decisiones y pendientes) del trabajo, de un documento o de la conversación.
`!document`	Documenta lo hecho: actualiza la wiki y la bitácora del agente, o genera la documentación técnica, funcional o ejecutiva que se pida.
`!prioritize`	Clasifica actividades, hallazgos o elementos por impacto, importancia y urgencia, y entrega el orden recomendado.
`!help`	Lista los comandos disponibles en este agente, agrupados, con su función.

Modos

Comando	Qué hace aquí
`!notime`	**Modo prisa:** apresura el desarrollo optimizándolo: agrupa preguntas en un solo mensaje, paraleliza, reutiliza lo ya validado, omite explicaciones y pasos opcionales y responde solo con lo esencial. **Nunca** se salta verificaciones obligatorias, puertas de aprobación ni la veracidad: acelera el camino, no recorta la calidad.
`!think-deep`	Análisis más profundo antes de responder: considera alternativas, casos límite y consecuencias, y muestra solo el razonamiento que importa.
`!critical-mode`	Modo crítico: cuestiona decisiones, supuestos y su propio trabajo; busca errores y contradicciones antes de entregar y los dice.
`!no-assumptions`	No supone: ante un dato faltante o ambiguo pregunta (todas las dudas juntas) en lugar de rellenar o inferir.
`!double-check`	Segunda verificación independiente de su propio resultado (releer la fuente, recalcular, comparar) y corrección de lo que encuentre; reporta qué verificó.
`!strict-mode`	Sigue al pie de la letra los requisitos e instrucciones dados: sin iniciativas, adiciones ni interpretaciones creativas.

Técnico y seguridad

Comando	Qué hace aquí
`!debug`	Identifica la causa de un error de software (script, conversión, verificación) y lo corrige; vuelve a correr la verificación para comprobarlo.
`!troubleshoot`	Investiga un problema técnico y determina su **causa raíz**, con evidencia. Diagnostica; no cambia nada.
`!security-check`	Revisa seguridad y confidencialidad: datos de otros clientes mezclados, credenciales, datos personales, información que no debe salir del proyecto. **Aquí:** Confidencialidad: cada proceso es confidencial; no mezcla información de una licitación con otra.
`!architecture`	Diseña o evalúa la arquitectura de una solución de software. **Aquí:** Diseña o evalúa la arquitectura de la solución de la propuesta técnica, con su justificación.

UX/UI y diseño

Comando	Qué hace aquí
`!visual-audit`	Examina proporciones, distribución, tipografía y consistencia visual del entregable, a la vista y página por página.
`!consistency-check`	Detecta inconsistencias entre secciones, componentes, cifras, nombres y estilos del mismo entregable.

Gestión de proyecto

Comando	Qué hace aquí
`!breakdown`	Divide un proyecto o solicitud compleja en actividades específicas, cada una con su entregable y su orden.
`!roadmap`	Construye una hoja de ruta de desarrollo con hitos y dependencias. **Aquí:** Cronograma de la propuesta, factible dentro del plazo exigido.
`!estimate`	Estima tiempos, recursos y complejidad, con el método y los supuestos escritos (un número sin método no se entrega). **Aquí:** Horas por módulo y especialidad, o tokens y costo de IA, con método y supuestos escritos.

Comando	Qué hace aquí
`!track-progress`	Informa el progreso y los pendientes (hecho, en curso, bloqueado, `[PENDIENTE]`) a partir de la bitácora y del trabajo actual.
`!risk-analysis`	Identifica riesgos, su probabilidad e impacto, y propone la mitigación de cada uno. **Aquí:** Riesgos del proceso y de la solución: plazos, ambigüedades, dependencias, descalificación.
`!requirements`	Extrae y estructura requerimientos funcionales y no funcionales de las fuentes, cada uno con su referencia de origen. **Aquí:** Construye la matriz de cumplimiento: cada requerimiento del RFP y de las aclaraciones con su respuesta y referencia.
`!generate-report`	Genera un informe ejecutivo (decisiones, resultado, pendientes) o técnico (método, evidencias, verificaciones) del trabajo.
`!final-delivery`	Prepara y valida la entrega final: corre todas las verificaciones obligatorias, completa el checklist de cierre (wiki y bitácora) y entrega solo si todo pasa. **Aquí:** Matriz completa, los cuatro entregables y los dos modelos conciliados, con la verificación cruzada hecha.
`!agent-handoff`	Prepara el traspaso al siguiente agente del flujo comercial: resumen, entregables con su ruta, decisiones tomadas y pendientes. **Aquí:** Si se adjudica, prepara el traspaso a **Perfeccionador de Contratos** (`contract-perfectionist`) con la propuesta y el modelo de costos radicado.

IA avanzada

Comando	Qué hace aquí
`!second-opinion`	Un subagente independiente evalúa el resultado **sin ver tu razonamiento**; se contrasta con el tuyo y se explican las diferencias.
`!cross-review`	Otro especialista (subagente con otro rol) revisa el resultado y devuelve observaciones; se aplican las válidas y se informa de las descartadas.

Multiagente

Comando	Qué hace aquí
`!orchestrate`	Reparte partes **independientes** del trabajo entre varios subagentes en paralelo y las integra en un solo resultado verificado. **Aquí:** Ingesta de los documentos y construcción de entregables independientes en paralelo; la conciliación y la verificación cruzada las hace el agente principal.
`!team-review`	Varios revisores con roles distintos evalúan por separado el mismo resultado y se consolida su evaluación. **Aquí:** Revisores con rol **técnico (arquitectura y horas)**, **costos** y **cumplimiento frente al RFP**.
`!consensus`	Consolida las evaluaciones disponibles (de subagentes o de revisiones previas) en una sola recomendación, con las discrepancias a la vista.

Mantenimiento

La bitácora `wiki/log.md` es append-only; cada entrada empieza con `## [YYYY-MM-DD] <tipo> | <descripción>` y el tipo es `setup`, `ingesta`, `build`, `ajuste`, `verificacion` o `lint`. El lint periódico revisa contradicciones entre páginas, precios de IA desactualizados, páginas huérfanas, licitaciones sin ficha y convenciones en desuso.

Quiero...	Qué tocar
Actualizar los precios de IA	`wiki/costeo-ia-tokens.md` (con fecha) y una entrada en `wiki/log.md`
Cambiar el método de estimación	`wiki/estimacion-horas.md` y la skill `estimacion-horas`
Cambiar la estructura de la propuesta	`wiki/estructura-propuesta-tecnica.md`
Agregar un bloque al checklist técnico	`wiki/consideraciones-tecnicas.md`
Cambiar un procedimiento	El `SKILL.md` de la habilidad correspondiente
Registrar una licitación nueva	Carpeta en `licitaciones/`, ficha en `wiki/licitaciones/`, `wiki/index.md` y `wiki/log.md`

Problemas frecuentes

Síntoma	Causa probable	Qué hacer
Falta uno de los cuatro documentos	El proceso no lo incluye o no se depositó	Preguntar; si no existe, dejarlo registrado en el análisis y en el log
RFP y aclaraciones se contradicen	Una aclaración modificó el alcance	Registrar ambas citas; si cambia alcance o precio, preguntar antes de resolver
El plazo no cabe con el equipo	Horas totales demasiado altas	Reportar con los números y las opciones; no ajustar la estimación
La diferencia entre `06` y `07` no se explica	Mapeo incorrecto o un costo no contemplado	Revisar el mapeo de celdas; si persiste, reportar
Fórmulas rotas tras insertar filas	No se propagaron las fórmulas	Reparar las filas nuevas y recalcular
El costo de IA parece alto	Se ignoró caché o *batch*, o se usó un modelo grande para una tarea simple	Revisar la elección de modelo y aplicar los mecanismos de ahorro
Precios de IA desactualizados	La tabla de la wiki es antigua	Actualizar antes de costear y anotarlo

Referencia rápida de archivos

Archivo	Para qué lo abres
`CLAUDE.md`	Rol, jerarquía documental, flujo y reglas duras
`habilidades/*/SKILL.md`	El procedimiento detallado de cada paso
`wiki/flujo-trabajo.md`	Las tres fases del proceso
`wiki/estimacion-horas.md`	Método, especialidades y factibilidad

Archivo	Para qué lo abres
`wiki/costeo-ia-tokens.md`	Precios, señales y método de tokens
`wiki/consideraciones-tecnicas.md`	Checklist de 15 bloques
`wiki/plantilla-verificacion-cruzada.md`	Plantilla de la verificación cruzada
`wiki/licitaciones/<slug>.md`	Ficha de cada licitación: decisiones y mapeo de modelos