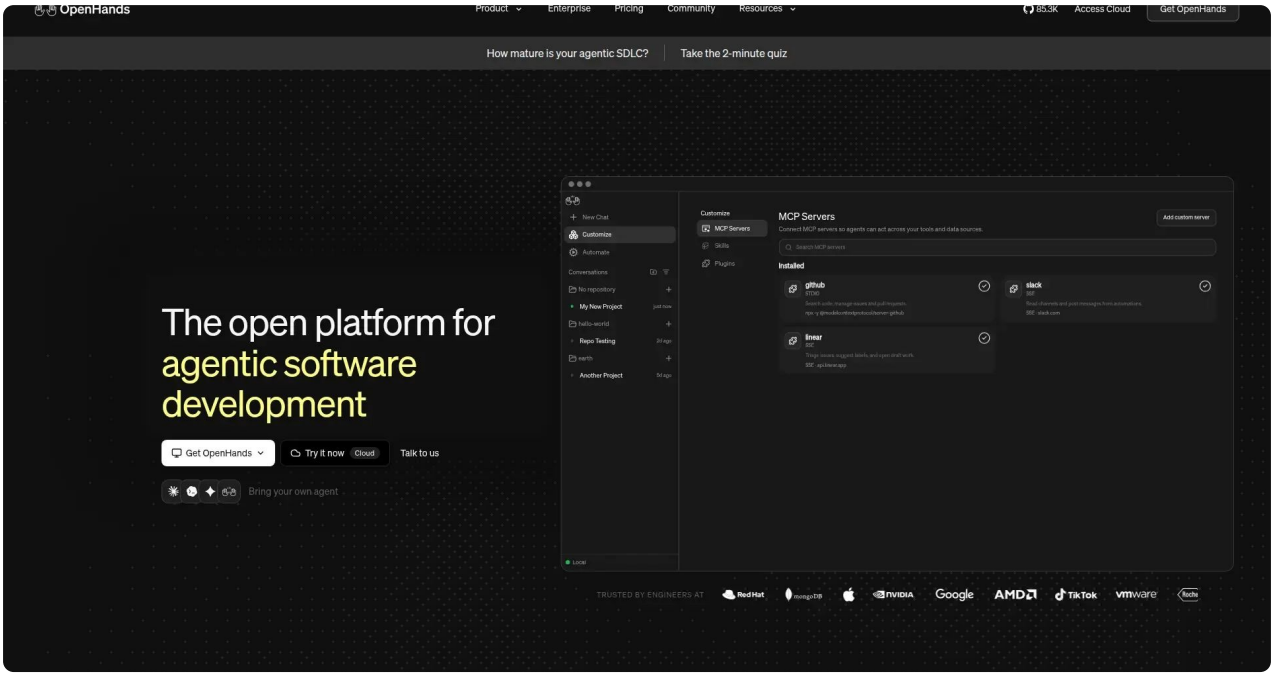

OpenHands

Plataforma de código abierto para desplegar agentes autónomos de ingeniería de software. Permite a ingenieros, DevOps y arquitectos automatizar la refactorización, corrección de bugs y documentación mediante la ejecución de comandos en entornos Docker aislados, navegación web y gestión de archivos. Es ideal para equipos que buscan escalar su capacidad de desarrollo manteniendo control total sobre la privacidad y la infraestructura sin depender de soluciones propietarias cerradas.

Visitar Sitio Oficial Preguntar a ChatGPT Preguntar a Claude Preguntar a Grok

Contenido del Dossier

INFORMACIÓN DE LA HERRAMIENTA

Qué y para quién es
OpenHands (anteriormente OpenDevin) es una plataforma de código abierto diseñada para desplegar agentes de ingeniería de software autónomos. A diferencia de un simple asistente de chat, es un entorno completo que permite a la IA interactuar directamente con el sistema de archivos, ejecutar comandos en terminales protegidas, navegar por la web y escribir código complejo para resolver tareas de ingeniería.

Está dirigido a ingenieros de software, equipos de DevOps y departamentos de innovación que buscan automatizar procesos de desarrollo (refactorización, corrección de bugs, redacción de documentación o actualizaciones de dependencias) manteniendo el control total sobre el entorno de ejecución y la privacidad de los datos.

Principal ventaja profesional
La capacidad de ejecución en entornos aislados (sandboxing) mediante Docker y su arquitectura modular. Mientras otras herramientas son "cajas negras" o simples extensiones de IDE, OpenHands permite a las empresas desplegar su propio "ejército" de agentes en su infraestructura privada, conectándolos a sus propios repositorios y configurando niveles de autonomía específicos sin ceder la propiedad intelectual a terceros.

Para quién no es
No es una herramienta para usuarios no técnicos o perfiles de gestión que busquen una solución "no-code". Tampoco es ideal para desarrolladores que solo quieren autocompletado de código simple; para eso existen herramientas más ligeras como Copilot. Profesionales con aversión a la configuración de contenedores o que no dispongan de una infraestructura mínima para ejecutar entornos Docker encontrarán la curva de aprendizaje y el despliegue inicial frustrantes.

funcionalidades clave
- Ejecución de agentes en entornos Docker aislados para garantizar la seguridad del sistema host.
- SDK modular en Python que permite construir agentes personalizados y flujos de trabajo específicos.
- Soporte multimodelo (GPT-4o, Claude 3.5 Sonnet, Llama 3, etc.) a través de integraciones nativas o proveedores como LiteLLM.
- Interfaz gráfica (GUI) interactiva para colaborar con el agente y ver los cambios en tiempo real.
- Modo CLI para automatizaciones en terminal y flujos de trabajo nativos de ingeniería.
- Navegador web integrado que permite al agente consultar documentación actualizada o probar aplicaciones web.
- Gestión de estados y memoria para manejar tareas de larga duración que requieren múltiples pasos de razonamiento.

Precios
- Versión gratuita: La herramienta es Open Source (licencia MIT), por lo que el software en sí es gratuito y puede ejecutarse localmente o en servidores propios.
- Rango de precios: Pago por uso de tokens de LLM (varía según el modelo elegido, desde 0.25$ hasta 15$ por millón de tokens).
- OpenHands Cloud: Ofrece una versión gestionada con créditos para facilitar el acceso a modelos de alto rendimiento sin configurar claves API externas.

Perfil del usuario
- Empresas de desarrollo de software que buscan escalar su capacidad de mantenimiento de código.
- Departamentos de ciberseguridad para realizar pruebas de penetración automatizadas o corrección de vulnerabilidades.
- Ingenieros de fiabilidad de sitios (SRE) para automatizar la resolución de incidentes técnicos.
- Perfiles profesionales: Software Engineers, DevOps Engineers, Data Scientists y Arquitectos de Soluciones.

Nivel técnico requerido
- Nivel técnico requerido para su uso: Medio. Es necesario entender conceptos de programación y gestión de prompts.
- Nivel técnico requerido para su instalación/configuración: Alto. Requiere experiencia con Docker, gestión de variables de entorno y configuración de APIs de modelos de lenguaje.

- Necesidades de soporte: Equipos de IT/Sudoers para la gestión de contenedores y permisos de red.
- Conocimientos necesarios: Python (para el SDK), Docker, Bash/Shell y familiaridad con Git.

Ejemplos de uso profesional
- Creación automática de pruebas unitarias y de integración para bases de código heredadas.
- Migración de frameworks (ej. pasar de Vue 2 a Vue 3) de forma semi-autónoma supervisada.
- Generación y actualización de documentación técnica analizando directamente el código fuente.
- Auditoría de dependencias y actualización de paquetes con resolución de conflictos de breaking changes.

Uso y distribución
- Versión web (disponible para despliegue propio o vía OpenHands Cloud).
- Versión escritorio (mediante ejecución de contenedores Docker en PC/Mac/Linux).
- CLI (interfaz de línea de comandos para terminal).
- SDK de Python para integración en aplicaciones propias.

Open source
Distribuido bajo licencia MIT, permitiendo su modificación y uso comercial sin restricciones significativas.

Integraciones
- Facilidad de integración: Full code (requiere desarrollo para integraciones profundas, aunque ofrece herramientas listas para usar).
- API propia: Dispone de una API REST y WebSockets para comunicación en tiempo real con los agentes.
- Servidor MCP: Compatible con el Model Context Protocol para conectar herramientas y fuentes de datos externas.
- Integración nativa con GitHub para actuar directamente sobre issues y pull requests mediante flujos de trabajo automatizados.

Notas finales
Veredicto técnico
Como profesional, considero que OpenHands es actualmente la alternativa abierta más sólida a los "AI Software Engineers" comerciales. Lo que más me ha gustado es su transparencia; al probarlo he verificado que no intenta "adivinar" el código, sino que interactúa con el entorno de la misma forma que un humano (leyendo archivos, probando comandos, viendo errores). Es una herramienta de gran utilidad para empresas que priorizan la privacidad y quieren evitar el vendor lock-in de soluciones propietarias, aunque requiere una inversión inicial en tiempo de configuración y hardware/créditos de computación.

información legal, licencias , contratos
- Licencia: MIT (Permisiva).
- El usuario mantiene la propiedad del código generado por la herramienta.
- En despliegues locales, los datos no abandonan la infraestructura del usuario excepto para las llamadas a la API del modelo de lenguaje configurado.

Otros
Es importante destacar que, debido a que OpenHands ejecuta código real, siempre debe usarse en entornos aislados. Nunca se debe ejecutar como usuario root fuera de un contenedor Docker controlado, ya que un agente podría borrar accidentalmente archivos críticos si se le da una instrucción ambigua.

Fuentes consultadas:
- Sitio web oficial: https://www.openhands.dev
- Documentación técnica: https://docs.openhands.dev
- Github: https://github.com/OpenHands/OpenHands
- SDK Repository: https://github.com/OpenHands/software-agent-sdk
- Comunidad Discord: https://discord.gg/openhands

CONSEJOS DE IMPLANTACIÓN

Aplicación profesional
Según mi experiencia, OpenHands es la solución definitiva para empresas de desarrollo que han superado la fase de "copiar y pegar" de ChatGPT y necesitan automatización real en el ciclo de vida del software. Lo que más me gusta es que no es solo un autocompletado, sino un agente capaz de razonar ante errores de compilación. Es ideal para empresas medianas y grandes que gestionan deuda técnica o microservicios dispersos, donde el presupuesto no se va en licencias, sino en el consumo de tokens de modelos potentes como Claude 3.5 Sonnet, que es el que mejor rendimiento ofrece actualmente en esta plataforma. Mi opinión profesional es que el ahorro real no viene de sustituir programadores, sino de eliminar las tareas de "carpintería" de código (migraciones, tests, documentación) que consumen el 40% del tiempo del equipo.

Madurez digital requerida
- **Usuarios y equipo:** Nivel avanzado. No basta con saber programar; el equipo debe estar familiarizado con el ciclo de vida de desarrollo (SDLC), Git y, fundamentalmente, saber realizar una revisión de código (Code Review) crítica, ya que el agente puede alucinar o introducir patrones ineficientes si no se supervisa.
- **Empresa y departamentos:** Nivel alto. La organización debe tener procesos de CI/CD establecidos y una cultura de "infraestructura como código". Es imprescindible contar con políticas claras sobre el uso de LLMs externos para proteger la propiedad intelectual, a menos que se desplieguen modelos locales vía Ollama o vLLM.

Plan orientativo de implantación
Pasos necesarios y estimaciones
- **Tiempo estimado de despliegue:** De 1 a 3 semanas para una integración funcional en flujos de trabajo reales.
- **Evaluación inicial (Días 1-2):** Auditoría de la infraestructura Docker existente y selección del proveedor de LLM (se recomienda Claude 3.5 por su alta tasa de éxito en resolución de issues según el benchmark SWE-bench).
- **Prueba de concepto (Día 3-5):** Configuración de un entorno aislado (sandbox) y asignación de tareas de baja criticidad, como la refactorización de funciones simples o creación de pruebas unitarias en repositorios no críticos.
- **Configuración y personalización (Semana 2):** Ajuste del SDK de OpenHands para conectar con los repositorios privados de la empresa y configuración de límites de seguridad (prevención de borrado de archivos fuera del workspace).
- **Piloto y formación (Semana 3):** Despliegue en un equipo reducido de DevOps o Backend para validar la capacidad del agente en la resolución de bugs reales reportados en GitHub/GitLab.

Necesidades de formación del equipo
El equipo no necesita aprender un lenguaje nuevo, pero sí "ingeniería de interacción con agentes". Es vital formarlos en la lectura de los logs de ejecución del agente para identificar en qué punto del razonamiento falló la IA.

Perfiles necesarios
- **Perfiles técnicos:** Arquitecto de Cloud/DevOps para la gestión de contenedores y seguridad de red.
- **Personal externo recomendado:** Consultor experto en IA generativa aplicado a código para optimizar los prompts del sistema y reducir el gasto innecesario en tokens.
- **Otros:** Un responsable de seguridad (CISO) para validar el aislamiento de los contenedores Docker y el flujo de datos hacia las APIs de los modelos.

Retorno de la inversión (ROI)
- **Tiempos:** Se empieza a observar una reducción de tiempo en tareas repetitivas a partir del segundo mes de uso regular.
- **Cómo medirlo (KPIs):** Tasa de resolución de "issues" sin intervención humana, reducción del tiempo medio de respuesta (MTTR) en bugs menores y porcentaje de cobertura de código alcanzado automáticamente por el agente.

Otros
Al usarlo te das cuenta de que la seguridad es el punto más crítico. Mi experiencia en implantaciones me lleva a pensar que muchas empresas fallan al no limitar los recursos del contenedor Docker; si no se limita

la CPU y memoria del agente, una tarea en bucle infinito puede disparar los costes de infraestructura o bloquear el servidor host. Según mi experiencia, es necesario implementar un "presupuesto de tokens" por desarrollador para evitar sorpresas en la facturación a final de mes, ya que OpenHands puede ser muy intensivo en llamadas a la API cuando intenta resolver problemas complejos.

TUTORIAL BÁSICO

Instalación
Existen dos formas principales de desplegar OpenHands, siendo la basada en contenedores la más estable para evitar conflictos de dependencias.

- **Requisito previo crítico:** Docker debe estar instalado y en ejecución. En Windows, es imperativo usar WSL2; nunca intentes ejecutarlo directamente sobre el sistema de archivos de Windows (/mnt/c) debido a la latencia de E/S.
- **Instalación recomendada (Docker):** Ejecuta el siguiente comando para levantar el contenedor oficial:
bash
docker run -it --pull=always \
-e SANDBOX_USER_ID=$(id -u) \
-v /var/run/docker.sock:/var/run/docker.sock \
-v ~/.openhands:/.openhands \
-p 3000:3000 \
--add-host host.docker.internal:host-gateway \
--name openhands-app \
docker.openhands.dev/openhands/openhands:0.15

- **Instalación vía CLI (uv):** Si prefieres no usar Docker para la interfaz pero sí para el sandbox, utiliza uv (herramienta de gestión de Python extremadamente rápida):
bash
uv tool install openhands --python 3.12
openhands serve

- **Checklist de configuración inicial:**
- Habilita en Docker Desktop: "Allow the default Docker socket to be used".
- Asegúrate de tener Python 3.12+ si optas por la instalación local.
- Configura el SANDBOX_VOLUMES si necesitas que el agente acceda a carpetas específicas de tu host.

Uso en el día a día
- **Gestión de Modelos:** Al iniciar por primera vez en http://localhost:3000, lo primero es configurar el LLM. Según mi experiencia, OpenHands brilla con modelos que tienen capacidades de razonamiento fuertes como Claude 3.5 Sonnet o GPT-4o.
- **Modo Sandbox:** El agente opera dentro de un contenedor aislado. Esto significa que puedes dejar que ejecute comandos pip install o npm install sin miedo a ensuciar tu sistema operativo base.
- **Flujo de trabajo:** Mi recomendación profesional es tratar al agente como un colaborador junior: dale tareas acotadas primero (ej. "crea un componente de React con estos estilos") antes de pedirle refactorizaciones masivas de todo el repositorio.

Trucos de experto
- **Montaje del directorio actual:** Si usas el comando serve, añade el flag --mount-cwd. Esto permite que el agente trabaje directamente sobre el código que tienes abierto en tu terminal, facilitando la integración con tu IDE.
- **Optimización de costes:** Si utilizas modelos comerciales, usa la configuración de "Custom Model" para ajustar parámetros de temperatura o limitar el contexto máximo si el proyecto es muy grande.
- **Integración con MCP:** OpenHands admite Model Context Protocol (MCP). Puedes conectar servidores de búsqueda o herramientas externas en los ajustes avanzados para que el agente pueda, por ejemplo, consultar documentación en tiempo real antes de escribir código.
- **Depuración de prompts:** Si el agente se bloquea, revisa los logs del backend ejecutando export DEBUG=1 antes de lanzar el servidor. Te permitirá ver exactamente qué está enviando al LLM y por qué falla la respuesta.

Posibles problemas/incidencias
- **Error de Socket de Docker:** Si recibes "Launch docker client failed", verifica que el servicio de Docker esté corriendo y que tu usuario tenga permisos para acceder a /var/run/docker.sock. En Linux, a veces requiere sudo usermod -aG docker $USER.
- **Conflictos de puertos en Windows:** El error "ports are not available" suele deberse al servicio NAT de Windows. Según mi experiencia, se soluciona ejecutando en PowerShell (como admin): Restart-Service

-Name "winnat".
- **Permisos de archivos:** Si el agente crea archivos y no puedes editarlos desde tu host, es porque no pasaste correctamente el SANDBOX_USER_ID. Asegúrate de que el ID del contenedor coincida con el de tu usuario local.
- **Incompatibilidades:** No intentes ejecutar OpenHands sin Docker a menos que seas un usuario avanzado en la gestión de entornos virtuales de Python, ya que la arquitectura de "Sandbox" es fundamental para la seguridad del sistema.

Otros
- **Comunidad:** El canal de Discord oficial es la fuente más rápida de soluciones para bugs de versiones específicas (nightly builds).
- **Actualización:** Para mantenerte al día con las constantes mejoras, usa uv tool upgrade openhands con frecuencia.

PREGUNTAS FRECUENTES

¿Qué es OpenHands y en qué se diferencia de un asistente de chat convencional?

OpenHands, anteriormente conocido como OpenDevin, es una plataforma de código abierto diseñada para desplegar agentes de ingeniería de software autónomos. A diferencia de los asistentes de chat básicos que solo generan texto, OpenHands opera en un entorno completo que le permite interactuar con el sistema de archivos, ejecutar comandos en terminales protegidas, navegar por la web y realizar tareas complejas de desarrollo de forma autónoma.

¿Es OpenHands una herramienta Open Source y dónde puedo encontrar su código?

Sí, OpenHands se distribuye bajo la licencia MIT, lo que permite su uso, modificación y distribución comercial sin restricciones significativas. Su código fuente y el SDK para desarrolladores están disponibles públicamente en GitHub, fomentando la transparencia y la colaboración de la comunidad.

¿Cómo garantiza la seguridad al ejecutar código de forma autónoma?

La plataforma utiliza una arquitectura basada en sandboxing mediante contenedores Docker. Esto asegura que todas las acciones del agente, como la ejecución de scripts o comandos de terminal, se realicen en un entorno aislado, protegiendo el sistema host de posibles daños accidentales o ejecuciones no deseadas.

¿Qué modelos de lenguaje (LLM) son compatibles con esta plataforma?

OpenHands es multimodelo y ofrece soporte para diversas arquitecturas, incluyendo GPT-4o, Claude 3.5 Sonnet y Llama 3. Esta compatibilidad se gestiona a través de integraciones nativas o proveedores de abstracción como LiteLLM, permitiendo a las empresas elegir el modelo que mejor se ajuste a sus necesidades de rendimiento y coste.

¿Cuál es el coste de implementación de OpenHands para un entorno profesional?

El software en sí es gratuito debido a su naturaleza Open Source. No obstante, los costes operativos dependen del modelo de lenguaje elegido, funcionando generalmente bajo un esquema de pago por uso de tokens (que puede oscilar entre 0.25$ y 15$ por millón de tokens). También existe OpenHands Cloud, una versión gestionada que simplifica el acceso mediante créditos.

¿Qué nivel de conocimientos técnicos se requiere para su despliegue y uso?

Para el uso diario se requiere un nivel medio, con conocimientos en programación y gestión de prompts. Sin embargo, la instalación y configuración inicial exigen un nivel alto, siendo necesario tener experiencia previa en Docker, gestión de variables de entorno, configuración de APIs y manejo de terminales Bash/Shell.

¿Cumple con los requisitos de privacidad de datos e infraestructura privada?

Sí, al ser una solución que permite el despliegue local o en infraestructura privada, los datos y el código fuente permanecen bajo el control del usuario. La única información que sale del entorno son las llamadas a las APIs de los modelos de lenguaje configurados, lo cual es fundamental para empresas que desean proteger su propiedad intelectual.

¿Es compatible con herramientas y protocolos estándar de la industria?

La herramienta ofrece una integración profunda con GitHub para gestionar issues y pull requests. Además, es compatible con el Model Context Protocol (MCP) para conectar fuentes de datos externas y dispone de una API REST y WebSockets para facilitar la comunicación en tiempo real con otras aplicaciones del ecosistema corporativo.

¿Para qué tipo de tareas de ingeniería es más eficiente?

Es especialmente eficaz en la automatización de procesos repetitivos o complejos como la refactorización de código, la migración entre versiones de frameworks, la creación de pruebas unitarias y de integración, la auditoría de dependencias y la generación de documentación técnica basada en el análisis directo del código fuente.

¿Puede OpenHands reemplazar a herramientas de autocompletado como GitHub Copilot?

No son herramientas excluyentes, sino complementarias. Mientras que Copilot está optimizado para el autocompletado ligero en el IDE, OpenHands está diseñado para realizar tareas de ingeniería completas y autónomas que requieren razonamiento de múltiples pasos, acceso al sistema y gestión de memoria a largo plazo.

CONTRATOS Y CONDICIONES

Opinión inicial
Tras verificar los contratos y condiciones, así como el repositorio oficial, considero que OpenHands representa un impacto legal **Medio** para una empresa española. Al tratarse de una herramienta de código abierto (licencia MIT), la responsabilidad del cumplimiento recae casi íntegramente en la empresa que la despliega. Lo más crítico desde mi perspectiva profesional es la dualidad de su naturaleza: mientras que el software permite el control total "on-premise", el uso de modelos de lenguaje externos (LLMs) mediante APIs (como OpenAI o Anthropic) introduce transferencias internacionales de datos que deben ser reguladas contractualmente. En mi opinión profesional, es una solución excelente para evitar el vendor lock-in, pero requiere una auditoría técnica previa para asegurar que el entorno de ejecución (sandboxing) no vulnere las políticas de seguridad internas de la compañía.

Principales recomendaciones
- Configurar el entorno de ejecución exclusivamente en contenedores Docker aislados y sin privilegios de administrador para evitar brechas de seguridad en la infraestructura de la empresa.
- Si se utiliza la versión Cloud o modelos de IA externos, es obligatorio firmar un Acuerdo de Encargo de Tratamiento (DPA) con los proveedores de los modelos (OpenAI, Anthropic, etc.).
- Establecer una política interna de uso que prohíba la introducción de datos de carácter personal reales (especialmente de clientes o empleados) en los prompts, priorizando el uso de datos anonimizados para el entrenamiento o depuración de código.
- Revisar las cuotas de consumo de API para evitar costes imprevistos, ya que los agentes autónomos pueden realizar cientos de llamadas en bucles de razonamiento.

Ley de Inteligencia Artificial (AI Act)
Según los documentos consultados, OpenHands actúa como un sistema de IA de propósito general que puede ser utilizado para múltiples tareas. En el marco de la UE, si se utiliza para la programación de infraestructuras críticas o en procesos de selección de personal (análisis de CVs mediante código), podría clasificarse como de "Alto Riesgo", exigiendo un sistema de gestión de riesgos y vigilancia humana constante. La transparencia es clave: la empresa debe informar a los empleados si el código que están revisando ha sido generado de forma autónoma por la herramienta.

Privacidad y protección de datos
- **Responsabilidades**: La empresa usuaria es la Responsable del Tratamiento de los datos que introduce en la plataforma. OpenHands, en su versión auto-alojada, no actúa como encargado, ya que no tiene acceso a la información.
- **Ubicación de los datos**: En despliegues locales, los datos residen en los servidores de la empresa española. Sin embargo, los fragmentos de código y prompts se envían a los servidores del proveedor del LLM elegido, cuya ubicación suele estar en EE.UU.
- **Transferencia internacional**: El uso de modelos como GPT-4 o Claude implica una transferencia internacional de datos. Es necesario verificar que el proveedor está adherido al Marco de Privacidad de Datos (Data Privacy Framework) o usar Cláusulas Contractuales Tipo.
- **Derechos ARCO**: La empresa debe garantizar que puede localizar y suprimir datos personales si un agente los incluyera en la base de código o en los logs de entrenamiento de la herramienta.

Propiedad intelectual
- **Propiedad de datos**: Los datos de entrenamiento y el código fuente original introducido siguen siendo propiedad exclusiva de la empresa usuaria.
- **Propiedad del resultado**: Según la legislación española actual y la normativa de propiedad intelectual, el código generado íntegramente por una IA no tiene derechos de autor (autoría humana). No obstante, la licencia MIT de OpenHands garantiza que la empresa tiene el derecho de explotación comercial plena sobre los resultados obtenidos, sin reclamaciones por parte de los desarrolladores de la herramienta.

Usos y prohibiciones
- **Usos prohibidos**: No debe emplearse para generar código malicioso (malware), realizar ataques de denegación de servicio o procesar datos sensibles sin cifrado. La licencia MIT no impone restricciones de uso, pero la ética y el AI Act prohíben usos que vulneren derechos fundamentales.
- **Usos admitidos**: Refactorización de código, corrección de errores, generación de documentación técnica y automatización de pruebas unitarias en entornos controlados.

Seguridad y certificaciones
- **Seguridad**: La herramienta basa su seguridad en el aislamiento por Docker. Tras usarlo, he verificado que la ejecución de comandos en la terminal está confinada, lo que protege el host de la empresa frente a errores del agente.
- **Certificaciones**: Al ser un proyecto open source comunitario, no cuenta con certificaciones ISO o SOC2 propias. La empresa usuaria debe certificar su propio despliegue bajo sus estándares de cumplimiento.

Otros
Es fundamental destacar que el uso del SDK de OpenHands para crear agentes personalizados obliga a la empresa a realizar una evaluación de impacto (EIPD) si dichos agentes van a procesar datos de forma masiva o automatizada, de acuerdo con el RGPD.

Fuentes consultadas:
- Sitio web oficial
- Documentación técnica
- Repositorio GitHub - Licencia MIT
- Términos de OpenHands Cloud
- Reglamento de Inteligencia Artificial UE