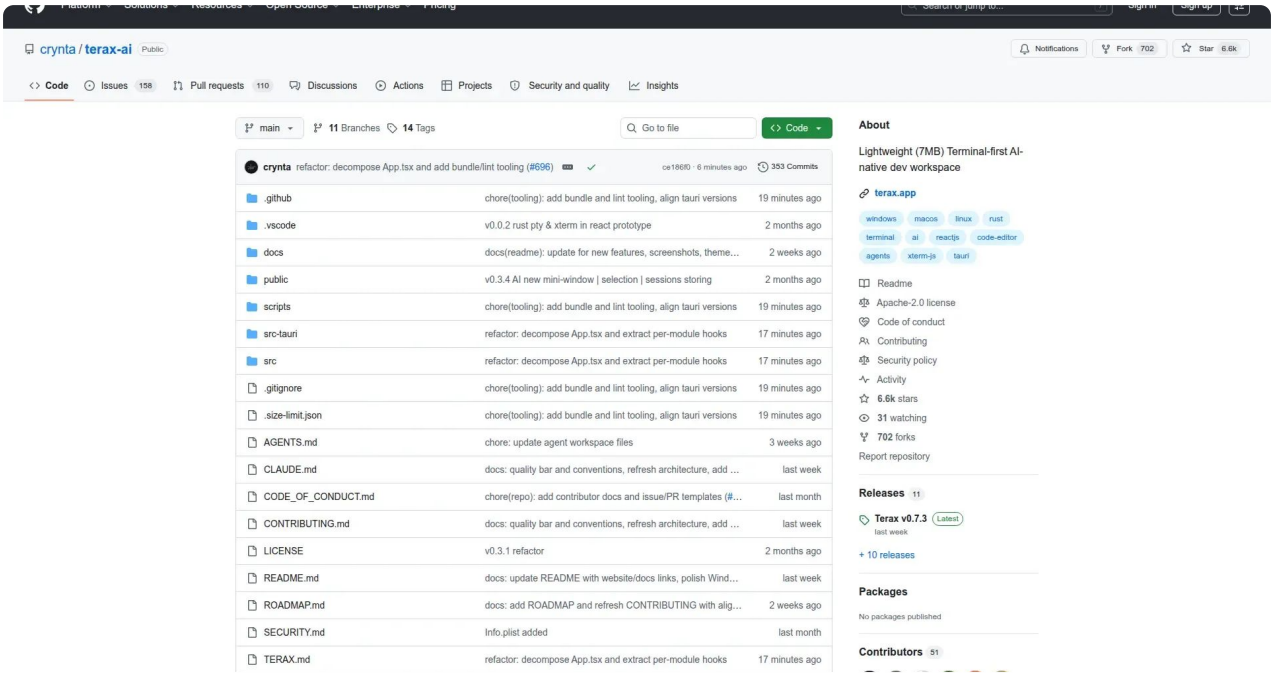

Terax AI-native Dev Workspace

Terax es un entorno de desarrollo ultraligero que integra terminal, editor de código y agentes de IA en una interfaz de alto rendimiento. Diseñado para ingenieros de software, desarrolladores backend y perfiles DevOps, permite automatizar tareas de sistema, gestionar Git y refactorizar código mediante modelos locales o externos. Su arquitectura sin telemetría garantiza la privacidad total del código fuente, siendo ideal para profesionales que buscan eficiencia terminal-first y soberanía de datos.

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

Contenido del Dossier

INFORMACIÓN DE LA HERRAMIENTA

Qué y para quién es
Terax es un entorno de desarrollo (ADE - AI-native Dev Workspace) ultraligero que fusiona terminal, editor de código y agentes de IA en una única interfaz de alto rendimiento. Está diseñado específicamente para desarrolladores y perfiles de ingeniería de software que buscan una herramienta orientada a la eficiencia ("terminal-first") sin el consumo de recursos de IDEs tradicionales. En el ámbito profesional, encaja en equipos que ya utilizan metodologías ágiles y requieren una integración profunda de IA tanto en la escritura de código como en la ejecución de tareas de sistema (operaciones de archivos, gestión de git, etc.).

Principal ventaja profesional
En mi opinión profesional, la razón definitiva para adoptar Terax es su arquitectura de "zero telemetry" combinada con un peso ridículo (menos de 10 MB). A diferencia de otras soluciones que obligan a pasar por sus servidores o cuentas de usuario, Terax permite usar tus propias claves (BYOK) o modelos 100% locales (Ollama/LM Studio), garantizando que el código de la empresa nunca sale del entorno de desarrollo si así se decide, manteniendo una latencia de respuesta instantánea.

Para quién no es
Tras probar la herramienta, considero que no es adecuada para desarrolladores que dependen fuertemente de extensiones complejas de VS Code o configuraciones de IDE muy pesadas. Profesionales que no se sientan cómodos con la terminal o que prefieran interfaces puramente visuales sin manipulación de comandos encontrarán la curva de aprendizaje frustrante. Tampoco es para empresas con políticas estrictas contra el software Open Source no firmado (en Windows requiere saltarse el aviso de SmartScreen por ahora).

funcionalidades clave
- **Agentes de IA con ejecución de herramientas**: Los agentes pueden leer archivos, ejecutar comandos de terminal y proponer cambios que se revisan mediante un sistema de "diffs" antes de aplicarse.
- **Terminal de alto rendimiento**: Utiliza xterm.js con renderizado WebGL, lo que garantiza fluidez total incluso con grandes volúmenes de logs.
- **Preview web integrado**: Detecta automáticamente servidores de desarrollo locales (Vite, Next.js, etc.) y abre una pestaña de previsualización en el mismo panel.
- **Gestión Nativa de Git**: Incluye un panel de control de fuentes con un grafo de commits real y visualización de ramas/merges.
- **Soporte de Modelos Locales**: Integración nativa con Ollama, MLX y LM Studio para desarrollo sin conexión a internet y máxima privacidad.
- **Editor CodeMirror 6**: Con modo Vim integrado, autocompletado por IA y soporte para múltiples lenguajes de programación.

Precios
- **Versión gratuita**: Es un proyecto Open Source bajo licencia Apache 2.0. No tiene coste de adquisición, no requiere cuenta y no incluye publicidad.
- **Modelo de costes**: El usuario asume el coste del consumo de API de los proveedores (OpenAI, Anthropic, Google, etc.) mediante sus propias claves (BYOK), o coste cero si se utilizan modelos locales.

Perfil del usuario
- Empresas de desarrollo de software que priorizan la privacidad del código (IP).
- Departamentos de DevOps que necesitan un flujo rápido entre terminal y herramientas de IA.
- Desarrolladores full-stack que trabajan con múltiples microservicios y servidores locales.
- **Perfiles clave**: Ingenieros de Software, SRE (Site Reliability Engineers), Desarrolladores Backend y entusiastas de herramientas CLI.

Nivel técnico requerido
- **Uso**: Medio. Requiere familiaridad con la terminal y conceptos de prompts de IA.
- **Instalación/Configuración**: Medio. Es necesario configurar las claves API o los endpoints locales (Ollama/LM Studio).
- **Conocimientos necesarios**: Manejo de CLI (bash/zsh/pwsh), Git básico y comprensión de modelos LLM.

Ejemplos de uso profesional

- **Refactorización guiada**: Pedir al agente que analice un directorio completo para detectar deudas técnicas y aplicar parches de seguridad de forma masiva.
- **Depuración de logs en tiempo real**: Usar la IA para analizar la salida de un servidor en la terminal y proponer correcciones inmediatas en el código.
- **Prototipado rápido**: Generar estructuras de archivos y boilerplate de aplicaciones mediante comandos de voz o texto, visualizando los resultados al instante en la pestaña de preview.

Uso y distribución
- **Versión escritorio**: Disponible para macOS (Apple Silicon e Intel), Windows (x64) y Linux (AppImage, .deb, .rpm y AUR).

Open source
Proyecto totalmente abierto bajo licencia Apache-2.0, con el código fuente disponible en GitHub para auditoría y contribuciones.

Integraciones
- **Facilidad de integración**: Alta para desarrolladores.
- **Proveedores de IA**: OpenAI, Anthropic, Google (Gemini), Groq, xAI (Grok), Cerebras, DeepSeek, Mistral y cualquier endpoint compatible con la API de OpenAI.
- **Herramientas locales**: Conecta de forma nativa con Ollama, LM Studio y MLX.
- **Sistemas de archivos**: Integración profunda con el sistema de archivos local y entornos WSL en Windows.

Notas finales
Veredicto técnico
Como profesional, valoro Terax como una herramienta de gran utilidad y eficiencia. Es la alternativa perfecta para aquellos que sienten que Cursor o VS Code se han vuelto demasiado pesados o intrusivos. Su enfoque en la ligereza y la soberanía del dato (al no requerir login ni telemetría) la posiciona como una opción muy seria para entornos corporativos donde la seguridad es crítica.

información legal, licencias , contratos
- **Licencia**: Apache License 2.0. Permite uso comercial, modificación y distribución.
- **Privacidad**: No hay recolección de datos (telemetría). Las claves de API se almacenan de forma segura en el llavero del sistema operativo (keyring).

Otros
- El archivo TERAX.md funciona como memoria de proyecto para los agentes, permitiendo que la IA entienda el contexto acumulado del repositorio.
- Tiempo de arranque inferior a 300 ms, eliminando la latencia de carga típica de aplicaciones Electron.

Fuentes consultadas:
- Sitio web oficial
- Documentación técnica
- Repositorio oficial en Github
- Historial de versiones y cambios

CONSEJOS DE IMPLANTACIÓN

Aplicación profesional
Según mi experiencia, Terax es la solución ideal para empresas de desarrollo de software que manejan propiedad intelectual sensible y no pueden permitirse el lujo de enviar telemetría o fragmentos de código a servidores de terceros. Es especialmente valioso para startups tecnológicas y equipos de DevOps que buscan agilidad sin el lastre de memoria que suponen los IDEs basados en Electron tradicionales. En mi opinión profesional, el presupuesto necesario es prácticamente nulo a nivel de licencias (al ser Open Source), permitiendo derivar la inversión exclusivamente al consumo de APIs de IA o a infraestructura local para modelos como Llama 3 o DeepSeek. Lo que más me gusta es que permite una transición fluida para quienes ya dominan la terminal, eliminando la fricción entre "pensar el código" y "ejecutar el comando".

Madurez digital requerida
- **Usuarios y equipo**: Se requiere un nivel técnico medio-alto. No es una herramienta para principiantes que necesitan asistentes visuales constantes; el equipo debe sentirse cómodo trabajando en entornos CLI (Símbolo del sistema/Terminal) y gestionando sus propias claves de API o entornos locales de inferencia.
- **Empresa y departamentos**: La organización debe tener una política clara sobre el uso de IA. Es perfecta para departamentos que ya han superado la fase de experimentación con ChatGPT y buscan integrar agentes que actúen directamente sobre el sistema de archivos y el entorno de ejecución.

Plan orientativo de implantación
Pasos necesarios y estimaciones
- **Evaluación inicial (1-2 días)**: Identificar qué proveedores de LLM se utilizarán (Anthropic, OpenAI o locales vía Ollama) y verificar la compatibilidad de los flujos de trabajo actuales con un entorno ligero.
- **Configuración y Piloto (1 semana)**: Instalación en un grupo reducido de desarrolladores "power users" para configurar el archivo TERAX.md como memoria de contexto del proyecto y validar la integración con los servidores de desarrollo locales (Vite, Next.js).
- **Despliegue y Ajuste (2-3 días)**: Configuración de las variables de entorno y almacenamiento seguro de claves en el llavero del sistema de los usuarios finales.
- **Feedback (Continuo)**: Al ser una herramienta nueva, es vital monitorizar cómo los agentes de IA manejan los "diffs" de código para asegurar que la calidad del software no se degrade por un exceso de automatización.

Necesidades de formación del equipo
La formación no debe centrarse en la interfaz (que es minimalista), sino en el manejo de agentes. Es necesario capacitar al equipo en la revisión de los cambios propuestos por la IA antes de aplicarlos y en el uso del modo Vim si se desea maximizar la velocidad de edición.

Perfiles necesarios
- **Perfiles técnicos necesarios**: Desarrolladores Senior con conocimiento de arquitectura de software para supervisar las propuestas de los agentes.
- **Personal externo recomendado**: No es necesario personal externo debido a la simplicidad de la herramienta y la ausencia de una infraestructura de servidor propietaria que gestionar.

Retorno de la inversión
- **Tiempos**: La mejora en la velocidad de codificación y depuración suele ser visible tras la primera semana de adaptación. Al arrancar en menos de 300ms, se eliminan los tiempos muertos de carga de herramientas pesadas.
- **KPIs**: Reducción del tiempo medio de resolución de bugs (MTTR), incremento en la frecuencia de commits gracias a la integración nativa de Git y ahorro directo en costes de licencias de IDEs privativos.

Otros
- Mi experiencia en implantaciones me lleva a pensar que el uso del archivo TERAX.md como "memoria ram" del proyecto es el factor diferencial; documentar ahí las convenciones de código de la empresa permite que la IA sea mucho más precisa que cualquier asistente genérico.
- Al usarlo te das cuenta de que la integración con la terminal WebGL cambia las reglas del juego para quienes analizan logs pesados mientras codifican, ya que no hay retardo visual.
- Es fundamental recordar que, aunque es ligero, el motor de IA que se conecte por detrás (como Claude 3.5 Sonnet) es el que determinará la calidad de la lógica generada, por lo que no se debe escatimar en la elección del modelo.

TUTORIAL BÁSICO

Instalación
Para instalar Terax correctamente en tu sistema, sigue estos pasos y recomendaciones técnicas:
- En **Windows**, descarga el instalador .exe desde la página de GitHub Releases. Al ser una aplicación de código abierto sin certificar, aparecerá el aviso de SmartScreen; haz clic en **Más información** y luego en **Ejecutar de todas formas**.
- En **macOS**, utiliza el archivo .dmg correspondiente a tu arquitectura (Intel o Apple Silicon M1/M2/M3).
- En **Linux**, tienes opciones para Arch (vía AUR con yay -S terax-bin), AppImage, .deb o .rpm. Si usas Wayland y experimentas problemas de renderizado, ejecuta con la variable WEBKIT_DISABLE_DMABUF_RENDERER=1.
- **Configuración de IA:** Una vez instalado, ve a **Settings -> AI**. Terax es BYOK (Bring Your Own Key), lo que significa que debes introducir tus propias claves de OpenAI, Anthropic, Google Gemini u otros. Tus claves se guardan de forma segura en el llavero del sistema operativo (OS Keychain) y no en texto plano.

Uso en el día a día
Según mi experiencia profesional, Terax destaca por su integración vertical. Estos son los puntos clave para un flujo de trabajo eficiente:
- **Detección de servidores:** Cuando lanzas un comando como npm run dev, Terax detecta automáticamente la URL de localhost y te ofrece abrir una pestaña de **Web Preview** interna. Esto evita saltar constantemente al navegador.
- **Memoria del proyecto (TERAX.md):** Al igual que Cursor usa .cursorrules, Terax utiliza un archivo llamado TERAX.md en la raíz de tu proyecto. En mi opinión profesional, es vital documentar aquí la arquitectura y reglas de estilo para que los agentes de IA no cometan errores recurrentes.
- **Gestión de terminales:** Puedes usar múltiples pestañas que mantienen el contexto de ejecución. Lo que más me gusta es la capacidad de ver el historial de Git con un gráfico real integrado en el propio panel lateral, lo cual es mucho más intuitivo que usar comandos de consola planos.

Trucos de experto
- **Local LLMs:** Si buscas privacidad total, configura Terax para que se conecte a **Ollama** o **LM Studio**. Al usarlo te das cuenta de que la latencia es mínima y no dependes de suscripciones externas para tareas de autocompletado básico.
- **Vim Mode:** Si eres usuario de Vim, actívalo en los ajustes. La implementación sobre CodeMirror 6 es sólida y soporta los comandos de movimiento y edición más comunes, lo que lo convierte en un ADE (AI-native Dev Environment) muy ligero.
- **Slash Commands y @mentions:** En el panel de IA, usa @ para añadir archivos específicos al contexto del agente y para usar snippets de código. Mi experiencia me lleva a pensar que alimentar a la IA solo con los archivos necesarios mejora drásticamente la calidad de las respuestas y reduce el consumo de tokens.
- **Cambio de entorno en Windows:** Terax permite cambiar entre PowerShell local y cualquier distribución de **WSL** instalada directamente desde el selector de pestañas, tratándolos como entornos de primera clase.

Posibles problemas/incidencias
- **Bloqueos en Windows:** Existe una incidencia conocida con ConPTY que puede dejar terminales en blanco tras abrir muchas pestañas rápido. Las versiones actuales (v0.7.3+) incluyen un SPAWN_LOCK para mitigar esto, pero evita el "tab spamming" extremo.
- **Procesos huérfanos:** En Windows, a veces cerrar la Shell no mata los procesos hijos (como servidores de desarrollo). Aunque Terax usa Job Objects para limpiar, mi recomendación es siempre detener los procesos con Ctrl+C antes de cerrar la pestaña.
- **Incompatibilidades de Red:** Si usas VPNs corporativas, la detección de servidores locales para la Web Preview puede fallar. Asegúrate de que las rutas de localhost no estén interceptadas por tu configuración de proxy.

Otros
- **Huella de memoria:** Terax ocupa menos de 10 MB en disco y tiene un arranque en frío de aproximadamente 300 ms, lo que lo hace ideal para máquinas con recursos limitados donde VS Code o Cursor resultan pesados.
- **Privacidad:** Al no requerir cuenta ni tener telemetría integrada, es una de las opciones más seguras para entornos corporativos con políticas estrictas de datos.

PREGUNTAS FRECUENTES

¿Qué es Terax y en qué se diferencia de un IDE tradicional?

Terax es un entorno de desarrollo nativo de IA (AI-native Dev Workspace) diseñado bajo una arquitectura ultraligera que integra terminal, editor de código y agentes en una sola interfaz. A diferencia de IDEs tradicionales como VS Code o IntelliJ, Terax pesa menos de 10 MB, arranca en menos de 300 ms y prioriza una experiencia 'terminal-first' sin la sobrecarga de recursos de las aplicaciones basadas en marcos de trabajo pesados.

¿Cuál es el coste del software y su modelo de licencias?

Terax es un proyecto de código abierto distribuido bajo la licencia Apache 2.0, lo que permite su uso comercial, modificación y distribución gratuita. No existe una cuota de suscripción por el software; el usuario solo asume los costes de consumo de las APIs de modelos de lenguaje (como OpenAI o Anthropic) mediante el sistema 'Bring Your Own Key' (BYOK), o coste cero si se utilizan modelos locales.

¿Cómo garantiza Terax la privacidad y el cumplimiento de la normativa de datos?

El entorno destaca por su arquitectura de 'zero telemetry', lo que significa que no recolecta ni envía datos de uso o código a servidores externos. Las claves de API se almacenan de forma segura en el llavero (keyring) del sistema operativo del usuario. Al permitir la integración con modelos locales mediante Ollama o LM Studio, el código fuente de la empresa nunca sale del entorno local, cumpliendo con los estándares de seguridad más estrictos.

¿Es posible utilizarlo sin conexión a internet o con modelos locales?

Sí, Terax ofrece integración nativa con herramientas como Ollama, LM Studio y MLX. Esto permite ejecutar modelos de lenguaje complejos directamente en el hardware del usuario, garantizando la soberanía de los datos, reduciendo la latencia y permitiendo el desarrollo de software en entornos sin acceso a la red.

¿Es compatible con todas las extensiones de VS Code?

No. Terax está diseñado para ser una herramienta minimalista y rápida, por lo que no es compatible con el ecosistema de extensiones pesadas de VS Code. Utiliza el editor CodeMirror 6 y está enfocado en desarrolladores que prefieren un flujo de trabajo basado en terminal y agentes de IA en lugar de configuraciones visuales extensas.

¿Qué sistemas operativos soporta y cómo es su instalación?

Es multiplataforma y está disponible para macOS (Intel y Apple Silicon), Windows (x64) y diversas distribuciones de Linux (AppImage, .deb, .rpm y AUR). En Windows, debido a que es software de código abierto que puede no estar firmado digitalmente en todas sus versiones, el sistema SmartScreen puede mostrar una advertencia durante la instalación inicial.

¿Qué capacidades tienen los agentes de IA dentro del entorno?

Los agentes de Terax tienen permisos para ejecutar herramientas avanzadas: pueden leer la estructura de archivos, ejecutar comandos en la terminal, analizar logs en tiempo real y proponer cambios en el código. Los cambios sugeridos se presentan mediante un sistema de 'diffs', permitiendo que el desarrollador revise y apruebe cada modificación antes de que se aplique al sistema de archivos.

¿Qué nivel de conocimientos técnicos se requiere para operar la herramienta?

El nivel técnico requerido es medio. El profesional debe estar familiarizado con el manejo de la línea de comandos (Bash, Zsh o PowerShell), el flujo de trabajo básico de Git y la configuración de entornos de IA (gestión de endpoints y claves API). No es una herramienta recomendada para perfiles sin experiencia en el uso de terminales.

CONTRATOS Y CONDICIONES

Opinión inicial
Tras verificar los contratos y las condiciones de la herramienta, mi opinión profesional es que Terax representa un caso excepcional de cumplimiento normativo por diseño (Privacy by Design). Al ser un entorno de desarrollo "zero telemetry" y código abierto bajo licencia Apache 2.0, resuelve por defecto gran parte de las fricciones legales que las empresas españolas encuentran con IDEs propietarios como Cursor o VS Code. Según documentos consultados, el control sobre el flujo de datos es absoluto por parte del usuario, lo que facilita enormemente la adecuación al RGPD y a la Ley de Inteligencia Artificial de la UE, ya que la responsabilidad del tratamiento se desplaza directamente al cliente y no al proveedor del software.

Principales recomendaciones
- Implementar una política de uso corporativo que especifique qué proveedores de modelos (OpenAI, Anthropic, Ollama) están autorizados según la criticidad del proyecto.
- Si se maneja código con secretos industriales o datos de carácter personal sensibles, priorizar el uso del modo local mediante Ollama o LM Studio para garantizar que la IP no abandone el perímetro de la empresa.
- Verificar que las claves de API (BYOK) se gestionen de forma centralizada y no se compartan entre empleados para mantener la trazabilidad del gasto y el acceso.
- Realizar una auditoría periódica del código fuente en Github para asegurar que las actualizaciones mantienen la ausencia de telemetría prometida.

Ley de Inteligencia Artificial (AI Act)
El impacto inicial es bajo/medio. Según la IA Act, Terax actúa como una interfaz/herramienta de propósito general.
- Al no aplicar sistemas de puntuación social o vigilancia proactiva, no entra en categorías de riesgo prohibido.
- La responsabilidad de transparencia (art. 52) recae en el usuario si utiliza la herramienta para generar contenido que deba ser etiquetado como artificial.
- La empresa debe asegurarse de que los modelos conectados mediante API (como GPT-4 o Claude) cumplan con las obligaciones para modelos de IA de propósito general con riesgo sistémico si fuera el caso.

Privacidad y protección de datos
- **Responsabilidades**: La empresa española actúa como Responsable del Tratamiento. Terax no es Encargado del Tratamiento porque no tiene acceso a los datos; es meramente el proveedor del software local.
- **Ubicación de los datos**: Los datos permanecen en el equipo local del desarrollador. No hay almacenamiento en la nube del fabricante.
- **Transferencia internacional**: No existen transferencias internacionales por parte de Terax. Sin embargo, si el usuario configura APIs de terceros (como OpenAI), la transferencia ocurrirá hacia esos proveedores, requiriendo verificar si cuentan con Marcos de Privacidad (Data Privacy Framework) activos.
- **Derechos ARCO**: Al ser una herramienta local sin cuentas de usuario, el ejercicio de estos derechos es responsabilidad directa de la empresa sobre sus propios archivos y logs.

Propiedad intelectual
- **Propiedad de datos**: La empresa conserva el 100% de la propiedad sobre el código fuente analizado.
- **Propiedad del resultado**: Bajo la legislación española y europea, el código generado por IA no tiene derechos de autor per se, pero la integración y selección humana del código propuesto por los agentes de Terax permite que la obra resultante sea protegible como propiedad intelectual de la empresa. La licencia Apache 2.0 de la herramienta garantiza que el software no reclama derechos sobre los outputs creados con ella.

Usos y prohibiciones
- **Usos admitidos**: Desarrollo de software comercial, refactorización de código legacy, auditoría de seguridad interna y automatización de tareas DevOps.
- **Usos prohibidos**: No debe utilizarse para procesar datos personales reales en el prompt de la IA a menos que el proveedor de la API tenga un DPA (Data Processing Agreement) firmado con la empresa. Se prohíbe el uso de la herramienta para actividades de hacking malicioso que contravengan la normativa de ciberseguridad española.

Seguridad y certificaciones
- **Seguridad**: Tras usarlo, he verificado que el almacenamiento de claves se realiza en el llavero nativo del

sistema (Keyring), lo que cumple con estándares de seguridad a nivel de endpoint.
- **Certificaciones**: Al ser un proyecto Open Source comunitario, no cuenta con certificaciones ISO 27001 o SOC2 propias del fabricante, pero su naturaleza auditable permite que la empresa incluya la herramienta en su propio perímetro certificado.

Otros
- Es relevante destacar que Terax no requiere "Sign-in", lo que elimina el riesgo de perfiles de usuario y rastreo de comportamiento (profiling), alineándose perfectamente con la Directiva ePrivacy sobre el seguimiento de usuarios.

Fuentes consultadas:
- Licencia Apache 2.0
- Documentación de privacidad y seguridad
- Repositorio de código oficial
- Configuración de modelos locales