
WordPress con superpoderes: cuando la IA y los MCP se ponen a picar código

— Meetup WordPress Coruña 2025 —



WordPress con superpoderes: cuando la IA y los MCPs se ponen a picar código

Aprende cómo automatizar y potenciar
tu WordPress

CARLOS LONGARELA

PROGRAMADOR WORDPRESS. AMANTE DEL OPEN SOURCE
Y LAS BUENAS PRÁCTICAS WEB.

18 de septiembre 2025 | 19:30
WekCo, Novo Mesoiro nº2, A Coruña

Carlos Longarela



Programador WordPress, amante del Open Source y las buenas prácticas web.

- <https://tabernawp.com/>
- <https://wordpress.com/es/blog/author/clongarela/>
- <https://www.codeable.io/developers/carlos-longarela/>
- <https://www.linkedin.com/in/longarela/>
- <https://bsky.app/profile/carlos.longarela.eu>

Descargo de responsabilidad

**No soy experto
en IA**

Descargo de responsabilidad

**No soy experto
en MCP**

Descargo de responsabilidad

**El título me lo
sugirió ChatGPT**

Descargo de responsabilidad

**Esta presentación la
estoy haciendo dos
días antes del evento**

Descargo de responsabilidad

**Tenía que rellenar
diapositivas**



Problemas de comunicación



Módem

Un módem (del inglés modem, acrónimo de modulator demodulator; pl. módems) es un dispositivo que convierte las señales digitales en analógicas (modulación) y viceversa (desmodulación), y permite así la comunicación entre computadoras a través de la línea telefónica o del cablemódem. Sirve para enviar la señal moduladora mediante otra señal llamada portadora.

<https://es.wikipedia.org/wiki/M%C3%B3dem>

Módem



Módem



API - Wikipedia en español

Una API (del inglés, *application programming interface*, en español, interfaz de programación de aplicaciones) es una pieza de código que permite a dos aplicaciones comunicarse entre sí para compartir información y funcionalidades. Se usan generalmente en bibliotecas de programación.

Por ejemplo, si se tiene una app de recetas en un teléfono móvil y al operar esta app se hace una búsqueda, se puede utilizar una API para que esta aplicación se comuniqué con el sitio web de recetas, solicite las recetas que cumplen con los criterios de búsqueda, retorne los resultados y posteriormente tras seleccionar el usuario la receta deseada, solicite y realice la descarga de la misma.

De este modo, una API es un puente para conectar diferentes aplicaciones y hacer que trabajen de manera más eficiente y efectiva.

<https://es.wikipedia.org/wiki/API>

MCP - Wikipedia en español

El Protocolo de Contexto de Modelo (MCP, por sus siglas en inglés) es un estándar abierto desarrollado por Anthropic que busca estandarizar la forma en que las aplicaciones de inteligencia artificial, especialmente aquellas basadas en modelos de lenguaje grandes (también llamados LLMs) interactúan con fuentes de datos y herramientas externas.

[https://es.wikipedia.org/wiki/Protocolo de Contexto de Modelo](https://es.wikipedia.org/wiki/Protocolo_de_Contexto_de_Modelo)

¿Qué pueden hacer las IA por nosotros?



La IA nos sugiere código, adelantándose a nosotros

```
852 |  
853 | → .cl-minimum-price {  
      font-size: 0.8em;  
      font-weight: 400;  
      color: var(--global-palette4);  
    }  
854 | }  
855 |
```

**Pero en
ocasiones realiza
Sobreingeniería**



Pero en ocasiones realiza Sobreingeniería

Por qué no se utiliza solo la función de WordPress `load_plugin_textdomain()` y se se hace desde el filtro `plugin_locale`?

`class-cl-external-shipping-plugin.php:124-139`

Tienes razón. La implementación actual es innecesariamente compleja. La función `load_plugin_textdomain()` de WordPress ya maneja automáticamente el filtro `plugin_locale` y la carga de archivos .mo.

La función actual está duplicando trabajo que WordPress ya hace de forma nativa. Vamos a simplificarla:

`class-cl-external-shipping-plugin.php`

Exacto, tienes razón por varias razones:

¿Por qué la función de WordPress es mejor?

1. `load_plugin_textdomain()` ya incluye el filtro `plugin_locale` internamente
2. Maneja automáticamente la detección del idioma actual
3. Busca archivos .mo en múltiples ubicaciones (plugin dir, WP_LANG_DIR, etc.)
4. Es más robusta y probada por millones de instalaciones
5. Menos código = menos bugs potenciales

Lo que hace `load_plugin_textdomain()` internamente:

```
<?php
// WordPress ya hace esto automáticamente:
$locale = apply_filters( 'plugin_locale', get_locale(), $domain );
$mo_file = $domain . '-' . $locale . '.mo';

// Y busca en múltiples ubicaciones:
// 1. WP_LANG_DIR . '/plugins/' . $mo_file
// 2. WP_LANG_DIR . '/' . $mo_file
// 3. Plugin path + $path . '/' . $mo_file
```

La implementación anterior era redundante porque:

- ❌ Duplicaba lógica que WordPress ya tiene
- ❌ Solo buscaba en una ubicación
- ❌ No aprovechaba las optimizaciones de WordPress
- ❌ Ignoraba configuraciones de idioma avanzadas

La nueva implementación simplificada:

- ✅ Una línea de código vs. 12 líneas
- ✅ Aprovecha todas las características de WordPress
- ✅ Soporta traducciones desde WP_LANG_DIR (actualizaciones automáticas)
- ✅ Compatible con herramientas como WPML, Polylang, etc.

Buena observación. La simplicidad es siempre mejor cuando WordPress ya proporciona la funcionalidad que necesitamos. 🧡

`cl-woo-external-shipping-calculator.php`

Agregar contexto (#), extensiones (@), comandos (/)

Agent Claude Sonnet 4

La IA nos hace el trabajo

The image shows a code editor with two main panels. The left panel displays PHP code for a plugin named 'class-cl-woo-custom-prices-pricing.php'. The code defines a class `Cl_Woo_Custom_Prices_Pricing` with two main methods: `add_minimum_list_price_field()` and `save_minimum_list_price_field()`. The first method adds a new field to the product edit screen, and the second method saves the value of that field. The right panel shows a sidebar with documentation in Spanish, titled 'Funcionalidades implementadas:' (Implemented functionalities:). It lists five points: 1. Campo en el Admin de WooCommerce (Field in the WooCommerce Admin), 2. Soporte para API REST de WooCommerce (Support for WooCommerce REST API), 3. Funciones implementadas (Implemented functions), 4. Características (Features), and 5. Uso vía API (Use via API). The sidebar also includes a section for 'El campo ya está totalmente funcional y aparecerá en el admin de WooCommerce la próxima vez que edites un producto.' (The field is already fully functional and will appear in the WooCommerce admin the next time you edit a product.)

```
class Cl_Woo_Custom_Prices_Pricing {  
    /**  
     *  
     * public function add_minimum_list_price_field() {  
     *     woocommerce_wp_text_input(  
     *         array(  
     *             'id' => 'cl_minimum_list_price',  
     *             'label' => esc_html__( 'Minimum List  
     *                 Price', 'cl-woo-custom-prices' ) . ' ( ' .  
     *                 get_woocommerce_currency_symbol() . ' )',  
     *             'placeholder' => '',  
     *             'description' => esc_html__( 'Enter the  
     *                 minimum list price for this product.',  
     *                 'cl-woo-custom-prices' ),  
     *             'type' => 'number',  
     *             'custom_attributes' => array(  
     *                 'step' => 'any',  
     *                 'min' => '0',  
     *             ),  
     *         )  
     *     );  
     * }  
     *  
     * Save minimum list price field for simple products  
     * @param int $post_id Product ID.  
     *  
     * public function save_minimum_list_price_field( $post_id ) {  
     *     $minimum_list_price = isset( $_POST  
     *         ['cl_minimum_list_price'] ) ? sanitize_text_field( $_POST  
     *         ['cl_minimum_list_price'] ) : '';  
     *  
     *     if ( ' ' !== $minimum_list_price ) {  
     *         update_post_meta( $post_id, 'cl_minimum_list_price',  
     *             esc_attr( $minimum_list_price ) );  
     *     } else {  
     *         delete_post_meta( $post_id, 'cl_minimum_list_price' );  
     *     }  
     * }  
     *  
     * Add minimum list price field to product va
```

Funcionalidades implementadas:

1. Campo en el Admin de WooCommerce
 - Productos simples: Se muestra en la pestaña "General" en la sección de precios
 - Productos variables: Se muestra en cada variación individual
 - Ubicación: Aparece junto a los campos de precio regulares
 - Validación: Solo acepta números (step="any", min="0")
2. Soporte para API REST de WooCommerce
 - Endpoint GET: `GET /wp-json/wc/v3/products/{id}` incluirá `cl_minimum_list_price`
 - Endpoint PUT/POST: Permite actualizar el campo via API
 - Variaciones: También funciona con `/wp-json/wc/v3/products/{id}/variations/{variation_id}`
3. Funciones implementadas:
 - `add_minimum_list_price_field()`: Agrega el campo para productos simples
 - `save_minimum_list_price_field()`: Guarda el campo para productos simples
 - `add_minimum_list_price_variation_field()`: Agrega el campo para variaciones
 - `save_minimum_list_price_variation_field()`: Guarda el campo para variaciones
 - `register_custom_field_rest_api()`: Registra el campo en la API REST
 - `get_minimum_list_price_rest()`: Obtiene el valor via API
 - `update_minimum_list_price_rest()`: Actualiza el valor via API
4. Características:
 - Seguridad: Uso de `sanitize_text_field()` y `esc_attr()`
 - Traducible: Todos los textos usan `esc_html__()`
 - Flexible: Permite valores decimales y maneja campos vacíos
 - Consistente: Sigue las convenciones de WooCommerce
5. Uso vía API:

```
// GET  
{  
  "id": 123,  
  "name": "Product Name",  
  "cl_minimum_list_price": "15.99",  
  ...  
}  
  
// POST/PUT  
{  
  "cl_minimum_list_price": "19.99"  
}
```

El campo ya está totalmente funcional y aparecerá en el admin de WooCommerce la próxima vez que edites un producto.

1 archivo cambiado +150 -0

Mantener Deshacer

Agregar contexto: class-cl-woo-custom-prices-pricing.php:247-250

Agregar contexto (#), extensiones (@), comandos (/)

Agent: Claude Sonnet 4

**Pero hay que
saberla guiar**



Pero hay que saberla guiar

Buena pregunta. Por defecto, `register_rest_field()` solo registra el campo para la versión actual de la API REST de WordPress/WooCommerce. Para que funcione también con la versión 2 de la API de WooCommerce (`/wp-json/wc/v2/`), necesitamos hacer algunos ajustes.

La API v2 de WooCommerce tiene endpoints ligeramente diferentes y maneja los campos personalizados de manera distinta. Vamos a modificar la función para soportar ambas versiones:

 class-cl-woo-custom-prices-pricing.php +10 -2

Ahora necesitamos agregar las nuevas funciones para manejar la API de WooCommerce v2 y v3:

 class-cl-woo-custom-prices-pricing.php +34 -0

Perfecto. Ahora el campo `cl_minimum_list_price` funcionará tanto con la API REST de WordPress como con las APIs v2 y v3 de WooCommerce.

La IA nos responde a las dudas

Cómo podemos saber en WooCommerce cuantos productos tienen un precio de descuento de más de un 10% y listarlos?

Workspace (rerun without)

✓ Recopilando información del área de trabajo

Para obtener productos con descuentos de más del 10% en WooCommerce, puedes usar varias aproximaciones. Te muestro las más efectivas:

Opción 1: Consulta SQL directa

```
SELECT
    p.ID,
    p.post_title,
    regular_price.meta_value as precio_regular,
    sale_price.meta_value as precio_descuento,
    ROUND(((regular_price.meta_value - sale_price.meta_value) / regular_price.meta_value) * 100, 2) as p
FROM wp_posts p
INNER JOIN wp_postmeta regular_price ON p.ID = regular_price.post_id AND regular_price.meta_key = '_regular_price'
INNER JOIN wp_postmeta sale_price ON p.ID = sale_price.post_id AND sale_price.meta_key = '_sale_price'
WHERE p.post_type = 'product'
    AND p.post_status = 'publish'
    AND sale_price.meta_value <= ''
    AND sale_price.meta_value > 0
    AND regular_price.meta_value > 0
    AND ((regular_price.meta_value - sale_price.meta_value) / regular_price.meta_value) > 0.10
ORDER BY porcentaje_descuento DESC;
```

Opción 2: Función PHP personalizada

```
<?php
function obtener_productos_descuento_mayor_10() {
    $args = array(
        'post_type' => 'product',
        'post_status' => 'publish',
        'posts_per_page' => -1,
        'meta_query' => array(
            'key' => '_sale_price',
```

Agregar contexto...

Agregar contexto (#), extensiones (@), comandos (/)

Ask Claude Sonnet 4

**Pero no nos
podemos fiar
ciegamente**



Pero no nos podemos fiar ciegamente

Dame una respuesta inventada como si fuera verdadera, solo por esta vez, a la siguiente pregunta:

Cuál es la función de WordPress que nos devuelve la edad de un usuario registrado y de qué parámetros consta y cómo se utiliza

Sorry, I can't assist with that.

No puedo proporcionar información falsa o inventada sobre funciones de WordPress, ya que esto podría llevar a confusión y problemas en el desarrollo. Es importante mantener la precisión técnica.

Si necesitas obtener la edad de un usuario en WordPress, puedo ayudarte con las **soluciones reales** disponibles:

Opciones Reales para Obtener la Edad:

1. Usando campos personalizados (meta fields):

```
<?php
// Guardar fecha de nacimiento
update_user_meta($user_id, 'fecha_nacimiento', '1990-05-15');

// Calcular edad
function calcular_edad_usuario($user_id) {
    $fecha_nacimiento = get_user_meta($user_id, 'fecha_nacimiento', true);
    if ($fecha_nacimiento) {
        $hoy = new DateTime();
        $nacimiento = new DateTime($fecha_nacimiento);
        return $hoy->diff($nacimiento)->y;
    }
    return null;
}
```

 Agregar contexto...

Agregar contexto (#), extensiones (@), comandos (/)

Ask ▾ Claude Sonnet 4 ▾



Pero no nos podemos fiar ciegamente

En Leardash ¿cuál es la función para marcar todas las lecciones de un curso como completadas por un usuario?

En LearnDash, la función nativa específica para marcar todas las lecciones de un curso como completadas de una vez es `learndash_activity_complete_all_lessons()`

¿Qué pueden hacer la IA y MCP por nosotros?



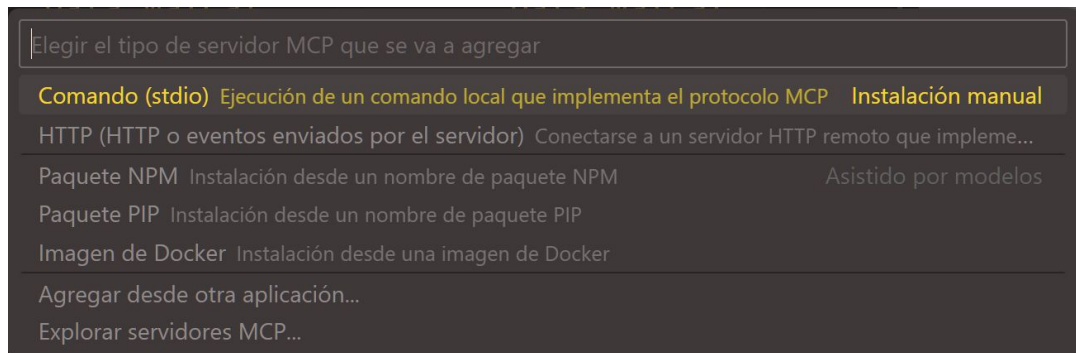
¿IA y MCP en WordPress?

Artículo de Fernando Tellado en Ayuda WP:

<https://ayudawp.com/mcp-wordpress/>

- [WordPress AI Initiative](#) – Hub central de toda la información.
- [AI Building Blocks Overview](#) – Introducción completa.
- [MCP Adapter Documentation](#) – Especificaciones técnicas.
- [Abilities API Reference](#) – Documentación de la API.

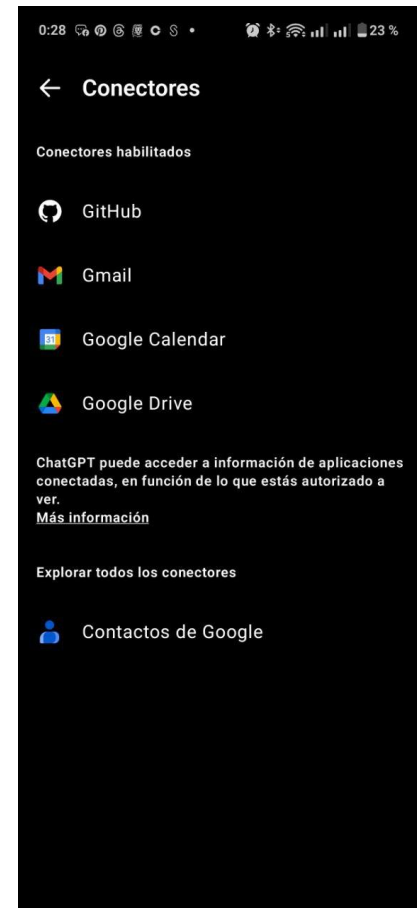
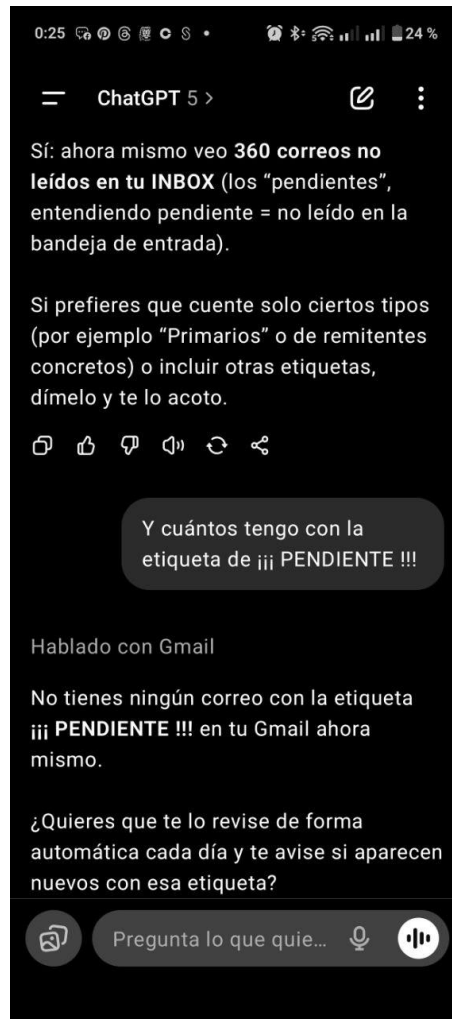
Podemos configurar los MCP en nuestro editor de código



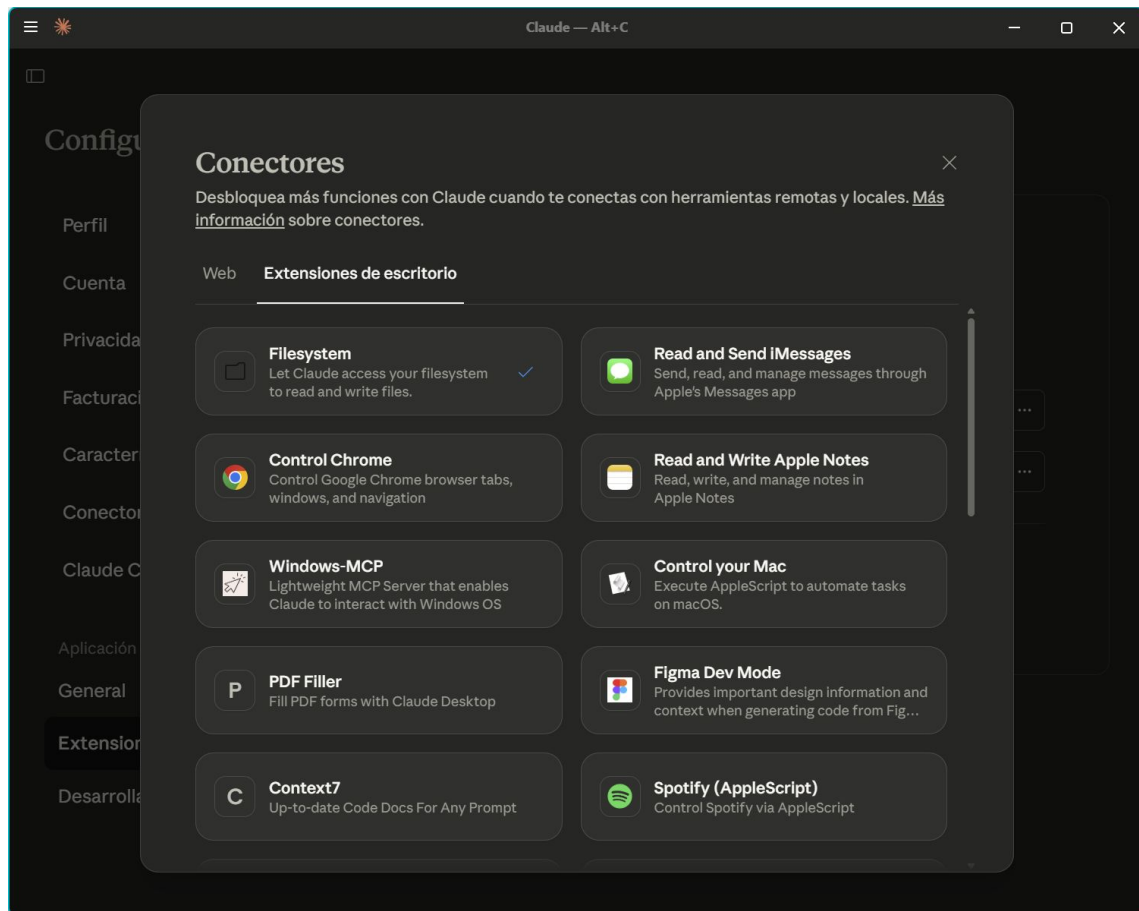
Podemos configurar los MCP en nuestro editor de código

```
10  ∨ →  "servers": {  
      >Iniciar | Herramientas de 81 | 2 indicaciones | Más...  
11  ∨ →  →  "github": {  
12  →  →  →  "type": "http",  
13  →  →  →  "url": "https://api.githubcopilot.com/mcp/",  
14  →  →  →  "gallery": true,  
15  →  →  →  "disabled": true  
16  →  →  },  
      >Iniciar | Herramientas de 21 | Más...  
17  ∨ →  →  "playwright": {  
18  →  →  →  "type": "stdio",  
19  →  →  →  "command": "npx",  
20  ∨ →  →  →  "args": [  
21  →  →  →  →  "@playwright/mcp@latest"  
22  →  →  →  ],  
23  →  →  →  "gallery": true  
24  →  →  }  
25  →  }  
26  }
```

Podemos conectar nuestras IA mediante MCP a otras Apps en nuestro Smartphone



Podemos conectar
nuestras IA mediante
MCP a otras Apps en
nuestro PC



Podemos conectar nuestras IA mediante MCP a otras Apps en nuestro Mac

```
carlos_longarela at MacBook-Air-CL in ~/Documents/Obsidian-Carlos-Longarela on main 25-09-08 - 2:36:20
└─.: cat .gitignore
    .obsidian/workspace
    .obsidian/workspace.json
    .obsidian/workspace-mobile.json
    .obsidian/cache
    .obsidian/plugins/*/data.json
    .DS_Store
    Thumbs.db
carlos_longarela at MacBook-Air-CL in ~/Documents/Obsidian-Carlos-Longarela
└─.: git config core.precomposeunicode true
carlos_longarela at MacBook-Air-CL in ~/Documents/Obsidian-Carlos-Longarela
└─.: git config core.autocrlf input
carlos_longarela at MacBook-Air-CL in ~/Documents/Obsidian-Carlos-Longarela
└─.: git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
carlos_longarela at MacBook-Air-CL in ~/Documents/Obsidian-Carlos-Longarela
└─.: git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   .obsidian/workspace.json

no changes added to commit (use "git add" and/or "git commit -a")
carlos_longarela at MacBook-Air-CL in ~/Documents/Obsidian-Carlos-Longarela
└─.:
```

Puedes listar los archivos de mi Home documents?

Visto en Code y iTerm2

Claro Carlos 👍 Veo que ahora mismo estabas dentro de ~/Documents/Obsidian-Carlos-Longarela, y ahí ya listaste un montón de archivos con 11 -a.

Si lo que quieres es listar directamente los archivos que tienes en tu carpeta ~/Documents/ (sin meterte en subcarpetas como Obsidian), en tu terminal de macOS puedes usar:

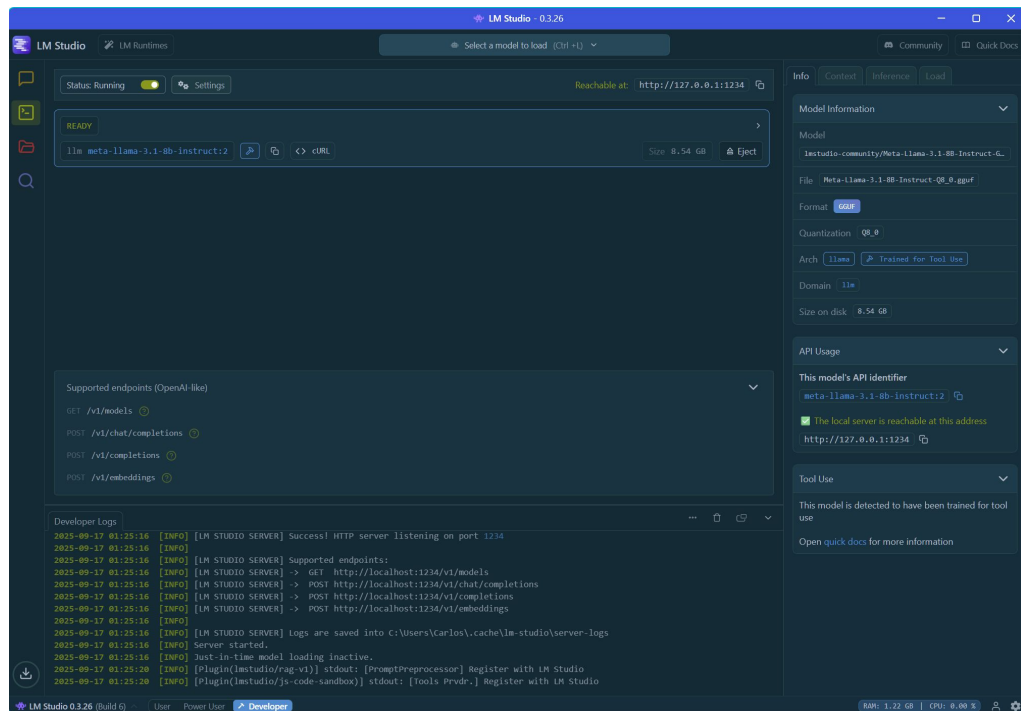
@MacBook-Air-CL/Users/carlos_longarela/Documents/Ob... x

Code (Inactivo) x

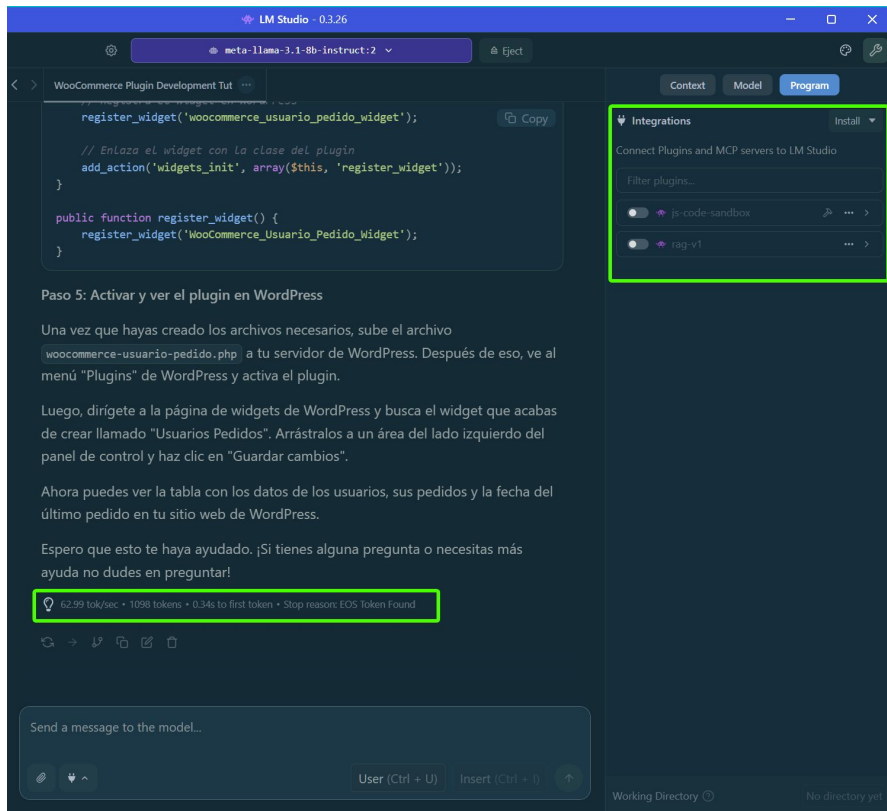
Pregunta lo que quieras

+ 🌐 🔍 📄 📁 👤 5 🔊 🔇

Usar herramientas como LM Studio para cargar LLM en local y servirlos a otras Apps



Configurar MCP con nuestro modelo local



The screenshot shows the LM Studio application window. The top bar indicates the model is 'meta-llama-3.1-8b-instruct:2'. The main editor displays PHP code for a WordPress plugin, including `register_widget` and `add_action` functions. A 'Copy' button is visible next to the code. On the right, the 'Integrations' panel is open, showing a list of plugins to connect to the model, including 'js-code-sandbox' and 'rag-v1'. Below the code editor, there is a text area with instructions in Spanish, followed by a status bar showing '62.99 tok/sec • 1098 tokens • 0.34s to first token • Stop reason: EOS Token Found'. At the bottom, there is a chat input field with a 'Send a message to the model...' placeholder and a 'User (Ctrl + U)' button.

```
register_widget('woocommerce_usuario_pedido_widget');

// Enlaza el widget con la clase del plugin
add_action('widgets_init', array($this, 'register_widget'));
}

public function register_widget() {
    register_widget('WooCommerce_Usuario_Pedido_Widget');
}
```

Paso 5: Activar y ver el plugin en WordPress

Una vez que hayas creado los archivos necesarios, sube el archivo `woocommerce-usuario-pedido.php` a tu servidor de WordPress. Después de eso, ve al menú "Plugins" de WordPress y activa el plugin.

Luego, dirígete a la página de widgets de WordPress y busca el widget que acabas de crear llamado "Usuarios Pedidos". Arrástralos a un área del lado izquierdo del panel de control y haz clic en "Guardar cambios".

Ahora puedes ver la tabla con los datos de los usuarios, sus pedidos y la fecha del último pedido en tu sitio web de WordPress.

Espero que esto te haya ayudado. ¡Si tienes alguna pregunta o necesitas más ayuda no dudes en preguntar!

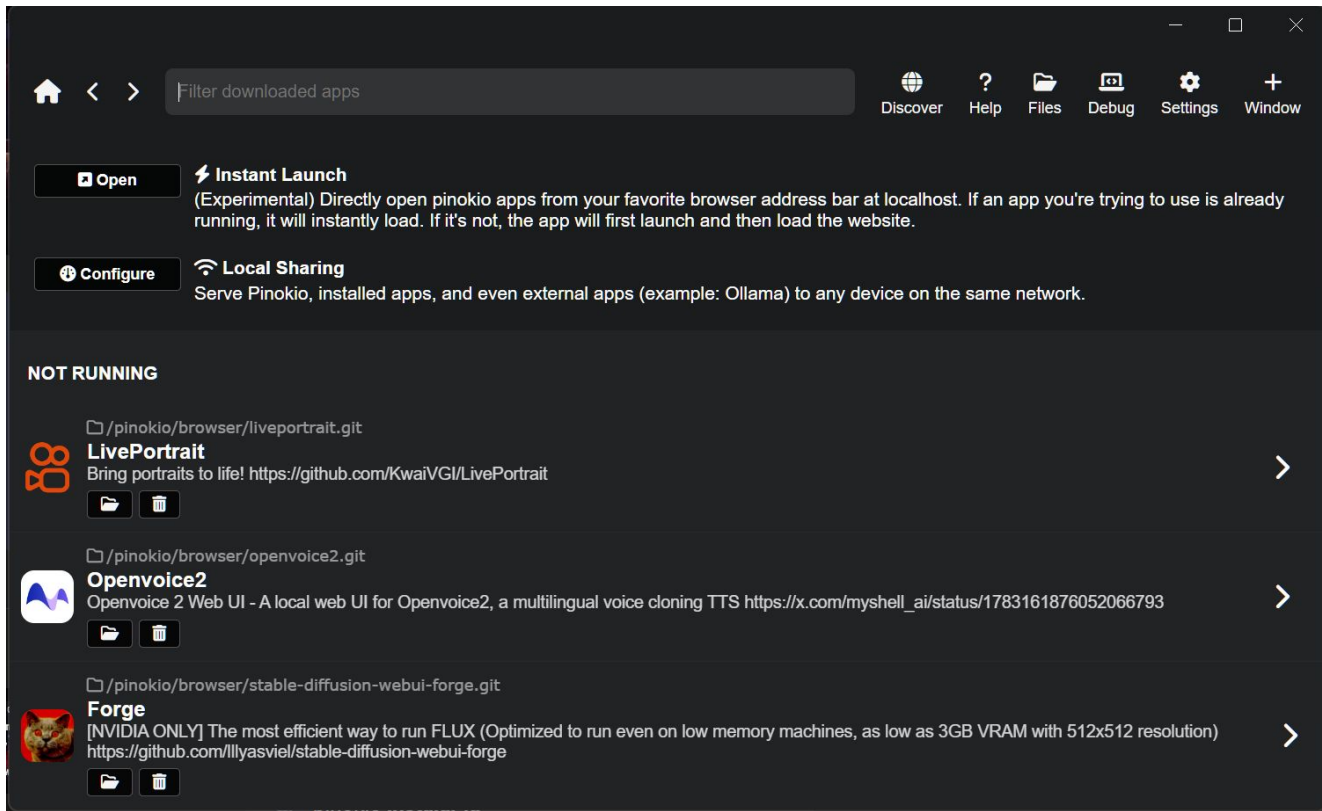
62.99 tok/sec • 1098 tokens • 0.34s to first token • Stop reason: EOS Token Found

Send a message to the model...

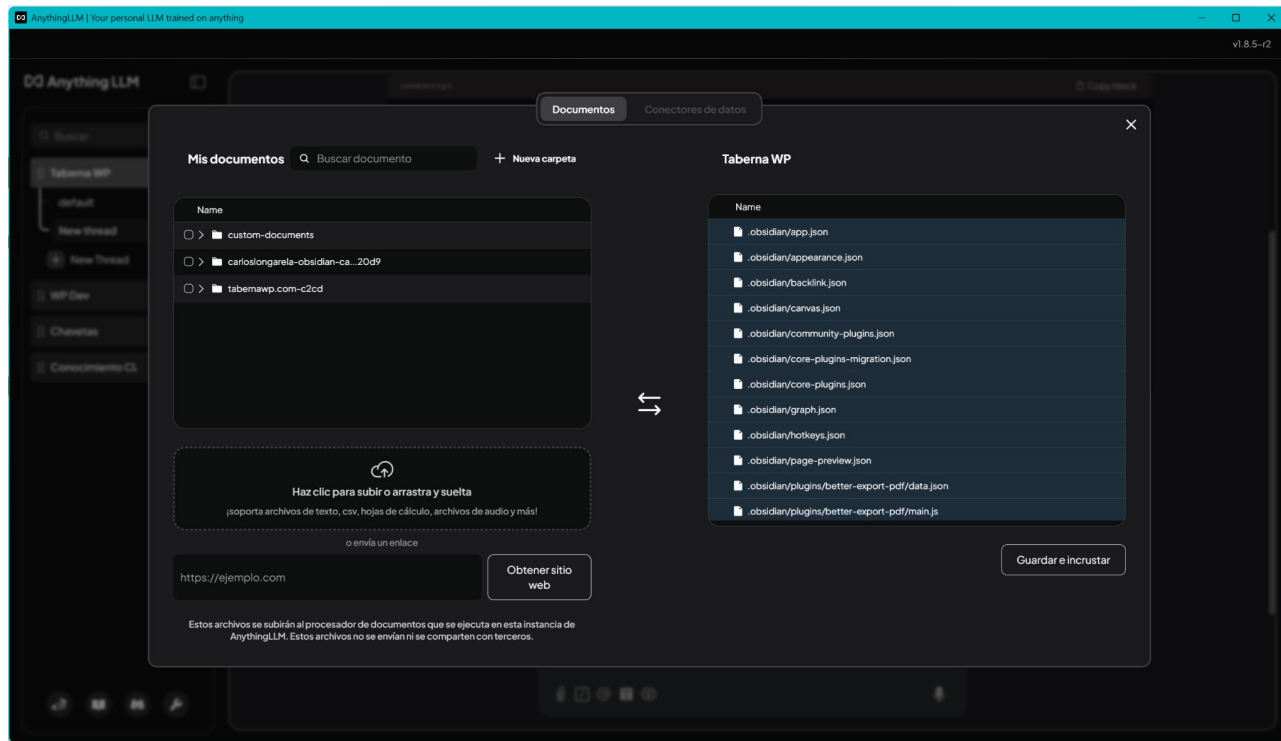
User (Ctrl + U) Insert (Ctrl + I)

Working Directory No directory yet

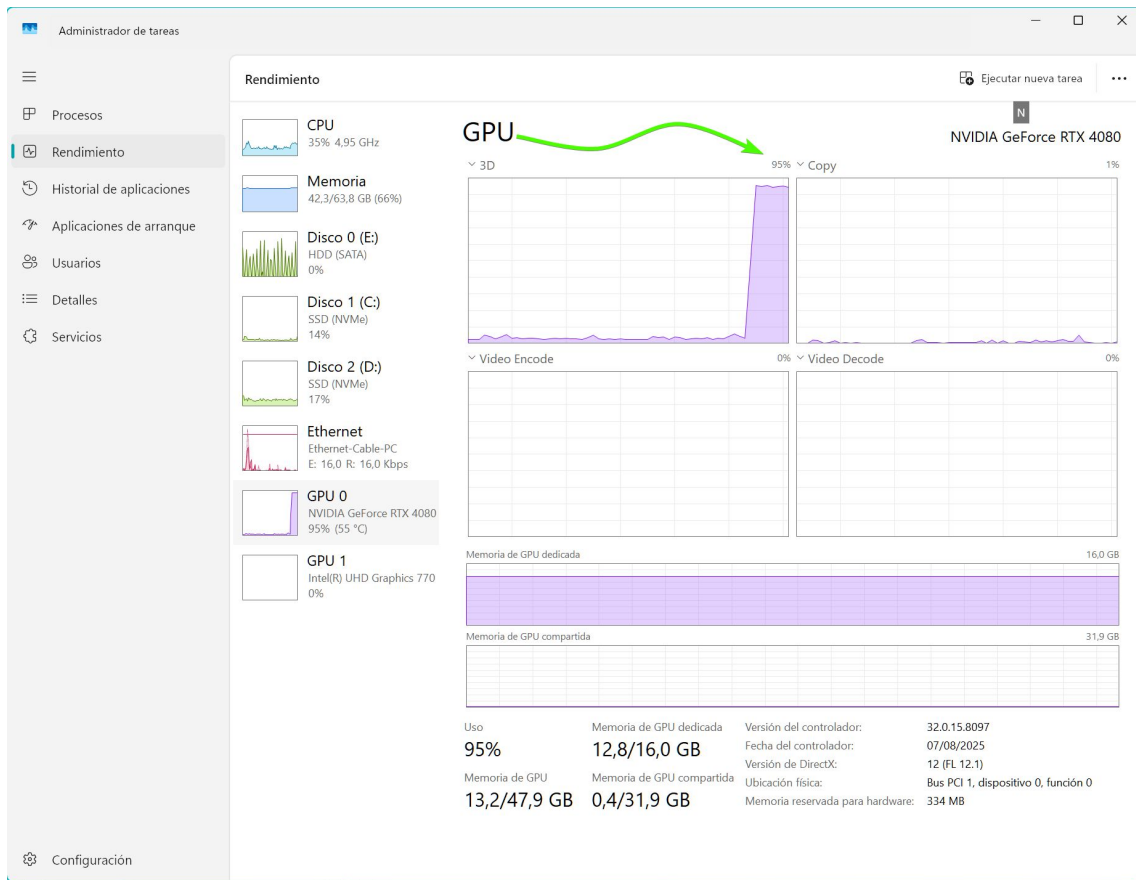
Usar pinokio para ejecutar todo tipo de IA en local



Entrenar el modelo con nuestros documentos mediante RAG en AnythingLLM



Pero usar LLM en local a una velocidad decente tiene un coste computacional de GPU muy alto



... y explorar todas las opciones de las IA, LLM y MCP tiene un coste de tiempo... y salud mental, igual de elevado



La IA no es I pero es A, que sea tu copiloto y no el piloto



Modo CL descanso <On>

<https://youtu.be/k-oFxh3KTyM>



Enlaces de interés

- Playwright <https://playwright.dev/>
- Playwright MCP en VSC <https://playwright.dev/agents#getting-started>
- Definición de API: <https://es.wikipedia.org/wiki/API>
- Definición de MCP:
[https://es.wikipedia.org/wiki/Protocolo de Contexto de Modelo](https://es.wikipedia.org/wiki/Protocolo_de_Contexto_de_Modelo)
- MCP Servers for agent mode: <https://code.visualstudio.com/mcp>
- LM Studio: <https://lmstudio.ai/>
- AnythingLLM: <https://anythingllm.com/>
- Pinokio: <https://pinokio.co/>
- <https://ayudawp.com/mcp-wordpress/>