Conversación con Gemini

Exported: 12/6/2026 11:53:13

Jordi H.

Has dicho hace poco se ha cambiado el sistema de tarificación en github copilot y el resultado es que mi cuenta (con un nivel medio de tokens) era operativa casi todo el mes, llegando al final del periodo con el contador baj o pero funcional. con la nueva tarificación me quedo sin tokens en muy pocos días. tengo la capacidad de ejecutar un LLM en local pero me gusta mucho todo el sistema de github. hay alguna manera de mantener una operativa mixta, con todas las funciones de github copilot pero sin el yugo de los tokens, ya que la IA la pon dría yo con mi modelo ejecutándose en mi serv

Gemini

Entiendo perfectamente tu frustración. El cambio en las políticas de cuotas y el recuento estricto de tokens de GitHub Copilot ha dejado a muchos usuarios intensivos contra las cuerdas a mitad de mes. Es una molesti a perder el flujo de trabajo cuando el contador llega a cero.La buena noticia es que**sí es posible mantener u** ue con ciertos matices de configuración. GitHub ha comenzado a abrir la mano con la filosofía**BYOM (Brin** odel)**, pero la integración nativa absoluta (que use tu servidor local de forma automática como fallback tran sparente cuando te quedas sin tokens) todavía está a medio camino en la extensión oficial.Aquí tiene s las mejores opciones para mantener todo el ecosistema de GitHub que te gusta pero derivando la carga de computación a tu propio servidor.

Opción 1: El enfoque nativo (BYOM en GitHub Copilot)

Recientemente, GitHub ha integrado la capacidad de añadir proveedores de modelos personalizados y endpoints compatibles con OpenAI directamente en la extensión de Copilot para VS Code.Si en tu servido r ejecutas tu modelo mediante un backend que exponga una API compatible (como**Ollama**, **vLLM** o **LM Studio**), puedes enlazar

Cómo configurarlo:

1. Asegúrate de tener la extensión de Copilot actualizada en VS Code.
2. Abre la configuración de VS Code (Ctrl + ,) y busca Copilot: Language Models o accede directamente a los ajustes del Chat de Copilot
3. En la sección de proveedores, puedes añadir un endpoint personalizado apuntando a tu servidor local (por ejemplo, http://<IP_DE_TU_SERVIDOR>:11434 si usas Ollama)
4. En el selector de modelos de la interfaz de Copilot Chat, ahora podrás desplegar la lista y elegir tu modelo local (por ejemplo, un DeepSeek-Coder o Llama-3 que tengas corriendo en tu máquina)

⚠ **El gran "pero" de la opción nativa:** Aunque uses tu propia infraestructura y tu modelo para procesar la respuesta, **sigues necesitando tener la sesión iniciada con una cuenta activa de GitHub Copilot** para que l a extensión valide el entorno. Además, algunas funciones avanzadas de agente (como @workspace) o el autocompletado en línea inline (el ghost text que aparece mientras escribes) pueden seguir intentando vali dar cuotas contra los servidores de GitHub de forma intermite

Opción 2: La alternativa ideal para entornos mixtos (Continue.dev)

Si la opción nativa de Copilot se te queda corta o te bloquea el autocompletado cuando agotas tus tokens de GitHub, la solución más robusta y recomendada por la comunidad de software libre es la extensió n **Continue**.**Continue** es una extensión de código abierto para VS Code y JetBrains diseñada específicamen

te para hacer exactamente lo que pides: gestionar una operativa mixta real sin atarte a un solo proveed
or. Reemplaza la interfaz de Copilot pero se integra de forma idéntica en tu flujo de trab

Por qué es la solución perfecta para tu caso:

- Autocompletado y Chat independientes: Puedes configurar Continue para que use el Copilot oficial de GitHub (mientras te queden tokens) para el chat, pero que use tu servidor local para el autocompletad o continuo (que es lo que más tokens devora
- Transición transparente: En el momento en que Copilot te diga "basta", cambias el desplegable del modelo en la barra inferior de la extensión a tu servidor local y sigues programando sin perder el flujo
- Control absoluto del contexto: Al ser software libre, gestionas tú mismo cómo indexa el código localmente (mediante embeddings locales), respetando la privacidad de tu código

Configuración básica en `config.json` de Continue:

Solo tienes que definir tus dos mundos en su archivo de configuración. Un ejemplo de operativa mixta apuntando a tu servidor:JSO

CODE

```
{
"models": [
{
"title": "GitHub Copilot (Cloud)",
"provider": "github-copilot",
"model": "gpt-4o"
},
{
"title": "Mi Servidor Local",
"provider": "ollama",
"model": "deepseek-coder:6.7b",
"apiBase": "http://<IP_DE_TU_SERVIDOR>:11434"
}
],
"tabAutocompleteModel": {
"title": "Autocompletado Local",
"provider": "ollama",
"model": "deepseek-coder:1.3b",
"apiBase": "http://<IP_DE_TU_SERVIDOR>:11434"
}
}
```

Resumen de la estrategia recomendada

Si estás muy cómodo con la interfaz visual estricta de Copilot, actualiza y prueba a mapear tu endpoint en la configuración **BYOM** de la extensión oficial.Sin embargo, si lo que buscas es no volver a preocuparte nunc a más por los contadores de tokens a fin de mes, instalar **Continue** y configurar tu servidor como motor princip al de Tab-Autocomplete (dejando a Copilot solo para consultas complejas de arquitectura en el chat hasta que se agote) te dará la libertad total que estás buscando.¿Qué software de inferencia estás utilizando actualment e en tu servidor para servir el mo

Jordi H.

Has dicho ahora mismo estoy con vllm y gpt-oss. pero cambiaría el modelo a uno más especializado para codificación. y dime, puedo hacer un mix similar pero en vez de usar mi LLM local, usar las capacidades d e Gemini sin perder funcionalidades de Copilot? aún más, cómo comparas el uso de Google Antigravity con el i

de desplegado contra un entorno github copil

Gemini

Sí, puedes hacer exactamente ese mix. Sustituir tu servidor local por la API de Gemini te permite descargar tu hardware sin renunciar a la libertad de orquestar los modelos a tu gusto

Integrar Gemini en la operativa mixta

Instalar extensiones como Gemini Code Assist directamente junto a la de GitHub Copilot en VS Code suele generar una guerra por ver cuál de las dos controla el autocompletado en línea (el ghost text).La solución más limpia para tu flujo de trabajo en Python es usar **Continue** como único orquestador. Al igual que harías con tu modelo local, puedes configurar Continue para que actúe como puente hacia la API de Google AI Studio, manteniendo a Copilot configurado para otras tareas.

CODE

```
{
"models": [
{
"title": "Copilot Chat",
"provider": "github-copilot",
"model": "gpt-4o"
},
{
"title": "Gemini 3 Pro",
"provider": "gemini",
"apiKey": "TU_API_KEY",
"model": "gemini-3.0-pro"
}
],
"tabAutocompleteModel": {
"title": "Gemini Autocomplete",
"provider": "gemini",
"apiKey": "TU_API_KEY",
"model": "gemini-3.0-flash"
}
}
```

Google Antigravity vs. VS Code + Copilot

Comparar un entorno tradicional vitaminado con Copilot frente a **Google Antigravity** es comparar un asistente de vuelo con un piloto automático completo. Antigravity no es un simple plugin; es un IDE completo (un fork de VS Code) lanzado recientemente y diseñado desde cero para el paradigma agent-firs

Característica	VS Code + Copilot / Continue	Google Antigravity
Paradigma	Reactivo: Autocompleta y responde a	Autónomo: Le asignas una tarea y el
Superficie	Edición de texto y contexto de	Controla el editor, lanza comandos
Verificación	Tú revisas el código línea a línea.	Genera Artefactos (planes, logs,
Gestión	Un chat lateral.	Vista Manager para orquestar

La perspectiva arquitectónica y del software libre

Aquí es donde entra el espíritu crítico. Antigravity ofrece una potencia desmesurada al permitirte delegar tareas

complejas (como levantar un backend entero o refactorizar dependencias) a agentes que usan nativament e Gemini 3, Claude 4.5 o GPT-OSS.Sin embargo,**Antigravity es software propietario**. Aunque actualmente es gratuito en su fase preview, te exige ceder el control del entorno de desarrollo a un ecosistema corporat ivo cerrado.Si priorizas la soberanía de tu infraestructura y te gusta trastear con el sistema, la combinació n de **VSCodium + Continue** es insuperable. Te mantiene a los mandos. Puedes usar la API de Gemini hoy por pura comodidad, y mañana, si las políticas cambian o necesitas máxima privacidad, puedes volver a apun tar el sistema a los modelos de codificación servidos en Ymir con solo modificar el config.json. No hay e ncierro tecnológico ni dependencias de un IDE opaco.Para profundizar en estas o