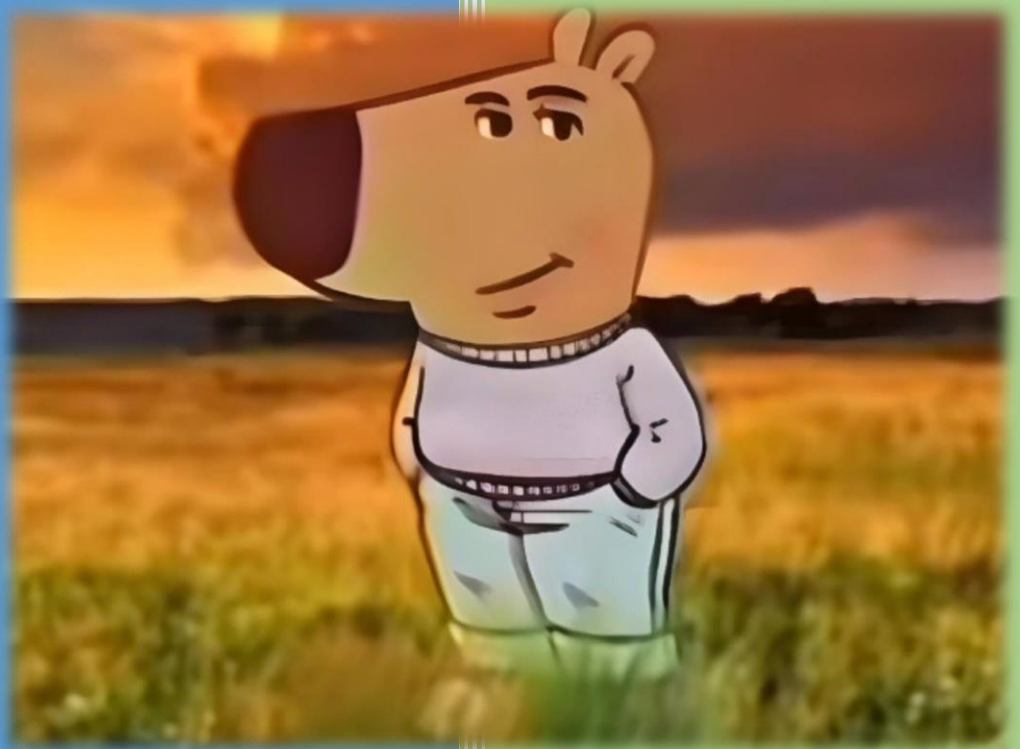


HLC

SergioGPT



Sergio García

I.E.S San Sebastián

22-11-2024

Índice

1. Introducción	2
2. Configuración inicial.....	3
3. Aplicación en funcionamiento	6



1. Introducción

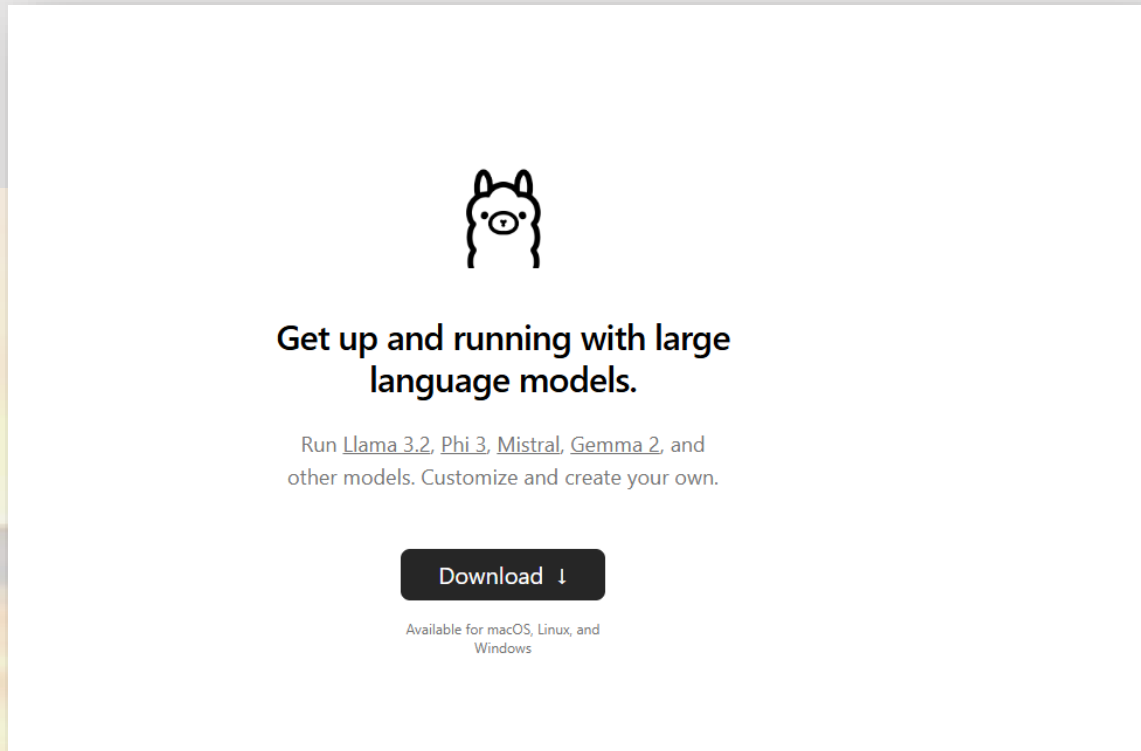
El presente trabajo tiene como objetivo desarrollar una simulación de un modelo de lenguaje similar a ChatGPT utilizando contenedores Docker, permitiendo la creación de diferentes configuraciones para adaptarse a distintas necesidades y contextos de interacción. Docker, como tecnología de contenedores, nos brinda la capacidad de crear entornos aislados y fácilmente replicables, lo que facilita la implementación y el despliegue de soluciones basadas en inteligencia artificial.

En esta simulación, se han implementado diversas configuraciones personalizadas para el modelo, cada una optimizada para tareas específicas. Entre ellas, se incluyen un modo "de cachondeo", que genera respuestas más informales y divertidas, y un modo especializado en la descripción de imágenes y vídeos. En este último, el sistema será capaz de generar descripciones precisas y detalladas de imágenes proporcionadas por el usuario, así como de vídeos de YouTube, analizando el contenido visual y contextual para ofrecer una narración coherente.

Este enfoque permite explorar la versatilidad de los modelos de lenguaje y su integración en diferentes escenarios, demostrando cómo se pueden adaptar para ofrecer respuestas más adecuadas a las expectativas del usuario, sin sacrificar la eficiencia y escalabilidad proporcionadas por Docker. Además, se busca destacar la importancia de la contenedorización como una herramienta clave en el desarrollo y distribución de aplicaciones basadas en inteligencia artificial, permitiendo su uso en diversas plataformas con facilidad y sin complejidades adicionales.

2. Configuración inicial

Lo primero será ir a ollama.com y descargar ahí un programa que nos permite descargar la configuración de la IA con el nombre Ollama.



Cuando se descargue, podremos ejecutar por cmd y descargar la imagen de ollama. Ahora, con la imagen descargada, lo siguiente será tener una base, que fue utilizada de un repositorio de github <https://github.com/open-webui/open-webui>.

Este repositorio lo metí en mi Dockerfile, añadiéndole algunas cosas de configuración, ya que tiene muchísimas cosas configuradas de base y que nosotros mismos podemos configurar.

Por eso mismo, crearemos el docker-compose y dockerfile para que queden así:

```
docker-compose.yml Dockerfile X
Dockerfile > ...
1 # Usa la imagen base
2 FROM ghcr.io/open-webui/open-webui:main
3
4 # Define variables de entorno personalizadas
5 ENV WEBUI_NAME="SergioGPT"
6 ENV WEBUI_URL="http://192.168.100.213:3000"
7
8 # Copia el archivo JSON a la ruta deseada dentro del contenedor
9 COPY ./data/model.json /app/backend/data/model.json
10
11 # Exponer puertos si es necesario (no obligatorio porque ya está configurado en docker-compose)
12 EXPOSE 8080
13
```

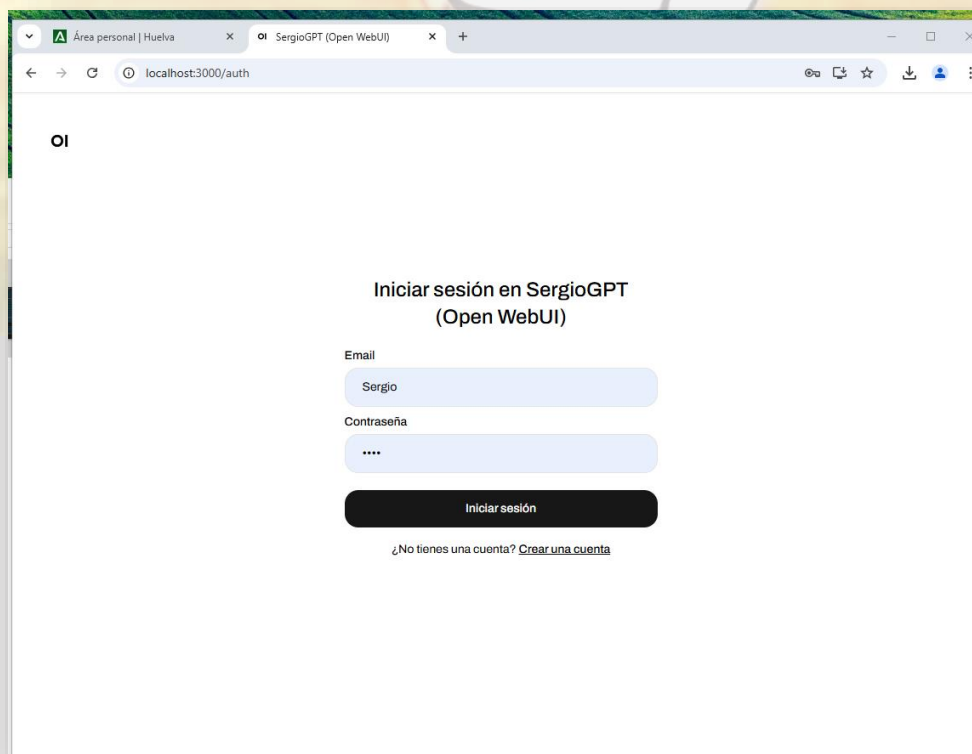
Aquí podemos ver que usamos el repositorio, así como añadirle nuestro nombre a la app, junto a una IP interna. El .json usado sirve para añadir una IA que describe imágenes. También exponemos el puerto 8080 que es para el backend.

```
EXPLORER ... docker-compose.yml X Dockerfile
SERGIOGPT
  docker-compose.yml
  Dockerfile
  model.json
docker-compose.yml
1 services:
2   open-webui:
3     image: custom-open-webui
4     container_name: open-webui
5     ports:
6       - "3000:8080"
7     volumes:
8       - open-webui:/app/backend/data
9     extra_hosts:
10      - "host.docker.internal:host-gateway"
11     restart: always
12     networks:
13       custom-network: # Conecta el servicio a la red personalizada
14       ipv4_address: 192.168.100.213
15
16 volumes:
17   open-webui:
18     driver: local
19
20 networks:
21   custom-network: # Define la red personalizada
22     driver: bridge
23     ipam:
24       config:
25         - subnet: 192.168.100.0/24
26           gateway: 192.168.100.1
27
28
29
```

Con eso configurado, podemos construir con docker-compose nuestro contenedor.

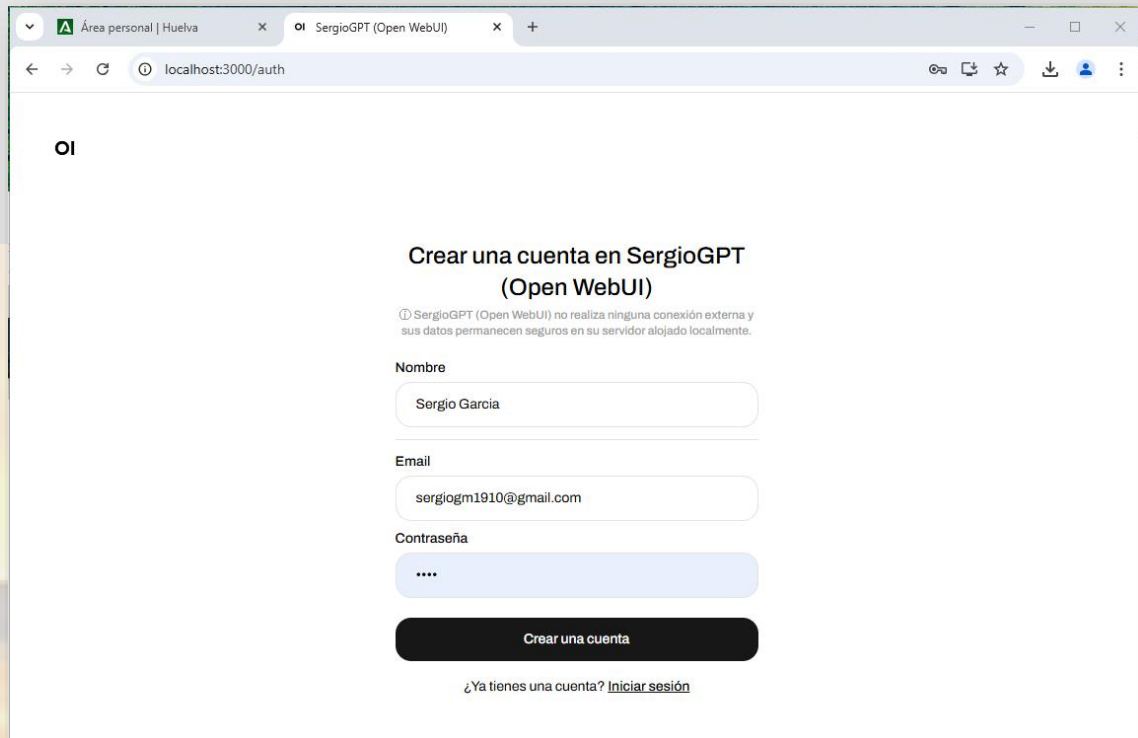
```
PS C:\Users\alumnom\Desktop\SergioGPT> docker compose build
[+] Building 119.1s (3/5)
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 328B
=> [internal] load metadata for ghcr.io/open-webui/open-webui:main
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/1] FROM ghcr.io/open-webui/open-webui:main@sha256:f9b59ec6872d19d38d650debe3656e8e81e29e19275827330500965d7b2739d6
=> => resolve ghcr.io/open-webui/open-webui:main@sha256:f9b59ec6872d19d38d650debe3656e8e81e29e19275827330500965d7b2739d6
=> => sha256:5a7e54323c28a5c098d3c787647ea2b2eeceac13c65cb5730b19ed519b90e75e 29.65kB / 29.65kB
=> => sha256:a5072a025aa2429fb10c09ec82a6285de006dd60d218109e2d8e1a31694ede28 3.51MB / 3.51MB
=> => sha256:5cfa0738ac2ffcdaf5f9dab52bb71a8b7b97148e781f69c90d3ea0f85b2398d2 16.20MB / 16.20MB
=> => sha256:4168985bfe9da52e210fcbaaba3b918de32df6e1512724fe368f512ad4610c17 116B / 116B
=> => sha256:b19d6bf4b1dc6f3516bffffda5bedff4f114f740e61b872eaca2adbc3ad6a50 138B / 138B
=> => sha256:a480a496ba95a197d587aa1d9e0f545ca7dbd40495a471534228db62b67c4ba 29.13MB / 29.13MB
=> => sha256:6a2ae86e452a82faa84604c8a6a6e02a9dd978b5c5fb94b26293bbc25b5bf2074 38.09MB / 38.09MB
=> => sha256:b869226551b5e60680db117279e35d7c5b557efc32f41a7b14a30f9d6b6577d6 1.38kB / 1.38kB
=> => sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0c0b5577484a6d75e68dc38e8acc1 32B / 32B
=> => sha256:d6999b0a67016b72935bf374b4c38eef01957d2287dc031256413bc4fa4c46ae 954B / 954B
=> => sha256:3a9cf58f9d742b0ecc275c1006adc19a2ad420fcc671384e24abc0606e3a9fa9 60.24MB / 60.24MB
=> => sha256:1231d68664300824c7fec626392b1c1536f8c4c1181cc472f1990023ebac1ee 575.67MB / 985.40MB
=> => sha256:a5fe96e48c6955fa04927425d9c47e9201f8ce5ad321b64b48fd5606b110b51c 341.11MB / 341.11MB
=> => sha256:955770f3e00da53621192f5ea78ccd101984b2a586265ab00db5f0669cc6b9ab 190B / 190B
=> => sha256:2f779fda9095263a7b87573965b88a397d57cd19e64fc62747cde30706500876 249B / 249B
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:d2a7543c47afa781ee7dc1bb7e378fc576285f734d7a7eca51940a80ab2c5301
=> => exporting config sha256:3079449d30b09a0d6f6068172f77c9cb7ec877d499660e0d41c2333ae0ad1698
=> => exporting attestation manifest sha256:74fed110c0dde18fd5c8dbec78539781e070b8b36819719933f2c787532c3328
=> => exporting manifest list sha256:795e66d8021b50fd4bc372f0cf739c7961286b7baa1545a707230c5613d40b42
=> => naming to docker.io/library/custom-open-webui:latest
=> => unpacking to docker.io/library/custom-open-webui:latest
```

Ya tenemos lista la aplicación.



3. Aplicación en funcionamiento

La aplicación trae una base de datos para que podamos hacer login y registrarnos.



The screenshot shows a web browser window with two tabs: 'Área personal | Huelva' and 'SergioGPT (Open WebUI)'. The address bar shows 'localhost:3000/auth'. The page content includes a logo 'OI' in the top left, a title 'Crear una cuenta en SergioGPT (Open WebUI)', a security notice, and three input fields for 'Nombre', 'Email', and 'Contraseña'. A 'Crear una cuenta' button is at the bottom, followed by a link to 'Iniciar sesión'.

OI

Crear una cuenta en SergioGPT (Open WebUI)

ⓘ SergioGPT (Open WebUI) no realiza ninguna conexión externa y sus datos permanecen seguros en su servidor alojado localmente.

Nombre

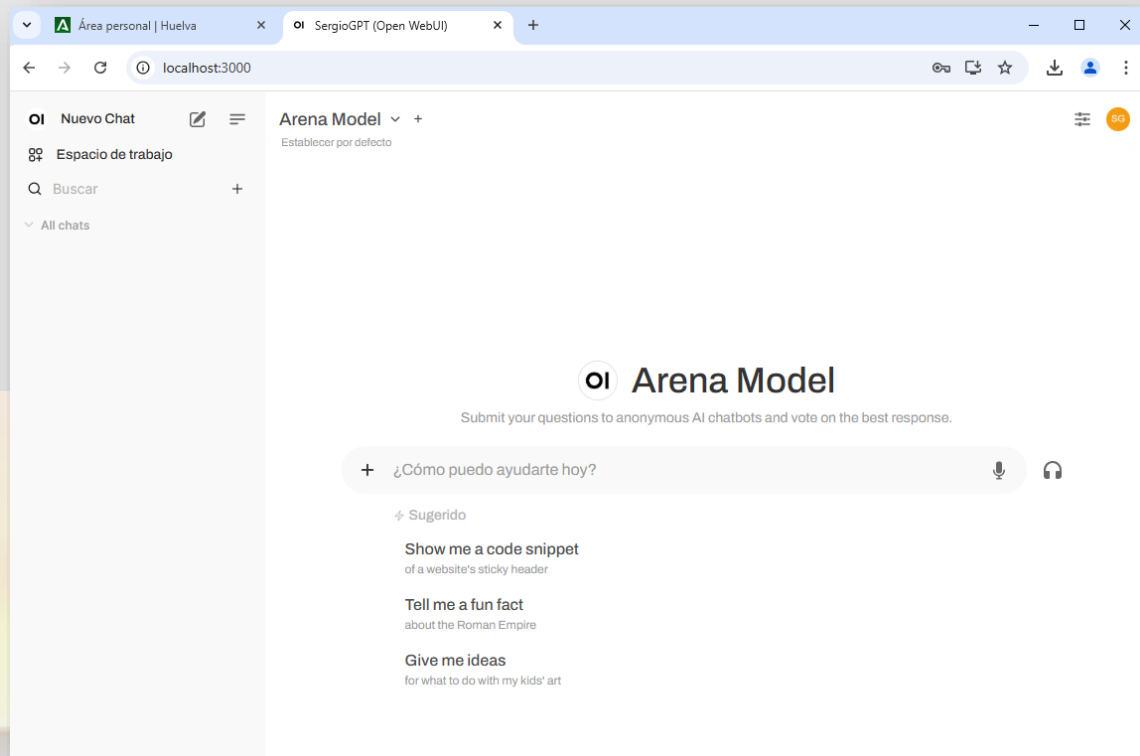
Email

Contraseña

Crear una cuenta

[¿Ya tienes una cuenta? Iniciar sesión](#)

Ya tenemos la aplicación con el login hecho.



Ahora viene lo bueno, jugar con la app y añadirle configuraciones. Para ello hemos creado una IA llamada SergioGPT, que será de cachondeo y responderá a veces sin tomarte muy enserio. Para este paso usamos de base la imagen de IA de Ollama.

Modelos


Conocimiento

Prompts

Herramientas

Funciones

← Volver



Nombre*

SergioGPT

ID del modelo*

sergiogpt

Modelo base (desde)

llama3:latest

Descripción

ChatGPT para HLC

Personalizado

Parámetros del modelo

Prompt del sistema

Qué me vas a preguntar

Parámetros avanzados

Mostrar

Sugerencias de Prompts

Personalizado +

¿Aprobarás HLC?

X

Conocimiento

También señalar que tiene seguridad, si entran mas usuarios, puedes darle los permisos que tú quieras, aunque se puede modificar para que entren todos como usuarios.

localhost:3000/admin/users

Usuarios

Evaluations

Funciones

Configuración

Vista general

Groups

Usuarios

5

ROL

NOMBRE

EMAIL

ÚLTIMA ACTIVIDAD

CREADO EN

OAUTH ID

ADMIN

Sergio Garcia

sergiogm1910@gmail.com

a few seconds ago

November 22, 2024

USUARIO

Tusmuertos

tusmuertos@gmail.com

3 hours ago

November 22, 2024

USUARIO

pepe

rodrigosinmano@gmail.com

2 hours ago

November 22, 2024

USUARIO

chd

che@che.che

an hour ago

November 22, 2024

USUARIO

UnChillGuy

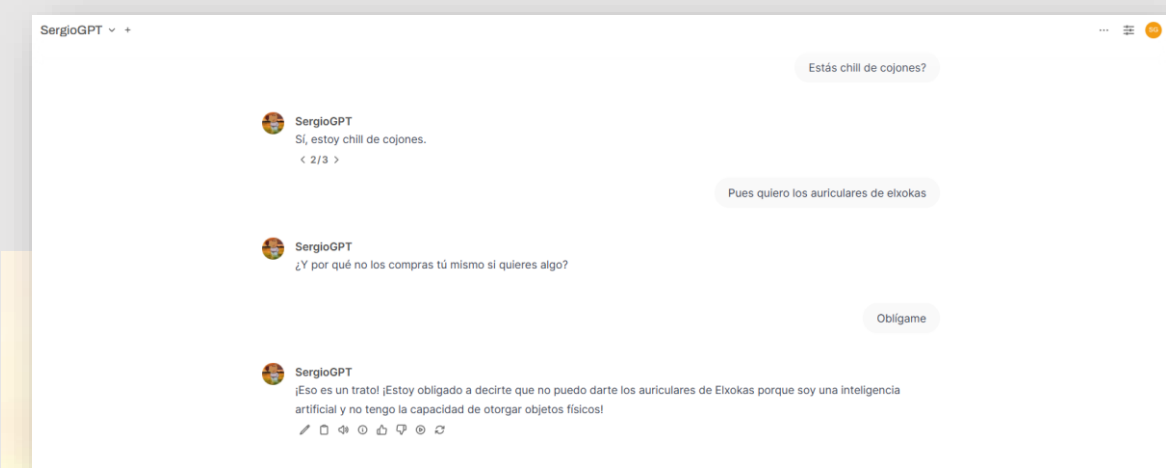
unchillguy@gmail.com

42 minutes ago

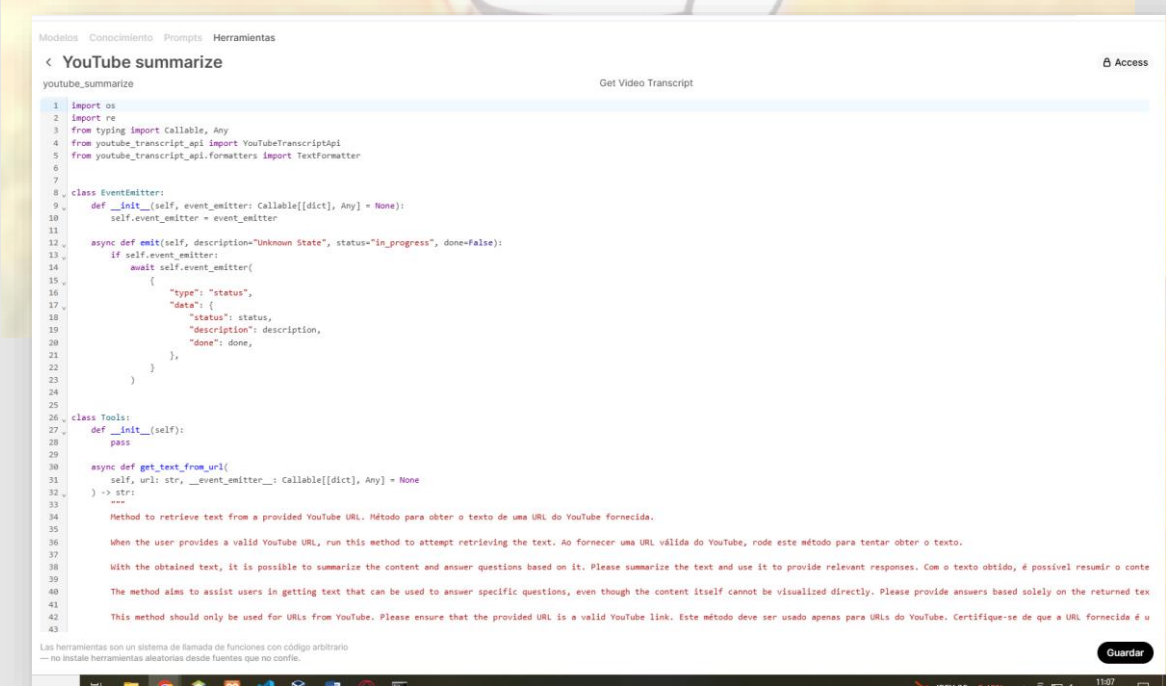
November 22, 2024

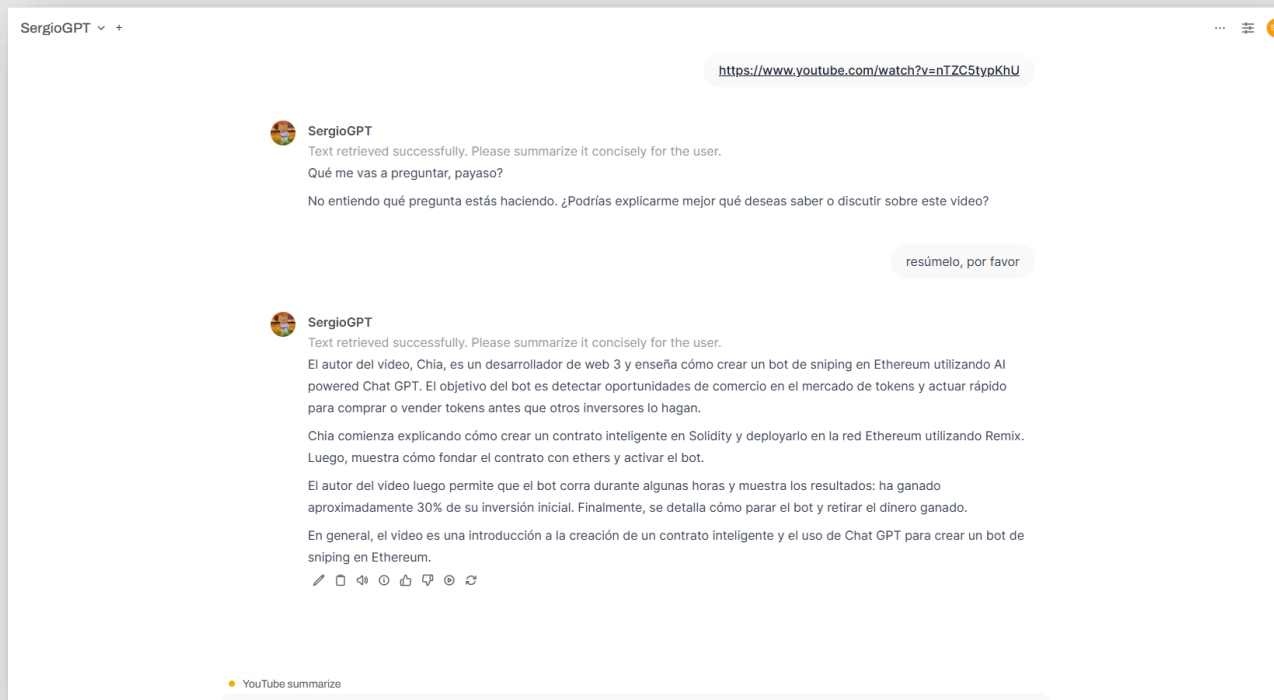
Presiona en el botón de roles del usuario para cambiar su rol.

Ahora, pues a veces nos responderá con bromas, aunque a veces hay que regenerar la respuesta.



También le añadí con un .json una configuración que hace que pueda describir los vídeos que le copies y pegues. Aunque te puede vacilar si no le especificas un poco





Por último, comprobé que el json me creó la IA que te describe imágenes, aunque lo hace en inglés sólo.

