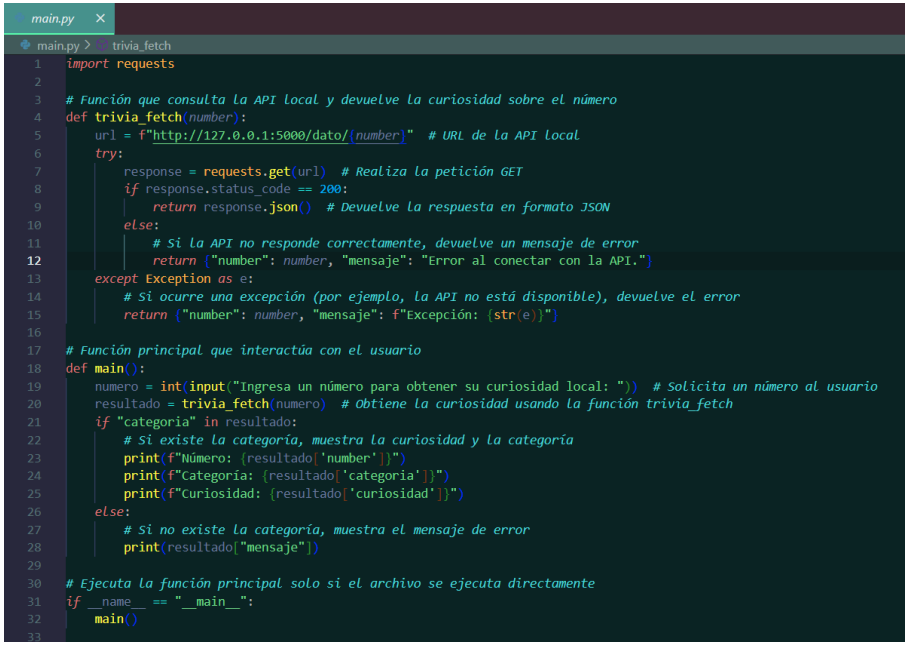
Trivia de Números - Proyecto Python

Main.py

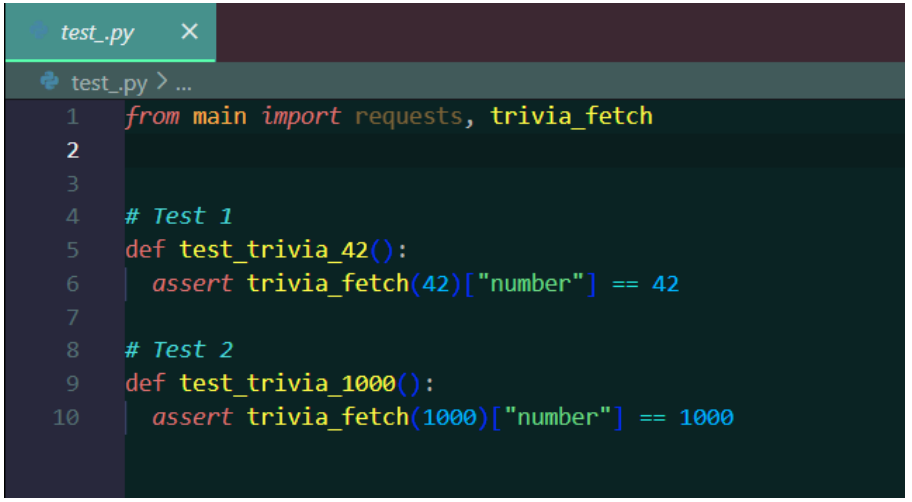
El archivo main.py es el programa principal que permite al usuario consultar curiosidades sobre un número.

Pregunta al usuario por un número.
Envía ese número a una API local usando la función trivia_fetch.
Recibe la respuesta de la API y muestra la curiosidad y la categoría si existen.
Si no hay curiosidad para ese número, muestra un mensaje de error.
En resumen: el usuario ingresa un número y el programa le muestra una curiosidad sobre ese número usando una API local.



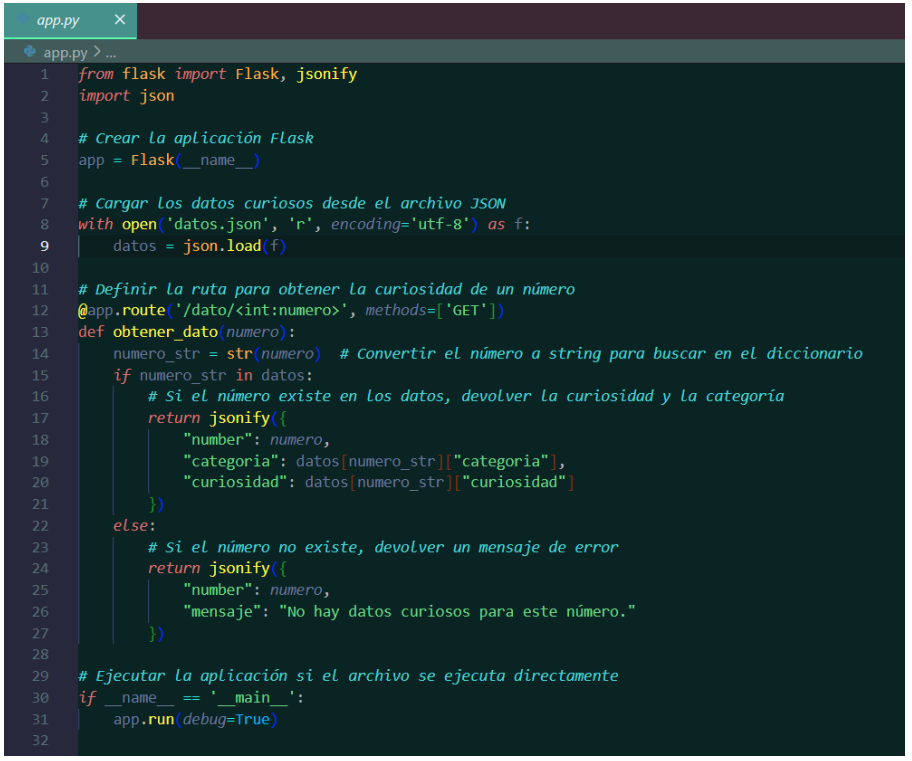
test_.py

El archivo test_.py contiene pruebas automáticas para tu proyecto. Verifica que la función trivia_fetch devuelve correctamente el número consultado en su respuesta.
Si el resultado es correcto para los números 42 y 1000, los tests pasan y tu función funciona bien.



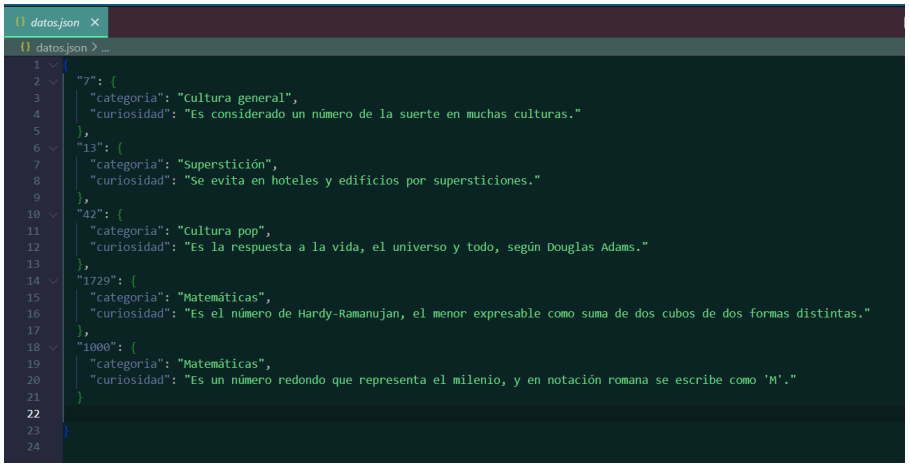
app.py

El archivo app.py crea una API usando Flask.
Lee curiosidades de números desde el archivo datos.json.
Cuando alguien visita la ruta /dato/<número>, la API responde con la curiosidad y la categoría de ese número si existe; si no, muestra un mensaje diciendo que no hay datos para ese número.



datos.json
El archivo datos.json contiene curiosidades sobre varios números.
Cada número es una clave y tiene dos datos asociados:

"categoria": el tipo de curiosidad (por ejemplo, "Cultura pop", "Matemáticas").
"curiosidad": una frase con información interesante sobre ese número.
Por ejemplo, para el número 42, se guarda que es famoso en la cultura pop por ser "la respuesta a la vida, el universo y todo".



Pasos realizados para el proyecto

1. Creación del archivo de datos (datos.json):

Se hizo un archivo con curiosidades y categorías para varios números.

2. Desarrollo de la API local (app.py):

Se creó una API con Flask que lee el archivo datos.json y responde con la curiosidad y categoría de un número cuando se consulta la ruta /dato/<número>.

3. Desarrollo del cliente (main.py):

Se hizo un programa que pide al usuario un número, consulta la API local y muestra la curiosidad y categoría si existen.

4. Implementación de la función trivia_fetch:

Esta función se encarga de hacer la petición a la API y devolver la información recibida.

5. Creación de pruebas automáticas (test_.py):

Se escribieron tests para verificar que la función trivia_fetch devuelve correctamente el número consultado.

6. Instalación de dependencias:

Se instalaron las librerías necesarias (flask y requests) usando pip.

7. Ejecución y prueba del sistema:

Se inició la API local, se ejecutó el cliente y se corrieron los tests para comprobar que todo funciona correctamente.

Uso de IA en el desarrollo

Se utilizó IA para investigar sobre APIs públicas de trivia de números, como numbersapi.com, y sobre cómo crear una API propia con Flask. La IA ayudó a identificar buenas prácticas de desarrollo, como el uso de funciones, separación de lógica interactiva y estructuración de pruebas automáticas.

Se empleó IA para generar ejemplos de código, sugerir mejoras y asegurar la compatibilidad con sistemas de evaluación automática.