

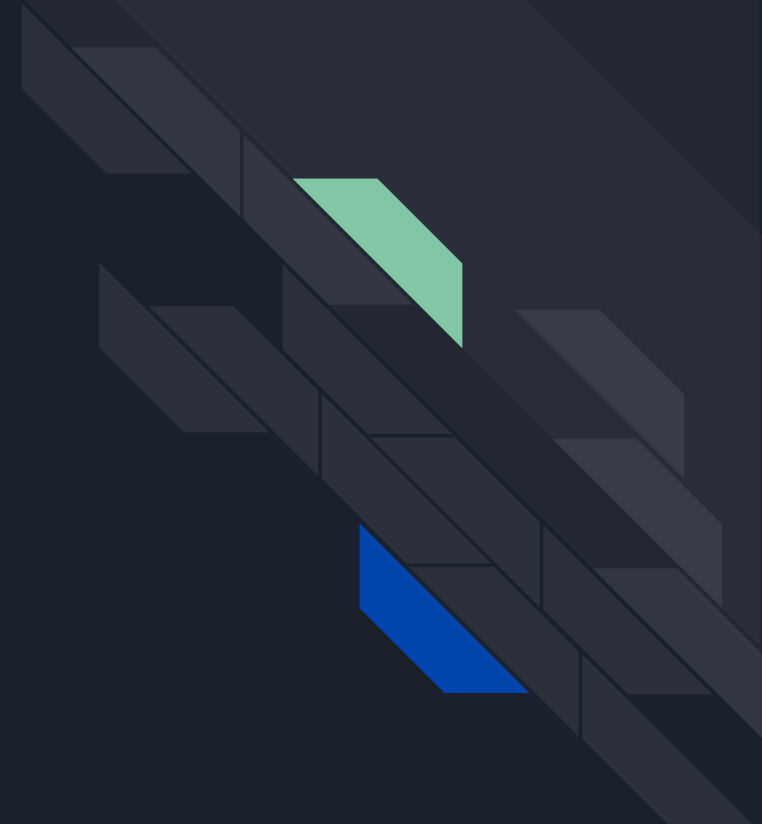


# GitHub Copilot

Ricardo Miralles Bernal

# Índice

- Introducción
- Historia y desarrollo
- ¿Cómo funciona?
- Ventajas y desventajas
- Principios legales y éticos

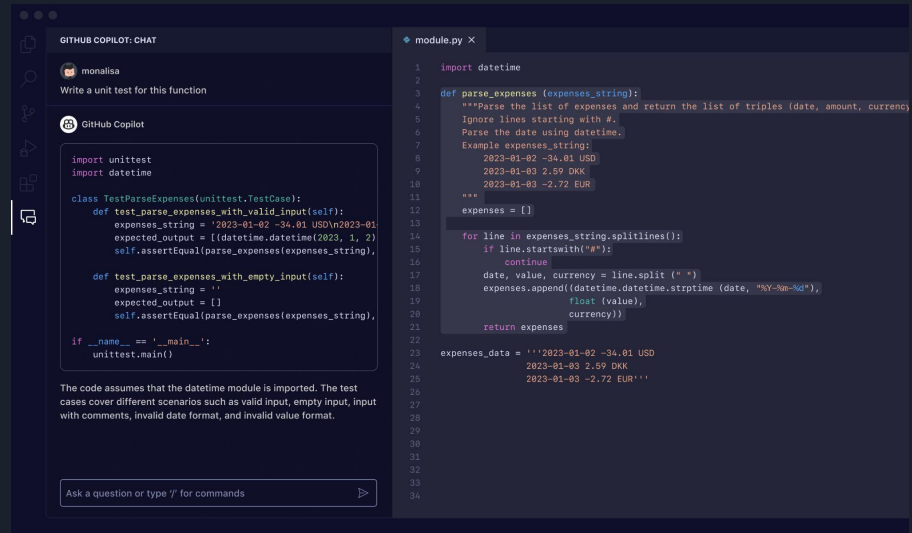


# Introducción

## ¿Qué es GitHub Copilot?

Es una herramienta de inteligencia artificial que actúa como asistente de programación.

Desarrollada por GitHub en colaboración con OpenAI, utiliza un modelo de lenguaje avanzado para ayudar a los desarrolladores a escribir código más rápido y con mayor eficacia. Su principal objetivo es facilitar el proceso de programación.



The screenshot displays the GitHub Copilot interface. On the left, a chat window titled 'GITHUB COPILOT: CHAT' shows a conversation with a user named 'monaliaa'. The user's prompt is 'Write a unit test for this function'. The Copilot's response is a Python unit test for a function named 'parse\_expenses'. The test includes a class 'TestParseExpenses' with two methods: 'test\_parse\_expenses\_with\_valid\_input' and 'test\_parse\_expenses\_with\_empty\_input'. The test cases cover valid input, empty input, and input with comments, invalid date format, and invalid value format. On the right, a code editor window titled 'module.py' shows the implementation of the 'parse\_expenses' function. The function takes a string of expenses and returns a list of tuples (date, amount, currency). It uses the 'datetime' module to parse the dates and the 'splitlines()' method to split the input string into lines. The function also handles comments and invalid date formats.

```
import unittest
import datetime

class TestParseExpenses(unittest.TestCase):
    def test_parse_expenses_with_valid_input(self):
        expenses_string = '2023-01-02 -34.01 USD\n2023-01-03 2.59 DKK\n2023-01-03 -2.72 EUR'
        self.assertEqual(parse_expenses(expenses_string), [(datetime.datetime(2023, 1, 2), -34.01, 'USD'), (datetime.datetime(2023, 1, 3), 2.59, 'DKK'), (datetime.datetime(2023, 1, 3), -2.72, 'EUR')])

    def test_parse_expenses_with_empty_input(self):
        expenses_string = ''
        self.assertEqual(parse_expenses(expenses_string), [])

if __name__ == '__main__':
    unittest.main()
```

The code assumes that the datetime module is imported. The test cases cover different scenarios such as valid input, empty input, input with comments, invalid date format, and invalid value format.

```
import datetime

def parse_expenses(expenses_string):
    """Parse the list of expenses and return the list of triples (date, amount, currency).
    Ignore lines starting with #.
    Parse the date using datetime.
    Example expenses_string:
    2023-01-02 -34.01 USD
    2023-01-03 2.59 DKK
    2023-01-03 -2.72 EUR
    """
    expenses = []

    for line in expenses_string.splitlines():
        if line.startswith("#"):
            continue
        date, value, currency = line.split(" ")
        expenses.append((datetime.datetime.strptime(date, "%Y-%m-%d"),
                        float(value),
                        currency))

    return expenses

expenses_data = '''2023-01-02 -34.01 USD
2023-01-03 2.59 DKK
2023-01-03 -2.72 EUR'''
```



# Historia y desarrollo

GitHub Copilot fue lanzado en junio de 2021, en un esfuerzo por mejorar la productividad de los desarrolladores.

La herramienta se basa en el modelo de lenguaje GPT-3 de OpenAI, que ha sido adaptado y entrenado específicamente con una gran cantidad de datos de código abierto, incluidos repositorios públicos de GitHub.





## ¿Cómo funciona?

GitHub Copilot utiliza un modelo de inteligencia artificial conocido como Codex, que es una versión especializada de GPT-3. Este modelo ha sido entrenado en una amplia variedad de lenguajes de programación y estilos de codificación, lo que le permite entender el contexto del código que se está escribiendo.

A medida que el desarrollador escribe, Copilot analiza el código y el contexto circundante, generando sugerencias relevantes en tiempo real.





# Ventajas y desventajas

## Ventajas

- Aumento de la productividad
- Mejora la uniformidad del código
- Acceso a buenas prácticas
- Adaptación al estilo del Usuario

## Desventajas

- Errores y Sugerencias Incorrectas
- Requiere una suscripción su acceso
- Dependencia del Usuario

# Principios legales y éticos

- Derechos de Autor: GitHub Copilot se basa en un vasto conjunto de datos de código abierto, lo que plantea preguntas sobre la propiedad intelectual.
- Responsabilidad del Usuario: Aunque Copilot ofrece sugerencias útiles, la responsabilidad final del código recae en el desarrollador.
- Transparencia: Los desarrolladores deben entender cómo funciona GitHub Copilot y los datos en los que se basa.





Fin

