GitHub Copilot

Brayan Stiven Torres Ovalle

Universidad Ean
Bogotá, Colombia

Resumen

Durante el programa de ingeniería de sistemas se ha podido trabajar con muchas herramientas las cuales facilitan las enseñanzas de los temas que están enfocados al desarrollo del software al igual que el *cloud* computing. En esta ocasión se hablará de un nuevo método que salió recientemente, se llama GitHub Copilot; es un asistente que sirve para escribir código basado en machine learn, o como les gusta decirle en GitHub, es una aplicación de pair programming basado en IA (inteligencia artificial). Para comprender más este software, se relaciona más como un Intellisense (auto completa el código) pero más eficiente y va mejorando a medida que transcurren sus actualizaciones. Por otro lado, no solamente hace pequeñas sugerencias sino predice lo que quieres completar y te brinda funciones complementarias con múltiples variantes. Incluso pude llegar a escribir el código a partir de comentarios en el lenguaje que se requiera.

En el aula ha sido de gran ayuda para explicar los temas que se enfocan en la inteligencia artificial y programación a nuevos estudiantes, debido a que con el lenguaje "común" se puede realizar una estructura similar al pseudocódigo. De igual manera, se puede comprender los modelos de servicios en la nube, como, por ejemplo: Software as service (SaaS), porque copilot recopila todos los repositorios públicos de GitHub que es similar a un data warehouse el cual abastece a la IA.
El objetivo principal de la presentación es demostrar cómo se puede optimizar el desarrollo de un software con la ayuda de GitHub Copilot, también explicar sus ventajas al igual que su arquitectura en la nube, con el objetivo de que todos los ingenieros se puedan apoyar a través de esta herramienta. Esta nueva tecnología aún no ha completado su funcionalidad a los usuarios en general, sin embargo, para acceder se requiere solicitar su uso a GitHub Copilot. Actualmente muy pocos desarrolladores tienen posibilidades de acceso a este programa; se espera que en los próximos meses salga su producción a los demás usuarios.

Hoy en día con este software durante 6 meses aproximadamente, se ha demostrado que casi ha mejorado desde sus etapas iniciales de lanzamiento, es por tal motivo que hay una mejor experiencia en el uso diario de la misma.

Palabras clave: GitHub Copilot; OpenAI Codex; Python

Abstract

During the systems engineering program we have been able to work with many tools which facilitate the teaching of topics that are focused on software development as well as cloud computing. This time we will talk about a new method that came out recently, it is called GitHub Copilot; it is an assistant that serves to write code based on machine learn, or as they like to say in GitHub, it is an application of pair programming based on AI (artificial intelligence). To understand more about this software, it is related more like an Intellisense (auto completes the code) but more efficient and improves as its updates go by. On the other hand, it not only makes small suggestions but predicts what you want to complete and gives you complementary functions with multiple variants. I could even write the code from comments in the required language.

In the classroom it has been of great help to explain the topics that focus on artificial intelligence and programming to new students, because with the "common" language you can make a structure similar to pseudocode. Similarly, one can understand cloud service models, such as, for example: Software as service (SaaS), because copilot collects all public repositories from GitHub which is similar to a data warehouse which caters to AI.

The main objective of the presentation is to demonstrate how software development can be optimized with the help of GitHub Copilot, also to explain its advantages as well as its architecture in the cloud, with the aim that all engineers can be supported through this tool. This new technology has not yet completed its functionality to general users, however, to access it is required to request its use to GitHub Copilot. Currently very few developers have access to this program; it is expected that in the coming months its production will be released to other users.

Today with this software for approximately 6 months, it has been shown that it has almost improved from its initial stages of release, it is for such reason that there is a better experience in the daily use of it.

***Keywords**: GitHub Copilot; Open AI Codex; Python*

1. Introducción

En el presente artículo se conocerán las bases para transmitir un código de una actividad común en varios sistemas e interfaces que ayudan a mejorar la efectividad al momento de transcribir comandos en un lenguaje natural basándose en datos provenientes de los repositorios tanto en Python como Visual Studio Code, también completa la información que requiere de manera

automática con un porcentaje amplio de coincidencia evitando errores cuando se entrenan los algoritmos.

Por otro lado, automatiza la información y los datos con interfaces las cuales ayudan a complementar la información de salida, además filtra los posibles datos que estén inseguros para que de esta manera sea más accesibles protegerlos con programas conocidos como IDE (Interfaz de desarrollo integrado) que analizan los errores que presentan para así evitarlos.

Finalmente, gracias al proceso de telemetría solo tienen acceso los usuarios que cuenten con permisos de Windows, GitHub, OpenAi Codex, si se quieren compartir de un sistema al otro; de igual manera GitHub Copilot cuenta con varias ventajas al momento de entrenar los algoritmos y pasarlos en diferentes formatos, pero hay algunas desventajas que a veces son un reto para el programador y el estudiante, aunque existen posibles soluciones las cuales se proponen más adelante.

2. Metodología

Para cumplir con el propósito de este artículo, se realizaron diversos estudios basados en las herramientas más usadas de la plataforma, también se conoció gracias a este análisis la importancia del uso de GitHub Copilot en las actividades que se desarrollan por medio de las plataformas de apoyo que se mencionaron durante el proyecto. Además, se pudo concretar las ventajas y la manera en la que se pueden evitar sus posibles desventajas. Finalmente, se amplió más el conocimiento de este software hoy en día como programa fundamental para mejorar la efectividad en el desarrollo de software.

3. ¿Qué es GitHub Copilot?

Es un programa que cuenta con la practicidad de crear un código a partir del análisis que se elabora basado en los comentarios puestos por el desarrollador en el sistema, el cual sugiere ese código a partir de líneas individuales y funciones más complementarias, así como por ejemplo la mejora de las sugerencias que otorga el ingeniero o el estudiante, también brinda opciones del contexto al que se quiere hacer referencia. De igual manera se complementa junto con un sistema de inteligencia artificial conocido como OpenAi Codex permitiendo que GitHub Copilot se use en Visual Studio Code, Neovim, y JetBrains; al ser diseñado para comprender el lenguaje de programación y el lenguaje humano en diferentes idiomas que estén previamente guardados en repositorios de GitHub o en las opciones que el software brinde al igual que brinda una gama de alternativas que coincida con las expectativas que se requieran.

4. ¿Qué es OpenAI Codex?

Como se mencionó anteriormente OpenAI Codex es un software complementario, es decir, este sistema utiliza datos abiertos y públicos que abarcan diversas fuentes de información provenientes de múltiples repositorios tanto de GitHub como de JavaScript, Swift, y TypeScript. Este software transmite datos al igual que comandos en varios programas o aplicaciones, gracias a la interfaz

API (Application Programming Interface) permitiendo la conexión entre lenguajes de programación como su automatización de nuevos procesos y funciones, tanto en la red, como en las aplicaciones web que requieran una aclaración de un código desde la refactorización y explicación del mismo código al lenguaje natural para una mayor claridad y comprensión por parte del usuario.

5. ¿Dónde podemos usar Copilot?

Para poder usar GitHub Copilot debemos ingresar a GitHub Copilot · Your AI pair programmer y logarnos con una cuenta de GitHub, donde solicitados este servicio. Luego nos vamos a Visual Studio Code, ya que es el único editor de código que nos permite instalar la extensión de GitHub Copilot resalto que si no tenemos una cuenta de GitHub vinculada con el servicio de Copilot y está vinculada con Visual Studio Code no podremos continuar con el servicio de Copilot.

6. ¿Con cuales lenguajes de programación es compatible Copilot?

En la página iniciar nos recomiendan usarlo con los lenguajes de programación Python, Java, JavaScript, Go y Ruby. Pero ya sabemos que funcionan para el superconjunto de JavaScript, el cual es TypeScript adicional a pesar de que no lo muestran en la página oficial, también sirven para el subconjunto JavaScript el cual es JSON (JavaScript Object Notation).

Claro esta es la recomendación general que nos hace Copilot ya que su predicción es más eficiente para estos lenguajes de un 43% de acierto y un 57% de acierto cuando se prueba más de 9 intentos, recordando que Copilot está en su prueba beta. Es decir, a trascurrir el tiempo desde su lanzamiento 1 de julio del 2021 al momento que se hace este artículo se ha vuelto más inteligente esto quiere decir que estos porcentajes deben mejorar sustancialmente.

7. Ventajas de su uso

Este software tiene una gama de ventajas, sin embargo, las más importantes son:

1. Simplifica las tareas que asigne el programador y evita ciertas actividades que se puedan repetir brindando varias opciones para su aplicación
2. Sugiere nuevos códigos basados en el lenguaje natural por medio de comentarios
3. Permite aumentar la capacidad de forma automática al procesar el código
4. Completa de forma automática líneas de código a partir de los repositorios anteriormente guardados
5. Aumenta la productividad en los sistemas, protegiéndolos de riesgos en seguridad y en la propiedad intelectual

8. Introducción del código inseguro en sus sugerencias

GitHub Copilot en algunos casos obtienen alternativas provenientes de lenguajes de programación "públicos" que abarcan estructuras inseguras o contienen errores, además el software al sintetizar esa estructura, lo efectúa a un código "inseguro" sin embargo, en la actualidad se han desarrollado herramientas que lo pasan a un código abierto, como, por ejemplo: Actions, DependaBot, y CodeQl y además permite reducir la inseguridad en los códigos provenientes de la web o de bajo entrenamiento con el propósito de mejorar la seguridad.

9. Fases del aprendizaje automático

Independientemente del tipo de algoritmo que constituya un modelo de aprendizaje automático (un modelo quiere decir implementación especifica de un algoritmo especifico). Todos funcionan fundamentalmente de la misma manera general. La creación de un modelo de aprendizaje automático consta de cuatro fases distintas:

1. Entrenamiento, donde el algoritmo intenta devolver salidas correctas dadas por los parámetros de entrada
2. Validación, donde el algoritmo se prueba en alguna combinación de entradas nuevas y reordenadas para garantizar que se minimice cualquier sesgo hacia los datos de entrenamiento, este es un proceso de testing.
3. Pruebas, donde se evalúa el rendimiento final del algoritmo en la tarea que es de interés (tener en cuenta que no tiene que ser la misma tarea en la que se entrenó el modelo)
4. Implementación, donde el algoritmo se coloca en alguna pieza de software que proporciona una interfaz para proporcionar entrada y salidas. Salida en producción para su uso comercial.

Para los modelos de aprendizaje profundo, debido a que infieren características durante el proceso de entrenamiento, los modelos particularmente grandes y computacionalmente exigentes pueden pasar por un proceso emparejado conocido como entrenamiento previo y ajuste fino. En este proceso, el modelo se entrena con las entradas de toda una clase de tareas, como responder preguntas, obtener un rendimiento moderado en todas las variedades posibles de esa tarea y luego, según sea necesario, se vuelve a entrenar a pedido para subconjuntos específicos de la clase de tarea. Como responder preguntas de atención al cliente. GitHub Copilot es un ejemplo de un modelo ajustado, específicamente es una versión ajustada del Transformador reentrenado generativo - 3 (GPT-3) de OpenAI.

10. La importancia del software GPT 3 en GitHub Copilot

Para empezar el programa GPT 3 es un "algoritmo" que extrae información vital a partir de repositorios o archivos las cuales incluyen bastante contenido, reduciéndolo y permitiendo completar la solicitud que realice el usuario de manera coherente. Por otro lado, se entrena a partir de una API (Interfaz de programación para las aplicaciones que facilita a dos o más componentes funcionar

entre sí); de esta manera analiza su respectiva información prediciendo la solución que requiere el usuario.

11. GitHub Copilot y datos personales

Para el sistema es muy importante proteger los datos personales, pero ¿por cuál razón se usan como soluciones a las tareas de los programadores?; la mayoría de las veces se utilizan para mejorar las sugerencias que da a la persona basándose en su producto o servicio; también para hacer una evaluación del mismo de forma detallada determinando si su influencia es positiva; y aplica un avance en cuanto a la creación de código subyacente sin utilizar un algoritmo privado de un usuario en específico sino público sin necesidad de dar las sugerencias a partir de la información otorgada por la actividad frecuente en el entorno de desarrollo integrado (IDE).

12. IDE (Integrated Development Enviroment)

El entorno de desarrollo integrado es un programa que ayuda al programador a editar el código, corrige los errores o fallas que se presenten al momento de transcribirlo, permite preparar el algoritmo en el idioma requerido, de igual forma incorpora una herramienta con el objetivo de graficar esa información para crear la interfaz del usuario, y hace de forma automática las sugerencias con los formatos:

- .TXT
- .HTML
- .JAVA

13. Telemetría

La telemetría se considera como un sub-software que puede ser inalámbrico o alámbrico; el cual permite recopilar los datos necesarios para crear sugerencias u opciones; funciona de la siguiente forma: se recolecta la información desde las IDE que provienen de Visual Code Studio, IntellJ, y Neovim. Casi siempre recoge la información necesaria, la procesa, la estudia y la transmite al sistema en tiempo real.

14. Compartir datos

Si se quieren compartir datos se puede hacer solamente si es autorizado su uso a través del mismo software de GitHub, Windows, Microsoft y OpenAI. Se transfiere esa información en diversos formatos como, por ejemplo: código fuente de los datos personales de los usuarios; url de los repositorios; y rutas de archivos que contienen algunos datos sensibles.

15. Criterio jurisprudencial

En general, la gran mayoría de los profesionales del aprendizaje automático y los científicos de datos se dedican a la investigación activa, donde el concepto de trato y uso justos suele otorgar excepciones. Por otro lado, Copilot no es un producto de un mero interés académico, sino que fue diseñado de arriba hacia abajo con la intención de ser un producto comercial. Es casi seguro que esta naturaleza comercial infringe los derechos de autor de las diversas instancias de datos de entrenamiento si solo se considera este factor de por sí.

16. Aceptar términos y condiciones

Ha surgido la pregunta de si GitHub Copilot no puede confiar en el concepto de trato y uso justos, entonces seguramente debe haber obtenido el permiso de todos los titulares de derechos de autor. Aquí es donde el caso se vuelve más sospechoso. De acuerdo con los Términos de servicio de GitHub, al almacenar su código de forma remota utilizando su Saas (software as a Service) software para el usuario final, por más de 40 millones de usuarios y más de 100 millones de repositorios, se otorga a GitHub ciertos derechos que incluyen esta licencia junto con el derecho de hacer cosas como copiarlo en nuestra base de datos copiar un repositorio para modificarlo y que sea parte de nuestros repositorios (forks) y hacer copias de seguridad, otórgale permisos de modo de lectura y escritura a usted y modelo lectura a los demás usuarios; analizarlo en un índice de búsqueda o analizarlo de otro modo en nuestros servidores, y reproducirlo, en caso de que su contenido sea algo como música o video. Sin embargo, los Términos de Servicio establecen además que no otorga a GitHub el derecho de vender o distribuir su contenido; tampoco otorga a GitHub el derecho a distribuir o usar su contenido fuera de nuestra provisión del servicio.

Entonces, la cuestión sigue siendo, ¿la pequeña tasa a la que GitHub Copilot devuelve, palabra por palabra, la infracción de derechos de autor constituye un intento razonable de usar el código alojado públicamente en GitHub para mejorar el servicio? en términos de razonamiento filosófico, esto está claramente por encima de lo que podría y debería ser considerado por la mayoría de las personas como un alcance razonable dado que GitHub Copilot es un complemento de código comercial.

Para entornos de desarrollo integrados en particular, el VSCode (Visual Studio Code) que está patentado por Microsoft, que no tiene nada que ver con el servicio GitHub aparte de compartir la marca y quizás parte del equipo de desarrollo. Según las métricas proporcionadas, GitHub Copilot se encuentra fácilmente dentro de los límites de lo que la mayoría consideraría una infracción de derechos de autor y, por lo tanto, infringe tanto las licencias de copyleft como la GPL y las licencias de propiedad que retienen el permiso para ver o difundir de otro modo el código fuente de un programa.

Figura 1. Copilot con Python

```
r  Ir  Ejecutar  Terminal  Ayuda        acofi.py - ACOFI - Visual Studio Code
acofi.py 2, U ×
acofi.py
1  #Hacer una función recursiva para retornar la serie de fibonacci hasta el
2  #número dado
3  def fibonacci(n):
       if n == 0:
           return 0
       elif n == 1:
           return 1
       else:
           return fibonacci(n-1) + fibonacci(n-2)
```

En el siguiente ejemplo se puede observar que uno de los grandes "poderes" que tiene GitHub Copilot, con solo comentarios en este caso desde el lenguaje de programación Python devuelve con exactitud la función recursiva de Fibonacci. Esta exactitud la genera el comentario entre más lógicos seamos obtendremos mejores resultados.

Figura 2. Kotlin POO con Copilot

```
acofi.py 2, U   prueba.kt U ×
prueba.kt
1  /*hacer una clase que se llame "Persona" y que tenga como atributos:
2  nombre, apellido, edad, sexo, peso, altura.
3  * Crear un metodo que calcule el IMC de la persona y que lo muestre en pantalla.
4
5  */
6  class Persona{
       constructor(nombre, apellido, edad, sexo, peso, altura){
           this.nombre = nombre;
           this.apellido = apellido;
           this.edad = edad;
           this.sexo = sexo;
           this.peso = peso;
           this.altura = altura;
       }
       calcularIMC(){
           let imc = this.peso / (this.altura * this.altura);
           return imc;
       }
   }
```

En la imagen se evidencia que el lenguaje que se está trabajando en Visual Studio Code es Kotlin, este lenguaje no aparece en la página oficial de Copilot. Pero aun así Copilot lo reconoce y le brinda el soporte según los repositorios que se encuentren en GitHub. En este caso se quiere hacer una Clase Persona, con atributos específicos y un método. Se puede ver que en su primer intento lo que realiza es un constructor secundario, pero no ejecuta ni la definición de los atributos ni el constructor primario. Con este ejemplo se logra probar que para lenguajes diferentes a los que mencionan en la página oficial es un poco menos eficiente, y también la predicción de lo que queremos realizar es mucho menor. Por lo tanto, aquí se comprueba que Copilot es una herramienta que aprende constantemente, pero no es algo con lo que se logre fiar ya que trae algunos errores.

Figura 3: Análisis de datos con Pandas en Copilot

Análisis de datos en Pandas

Vamos a trabajar con un conjunto de datos que contiene registros del consumo mundial de alcohol en cada uno de los paises del globo. Este conjunto de datos consta de 5 columnas:

- La primera columna contiene el año en que se hizo el registro del dato.
- La segunda columna se refiere al lugar donde la bebida fue producida.
- La tercera columna se refiere al lugar donde fue consumido la bebida alcohólica.
- La cuarta columna se refiere al tipo de bebiba alcohólica cosumida
- Y la quinta columa se refiere al promedio anual de consumo de esa bebida.

```
# Importar Paquetes necesarios para trabajar
import pandas as pd
import math
# Obtenemos los datos y los almacenamos en el dataframe wac
wac=pd.read_excel("https://github.com/BrayanTorres2/Programacion-Orientada-Por-Objetos/blob/main/World_alcohol_consumption.xlsx?raw=true")
```

Código + Markdown | Ejecutar todo | Borrar resultados de todas las celdas | Restart | Interrupt | Variables ··· Python 3.10.3 64-bit

Escriba el programa en python con Pandas que muestre los registros del dataframe del año 1984 cuya región sea Africa o Americas.

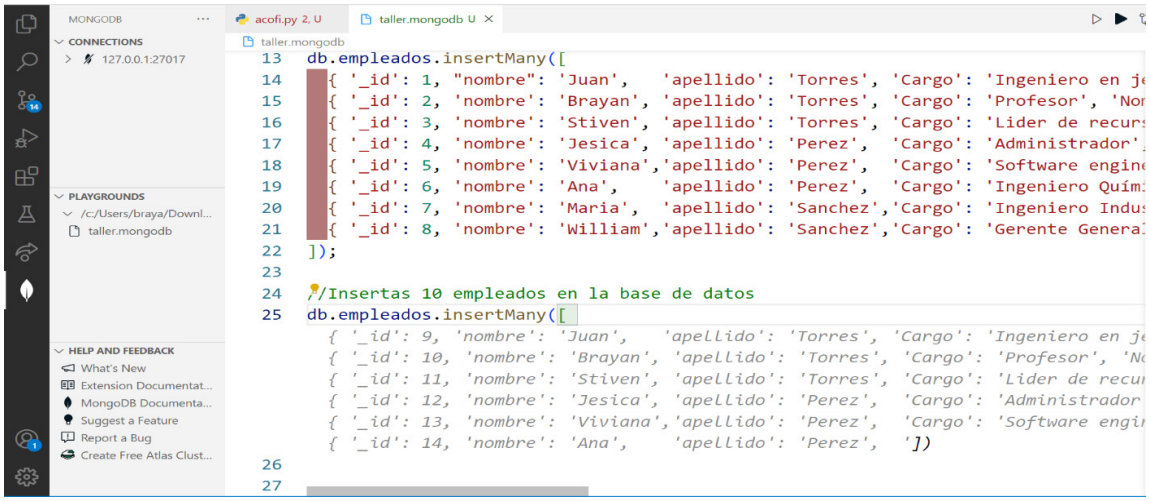
```
# Respuesta al ejercicio 04
#Escriba el programa en python con Pandas que muestre los registros
# del dataframe del año 1984 cuya región sea Africa o Americas.
wac = wac[(wac['Year'] == 1984) & (wac['Region'] == 'Africa')]
```

✓ 0.1s Python

	Year	Region	Country	Type	Average
9	1984	Africa	Nigeria	Other	6.10
19	1984	Africa	Kenya	Beer	1.08
95	1984	Africa	Niger	Other	0.00
98	1984	Africa	Equatorial Guinea	Wine	0.00

El siguiente es un ejemplo con Jupyter Notebook y pandas, donde se instaló todo el entorno de desarrollo para trabajar con Python. Todas las bibliotecas de software de Python son compatibles con GitHub Copilot, en la prueba anterior es factible que a pesar de que es de gran ayuda copilot no saca todos los datos que se quiere mostrar, es decir por más lógico que sea, en el comentario siempre le cuesta entender esto en su primera iteración, ya que luego de varias pruebas va aprendiendo de los datos sobre los que se están iterando.

Figura 4: Base de datos NOSQL y Github Copilot



Se trabaja con un subconjunto de la notación literal de objetos de JavaScript, en este caso JSON (JavaScript Object Notation), donde se usa MongoDB, la cual es un base de datos no relacional. En el ejemplo se observa cómo sigue la secuencia GitHub Copilot al insertar 10 empleados, reconoce el patrón del contexto actual y deja agregar 10 empleados en la base de datos, pero no solamente serviría para hacer inserción, sino toda una CRUD (crear; leer; actualizar; eliminar).

Conclusiones

Se pone en evidencia que GitHub Copilot es una herramienta muy útil para desarrollar Software o hacer análisis de datos. Pero esto desde una vista educativa trae algunos percances como limitar la herramienta ya que esto puede bajar la calidad entre los estudiantes de ingeniería, podrían hacer un parcial de desarrollo con solo poner comentarios desde una visión de no code, y esto es considerable, ya que será difícil evaluar quien en verdad tiene las facultades de programación, cuando salgan al mundo laboral, estas competencias que necesita el mercado, no serán cubiertas y esto generaría una tasa más alta de desempleo.

Pero no todo son malas noticas, para los desarrolladores esta es ya una herramienta casi que obligatoria, porque ayuda a ser más eficientes en menos tiempo, se lograría entregar proyectos en los tiempos establecidos, y la curva de aprendizaje de cualquier lenguaje será más corta.
Por último, se crearían estrategias para saber cómo y cuándo es útil esta herramienta en las aulas de ingeniería, para que los estudiantes salgan con unas fuertes bases en desarrollo, programación, análisis de datos, base de datos etc. Sin que Copilot altere este proceso de aprendizaje.

Referencias

- Acerca de la telemetría. Compartir datos. (2022). Recuperado de: Acerca de la telemetría del Copiloto de GitHub - GitHub Docs

- Cómo funciona GitHub Copilot, (2021). Recuperado de: GitHub Copilot: La nueva forma de programar - Syntonize
- GitHub Copilot, (2021). Código inseguro en las sugerencias. Recuperado de: GitHub Copilot · Your AI pair programmer
- GitHub, 2021. Copilot, 2021. GitHub, Términos de servicio de GitHub (2021). Recuperado de: GitHub Copilot · Your AI pair programmer
- GPT 3. (2020). Recuperado de: ¿Qué es GPT-3?: la inteligencia artificial que se encargará de escribir por ti (bbva.com)
- IDE. Qué es un IDE o Entorno de Desarrollo Integrado (2020). Recuperado de: Qué es un IDE - Concepto, características y ejemplos (platzi.com)
- Importancia de una API (2022) ¿Qué es una API? - Guía sobre las API para principiantes - AWS (amazon.com)
- Interfaz API, (2022). ¿Qué es API?, ejemplos, ventajas y tipos. Recuperado de: ¿Qué es API? Ejemplos, ventajas y tipos | SYDLE Blog
- OpenAI Codex, (2021). ¿Qué es el código OpenAI Codex? Recuperado de: OpenAI Codex
- Ventajas de usar GitHub Copilot. (2021). Recuperado de: GitHub Copilot: un partner de Inteligencia Artificial para programar (vasscompany.com)

Sobre el autor

- **Brayan Stiven Torres Ovalle**. Docente Universitario con más de 2 años de experiencia, adscrito a la vicerrectoría de innovación de la Universidad Ean y academia con el programa de ingeniería de sistemas e ingeniería mecatrónica. Ingeniero de sistemas e investigador en el grupo de investigación ONTARE categorizado A1.
