UNIVERSIDAD TÉCNICA DE BABAHOYO

FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA.

PROCESO DE TITULACIÓN

JUNIO 2023 – SEPTIEMBRE 2023

EXAMEN COMPLEXIVO DE GRADO O DE FIN DE CARRERA

PRUEBA PRÁCTICA

PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

INGENIERO EN SISTEMAS DE INFORMACIÓN

TEMA:

ANÁLISIS DE LA EXTENSIÓN GITHUB COPILOT, EN EL ÁMBITO DE COLABORACIÓN HACIA LOS DESARROLLADORES, PARA AUMENTAR, LA EFICACIA EN EL DESARROLLO DE PROYECTOS.

.

ESTUDIANTE:

KEVIN BALTAZAR LATA JACOME

TUTOR:

ING. BELTRAN MORA MAROLA NARCISA.

AÑO 2023

RESUMEN

Este proyecto se enfoca en llevar a cabo un análisis detallado de la extensión GitHub Copilot, centrándose en su aplicación en la colaboración entre desarrolladores. El objetivo principal es evaluar cómo GitHub Copilot puede mejorar la eficiencia en el desarrollo de proyectos de software. Se explorarán sus características clave, ventajas y posibles limitaciones.

Además, se realizarán pruebas para medir su impacto en la colaboración y el desarrollo de proyectos. Este análisis proporcionará una comprensión profunda de cómo GitHub Copilot puede beneficiar a los equipos de desarrollo y su papel en la evolución de la programación.

Palabras clave: GitHub Copilot, colaboración, desarrolladores, eficiencia, proyectos de software, análisis, extensión, programación

ABSTRACT

This project focuses on conducting a detailed analysis of the GitHub Copilot extension, with a focus on its application in developer collaboration. The main objective is to evaluate how GitHub Copilot can enhance efficiency in software project development. Key features, advantages, and potential limitations of GitHub Copilot will be explored. Additionally, tests will be conducted to measure its impact on collaboration and project development. This analysis will provide a deep understanding of how GitHub Copilot can benefit development teams and its role in the evolution of programming.

Keywords: GitHub Copilot, collaboration, developers, efficiency, software projects, analysis, extension, programming

CONTENIDO

PLANTEAMIENTO DEL PROBLEMA

La herramienta tecnológica que contiene una inteligencia artificial que se basa en el procesamiento de datos y la toma de decisiones automatizada, ahora esta tecnología la vemos implementada en nuestro día a día, desde ya hace diversos años atrás y sin duda transformará el futuro de nuestra generación. Dentro del ámbito de la programación, se encuentran presentes inteligencias artificiales que ejercen la función de desarrolladores, que son profesionales altamente capacitados para desempeñar un papel clave para desarrollo de software y sistemas informáticos capaces de efectuar labores que singularmente requieren la inteligencia humana.

Los desarrolladores de software enfrentan una serie de obstáculos significativos en su labor, especialmente en el contexto del desarrollo de aplicaciones web, uno de los desafíos más destacados es la presencia recurrente de errores al momento de compilar los proyectos. Sabemos que el entorno de desarrollo web está en constante evolución, con una amplia gama de lenguajes de programación, frameworks, bibliotecas y herramientas disponibles que les tocaría aprender y perfeccionarlas para dedicarse al ámbito de desarrollo de software. Después de invertir considerable tiempo en el perfeccionamiento de un proyecto, estos se encuentran con una larga lista de errores al momento de realizar pruebas, esta situación representa una pérdida de tiempo al momento de entregar sus proyectos, cosa que cuando haya que resolverlos se torna trabajoso y repetitivo.

Otro obstáculo es la complejidad inherente a la escritura de extensas líneas de código necesarias para desarrollar aplicaciones web. Los programadores se encuentran ante el desafío de grandes volúmenes de código, esta carga puede ser pesada, los desarrolladores de aplicaciones web buscan mejorar y aumentar los servicios que ofrecen, tienen que identificar y solucionar errores en una aplicación puede ser un proceso complejo. Las pruebas exhaustivas y la

depuración son esenciales para garantizar que la aplicación funcione correctamente y se entregue sin problemas. Después de que una aplicación web es puesta en marcha, es necesario realizar un mantenimiento continuo para corregir errores, actualizar tecnologías y agregar nuevas características. Esto puede requerir una inversión constante de tiempo y recursos.

Además, los desarrolladores de aplicaciones web deben buscar constantemente formas de mejorar sus servicios y habilidades. Evaluar sus capacidades en términos de conocimiento de lenguajes de programación, habilidades en la creación de bases de datos y otros aspectos relevantes se convierte en una necesidad, ya que enfrentan una serie de desafíos que pueden influir en la eficiencia y efectividad de su trabajo.

JUSTIFICACIÓN

En general, la extensión de GitHub Copilot facilita el proceso de desarrollo de software, además permitirá aumentar la eficiencia y ahorrar tiempo a los desarrolladores, ya ofrece sugerencias relevantes y útiles en tiempo real, la combinación de inteligencia artificial y desarrollo de software promete un futuro emocionante y lleno de posibilidades para la industria tecnológica.

Puede impactar en la colaboración entre desarrolladores si bien GitHub Copilot ofrece el potencial de aumentar la velocidad y la precisión en la creación de código, es fundamental evaluar cómo esta herramienta puede facilitar la interacción y la comunicación entre los miembros del equipo. Aún más en vista de los retos recurrentes que los desarrolladores enfrentan en el proceso de creación de aplicaciones web, como la presencia de errores frecuentes durante las etapas de prueba y lanzamiento, la complejidad de producir grandes volúmenes de código y la necesidad constante de mantenerse actualizados con las tecnologías emergentes.

La capacidad de GitHub Copilot para abordar estos desafíos y mejorar la colaboración entre los equipos de desarrollo podría tener un impacto significativo en la eficacia general del proceso, en la calidad de las aplicaciones web resultantes, la implementación de esta tecnología es una inteligencia artificial que al momento del desarrollador web escribir una letra o línea de código, genera líneas de código de sugerencia, líneas individuales y funciones completas al instante.

En el ámbito de colaboración hacia los desarrolladores en el contexto del desarrollo de aplicaciones web no solo aborda una necesidad actual en la industria, sino que también tiene el potencial de generar conocimiento valioso que beneficiará a los profesionales involucrados en la creación de aplicaciones web y aportará a la evolución continua de las prácticas de desarrollo.

OBJETIVOS

Objetivo general.

Analizar la extensión GitHub Copilot, en el ámbito de colaboración hacia los desarrolladores.

Objetivos específicos.

- Identificar las funcionalidades clave tales como sus ventajas, limitaciones y potenciales de la extensión GitHub Copilot.
- Evaluar el impacto de la extensión GitHub Copilot en colaboración con los desarrolladores de aplicaciones web.
- Ejecutar pruebas de concepto que simulen situaciones de colaboración y desarrollo usando GitHub Copilot, con el propósito de medir su impacto en la eficacia.

LINEAS DE INVESTIGACIÓN

El presente caso de estudio estructura su investigación siguiendo la orientación de la línea de investigación **“**Sistemas de información y comunicación, emprendimiento e innovación.” y en la Sub línea de investigación “Redes y tecnologías inteligentes de software y hardware”. En la cual la inteligencia artificial en la ingeniería de sistemas está íntimamente vinculada a las últimas tecnologías de desarrollo, con el propósito de simplificar la ejecución de proyectos de programación. Es relevante señalar que la aplicación que cuenta con inteligencia artificial en este campo permite generar soluciones inteligentes que tienen el potencial de mejorar la eficacia y el proceso de toma de decisiones.

MARCO CONCEPTUAL

Inteligencia e Inteligencia artificial.

La inteligencia representa una aptitud mental de carácter bastante abarcador, englobando la destreza en el razonamiento, la planificación, la resolución de problemas, el pensamiento abstracto, la asimilación de conceptos complejos, la rapidez en el aprendizaje y la habilidad de extraer lecciones de la experiencia. Revela una capacidad amplia y profunda para la comprensión del entorno, para captar el significado subyacente en las cosas y conferirles un propósito, así como para desenvolverse hábilmente al determinar cómo proceder (Felipe, 2021, como se cita en Sanz, 2020).

acotando la idea del autor Felipe (2021) debemos recalcar que la inteligencia se muestra como la capacidad central para el ser, la ejecución y la interacción en una variedad de contextos tan variados como: los dominios relacionados con la lógica y las matemáticas, las aptitudes en el lenguaje, las habilidades musicales, lo interno e interpersonal y lo social, y la dimensión corporal sistémica. Desde la perspectiva de la psicología y la neurociencia, se refiere sin duda a un fenómeno cuantificable, cuyo desarrollo o retroceso está ligado a factores de origen filogenético, ambiental, educativo y cultural. Siguiendo esta línea de pensamiento, el concepto de inteligencia artificial guarda cierta dependencia de las teorías y conceptos relativos a la inteligencia humana, pero con la particularidad de que en este momento.

Quizás la inteligencia artificial podría sobrepasar en varios aspectos las limitaciones y paradojas de la inteligencia humana, intensificando su papel como un complemento de esta o, en contraste, llegar a convertirse en un elemento opuesto; lo que depara el futuro está por verse. Son precisamente estas ideas y otras semejantes las que motivaron la creación de este ensayo. Es pertinente recordar que, en diversas obras literarias o cinematográficas, tales como "Yo, Robot"

de Isaac Asimov o "Terminator" dirigida por James Cameron, la inteligencia artificial en su forma robótica y antropomórfica se enreda en conflictos con la humanidad a medida que adquiere niveles altos de autonomía que le facultad a estas "entidades" tomar decisiones que pueden generar controversias desde una perspectiva ética, ontológica o jurídica. De cualquier modo, no cabe duda de que en el siglo XXI la inteligencia artificial se constituye en una realidad que supera en muchos aspectos a la ficción, ya que se encuentra de alguna manera en todos los sectores de la vida social contemporánea, desde los motores de búsqueda en línea que definen las preferencias y selecciones en el acceso a la información digital, hasta los refrigeradores inteligentes que pueden generar solicitudes de compra para garantizar la disponibilidad continua de ciertos alimentos a medida que se van agotando (pág. 504).

La inteligencia artificial es una tecnología con capacidad de hacer pensar por sí sola una máquina, según Rouhiainen (2018) define la inteligencia artificial como «la habilidad de los ordenadores para hacer actividades que normalmente requieren inteligencia humana». No obstante, para ofrecer una definición más elaborada, podríamos afirmar que la inteligencia artificial implica la habilidad de las máquinas para aplicar algoritmos, adquirir conocimiento a partir de datos y aplicar lo aprendido en la toma de decisiones de manera similar a un ser humano. Sin embargo, al contrario de los humanos, los dispositivos que se sustentan en la inteligencia artificial no necesitan reposo y pueden examinar volúmenes extensos de información de manera simultánea. Además, las máquinas que desempeñan las mismas tareas que sus homólogos humanos tienden a cometer significativamente menos errores.

En su análisis, Rouhiainen afirma que la idea de que los ordenadores o los programas informáticos puedan tanto aprender como tomar decisiones es particularmente importante y algo sobre lo que debemos estar conscientes, ya que sus procesos están experimentando un

crecimiento exponencial con el transcurso del tiempo. Gracias a estas dos habilidades, los sistemas de inteligencia artificial ahora son capaces de llevar a cabo numerosas tareas que solían ser exclusivas de los seres humanos.

Las tecnologías fundamentadas en la inteligencia artificial ya se están empleando para asistir a las personas a aprovechar mejoras significativas y obtener una mayor eficacia en prácticamente todos los aspectos de la vida. Sin embargo, el rápido avance de la inteligencia artificial también nos obliga a estar vigilantes con el fin de prevenir y analizar las posibles desventajas directas o indirectas que puedan surgir debido a la expansión de la inteligencia artificial. (pág. 17).

Figura 1.

Ejemplos donde podemos utilizar la inteligencia artificial.

Nota. Estos son los usos que le damos a la Inteligencia Artificial hoy en día. Tomado de (Cornieles, 2018).

Colaboración en el desarrollo del software.

Las acciones de los participantes en el desarrollo de software se asemejan en ciertos aspectos a las interacciones en dinámicas grupales, así como a los roles y reglas que emergen

para lograr exitosamente la realización orgánica y descentralizada de proyectos de programación en línea.

Conforme a lo expuesto por López Gil (2018) en el desarrollo software es importante saber lo que ocurre en nuestro entorno de trabajo, si además trabajamos en equipo se requiere un buen nivel de la conciencia situacional, para que los cambios significativos en tareas interconectadas y que por lo tanto afectan a varios miembros del equipo reciban la información necesaria para poder actuar en consecuencia. Si, además, este desarrollo software, es con metodologías ágiles, los miembros del equipo deben ser capaces de percibir los cambios y predecir cómo evolucionará el proyecto. De no ser así, los tiempos de desarrollo y pruebas aumentarán, la calidad del producto se verá afectada y la probabilidad de rechazo por parte del cliente se incrementará (pág. 21).

La colaboración en el desarrollo de software destaca, según López Gil (2018) debido a que las metodologías agiles, dan mucha más importancia a las personas que van a participar en el proyecto, siendo fundamental la comunicación entre todos sus miembros: cliente, director de proyecto, equipo del proyecto, interesados. No se centran en una entrega final con el producto terminado, sino que busca un desarrollo iterativo e incremental. Estas nuevas metodologías están dando muy buenos resultados por la adaptabilidad y flexibilidad que tiene hacia los cambios, ya sean de características del producto requisitos o tiempo de desarrollo. Los proyectos manejados mediante enfoques ágiles comienzan sin contar con una descripción detallada de lo que se construirá. A nivel comercial, los proyectos pueden ser vendidos como servicios y no como productos (pág. 82).

El análisis de López Gil (2018) asegura que para extraer unos factores que se puedan utilizar para hacer una comparativa de por qué hay proyectos que fracasan y otros que se

resuelven con éxito. Se desea comprobar en qué medida existen unos proyectos y otros y cuáles son las variables que hacen que finalicen de esta manera (pág. 98).

Por otro lado, conforme a lo expuesto por Ospino et al. (2022) afirma que los proyectos de creación de software requieren un nivel significativo de conocimiento, ya que involucran procedimientos de administración meticulosos que deben combinarse con normativas efectivas de conocimiento y gestión. Esta fusión guarda una estrecha conexión con la transmisión de saberes, la cual resulta esencial en este tipo de proyectos. Sin embargo, la falta de colaboración, la escasa comunicación, la falta de motivación para compartir el conocimiento y la incapacidad de ponerlo a disposición son algunas de las complicaciones dentro del ámbito de la gestión del conocimiento en los proyectos de desarrollo de software. Por otro lado, la gamificación impacta en las conductas individuales, fomentando la motivación, el compromiso y la cooperación en los grupos laborales. En este sentido, la gamificación puede ser una estrategia potencial para transformar estas dificultades (pág. 2)

La colaboración efectiva entre desarrolladores es esencial para el éxito de los proyectos de desarrollo de software. En este contexto, GitHub Copilot emerge como una herramienta potencialmente revolucionaria al utilizar inteligencia artificial y aprendizaje automático para proporcionar sugerencias de código y automatizar tareas rutinarias. GitHub Copilot se integra con la plataforma GitHub, un sistema de control de versiones ampliamente utilizado para la gestión colaborativa de proyectos de desarrollo (GitHub, 2021).

Por otro lado, el desarrollo de software con frecuencia es un área incompatible con las necesidades de sus actores; por ejemplo, existen procesos fríos e inflexibles que no reconocen la dimensión humana de los programadores es relevante; no obstante, en un entorno de desarrollo de software, los participantes son en última instancia individuos colaborando con otros

individuos.

Por lo tanto, es desconcertante cómo a veces son los propios involucrados quienes pasan por alto o minimizan las necesidades ajenas, o incluso explotan sus propias habilidades en beneficio propio. En el proceso de concebir programas informáticos, a menudo surgen exigencias laborales, tensiones, plazos apremiantes, largas jornadas y relaciones desafiantes entre colegas. Estos elementos deben ponderarse en favor de los desarrolladores con el objetivo de lograr simultáneamente mayores niveles de productividad laboral y de bienestar personal (Hernández, 2020).

Lenguajes de programación.

Un lenguaje de programación es un lenguaje estructurado creado con la finalidad de ejecutar tareas que son susceptibles de ser realizadas por dispositivos como las computadoras. Según Prieto & Te (2018) explica que un lenguaje de programación es una serie organizada de pautas, símbolos, caracteres y reglas que posibilitan que un programador logre expresar el tratamiento de información y los resultados obtenidos a través del empleo de computadoras. Cada lenguaje presenta una sintaxis propia, lo que resulta en notables distinciones entre ellos; se emplean para diseñar programas que encierran algoritmos, a través de los cuales el programador puede establecer una comunicación con la computadora y de esta manera direccionar las operaciones que ésta llevará a cabo. (pág. 4).

Como bien se conoce que las aplicaciones que utilizamos en el día a día, fueron creadas mediante algún lenguaje de programación, así lo explica Patricia Layedra Larrea et al. (2022) que desde la creación de instrucciones codificadas en código máquina hasta la elaboración de aplicaciones avanzadas para web y dispositivos móviles, los lenguajes de programación han tenido un rol esencial en este camino de la era informática.

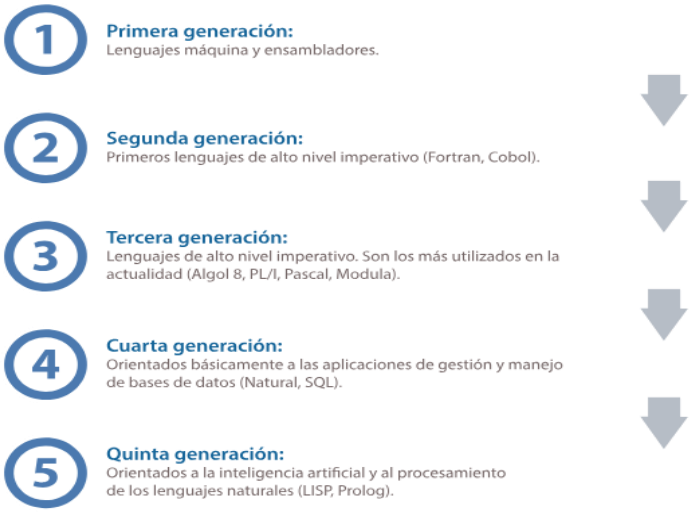
El desarrollo de sistemas se realiza utilizando entornos de programación que soportan los diferentes lenguajes de programación existentes, para crear aplicaciones que se puedan instalar en las diversas plataformas que están presentes en la actualidad y cuentan con una demanda significativa por parte de los usuarios de dichos sistemas. La formación que se brinda a los profesionales del desarrollo de software está basada en la enseñanza de los lenguajes de programación (pág. 1604).

Existen tres tipos de lenguajes de programación los cuales son:

- **Lenguaje de maquina:** Consiste en un conjunto de códigos binarios o bits que la computadora lee e interpreta; además, son los únicos sistemas de comunicación que las computadoras reconocen. Aunque las computadoras los comprenden con facilidad, los lenguajes de máquina resultan prácticamente inutilizables para los seres humanos, dado que están compuestos en su totalidad por números.
- **Lenguaje ensamblador**: Son convertidos al lenguaje de máquina mediante un software denominado ensamblador.
- **Lenguaje de alto nivel:** Los lenguajes de alto nivel nos habilitan para redactar códigos informáticos mediante indicaciones que guardan semejanza con el lenguaje común usado en la comunicación diaria, y posteriormente, estos códigos son transformados al lenguaje de máquina para ser ejecutados. Antes de poder ser ejecutados, los programas escritos en un lenguaje de alto nivel necesitan ser convertidos al lenguaje de máquina (Gervacio, 2018).

Figura 2.

Mapa conceptual de los tipos de lenguajes de programación ordenados según su generación.



Nota. La clasificación se da con base en el desarrollo de los lenguajes desde la aparición de las computadoras, que sigue un cierto paralelismo con las generaciones establecidas en la evolución de aquéllas. Tomado de (Ceballos, 2004).

Machine Learning.

El aprendizaje automático, también conocido como aprendizaje automatizado, aprendizaje de máquinas o aprendizaje computacional, constituye un subcampo dentro de las ciencias de la computación y una subdivisión de la inteligencia artificial. Su finalidad principal es forjar metodologías que capaciten a las computadoras para adquirir conocimiento por sí mismas. Basándonos en lo que Rojas (2020) expone que el machine learning, resuelve situaciones por sí solo a partir de un análisis de datos y cuantos más datos tengan resultados positivos, además, se emplean algoritmos en el proceso de análisis que modelan distintos datos de acuerdo con los requerimientos. A través de los datos de entrada, machine learning ejecutar un algoritmo y como resultado, genera más información para el problema. El objetivo de generar mayor cantidad de datos se fundamenta en las siguientes técnicas:

- Regresión lineal y polinómica.
- Árboles de decisión.

- Redes neuronales.
- Red bayesiana.
- Cadenas de Markov.

Estas técnicas permiten a machine learning reconocer patrones, extraer conocimiento, descubrir información y hacer predicciones. Se considera que cada persona aprende de una manera particular, utiliza los sentidos, la experiencia y sus habilidades cognitivas, también puede confiar en estrategias personales y técnicas de aprendizaje, por ejemplo, registrar apuntes, resolver problemas, leer, retener información, y señalar en libros.

En el entorno informático, el objetivo consiste en que las computadoras adquieran autonomía y de este modo sean capaces de aprender de manera automática sus propias destrezas, las cuales son definidas mediante algoritmos para el aprendizaje y la gestión de datos. El machine learning no es auto programación, sino auto aprendizaje de datos y experiencia para generar patrones y resolver nuevas tareas. Este aprendizaje es la combinación de técnicas, datos, conceptualización de análisis de datos y algoritmos para generar nuevos patrones o modelos de predicción (pág. 587).

En el análisis de Osorio, A., & Enerieth, N. (2020.) argumentan que el aprendizaje de máquinas, o machine learning en inglés, es el campo de estudio que permite a los computadores la posibilidad de aprender, a partir de datos, No es necesario realizar un análisis de programación explícita. En ese sentido, se plantea la definición de los tipos de aprendizajes los cuales son: Aprendizaje supervisado, aprendizaje no supervisado y aprendizaje reforzado (pág. 334).

- **Aprendizaje supervisado:** El tipo más simple de ml es el modelo de aprendizaje supervisado. Su función principal consiste en adquirir un modelo mediante información previamente categorizada, con el fin de luego atribuir una categoría a

datos no categorizados. Se le llama "supervisado" debido a que el programador establece las categorías que representan las salidas ideales para cada dato ingresado; es decir, es el programador quien especifica las clases o etiquetas en las que se quiere clasificar algo (Osorio, A., & Enerieth, N, 2020, pág 334.) .

- **Aprendizaje no supervisado:** El modelo de aprendizaje sin supervisión opera sin ningún tipo de etiquetas y los datos que se procesan son datos no estructurados o cuya estructura es desconocida. Este modelo examinar los datos y obtener la información pertinente según la situación, sin requerir una programación específica relacionada con una variable o función de búsqueda. En otras palabras, en el aprendizaje supervisado, el propósito es aprender a asignar entradas a salidas cuyos valores correctos son suministrados por el programador. En el aprendizaje no supervisado solo se tienen datos de entrada, sin la clasificación del programador (Osorio, A., & Enerieth, N, 2020, pág 337.).
- **Aprendizaje reforzado:** El modelo de ml de aprendizaje reforzado es un algoritmo usualmente implementado en el aprendizaje supervisado de análisis de regresión. De esta manera, el modelo adquiere conocimiento a través de un proceso de prueba y error. A pesar de lo mencionado, esta técnica puede ser aplicada en contextos de aprendizaje no supervisado y no siempre se refiere a una predicción; en ocasiones, puede estar relacionada simplemente asociada a un objetivo en específico, como lograr que una pintura se vea como una obra pintada por un ser humano (Osorio, A., & Enerieth, N, 2020, pág 336.).

GitHub Copilot.

La extensión 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, en el análisis de Torrres Ovalle (2022) explica que GitHub Copilot sugiere 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 forma similar, se integra en conjunto con una solución de inteligencia artificial denominada OpenAI Codex, habilitando así el uso de GitHub Copilot Diseñado para comprender tanto el lenguaje de programación como el lenguaje humano en diversos idiomas previamente registrados en repositorios de GitHub o en las opciones proporcionadas por el software, esto es aplicable en entornos de desarrollo integrado como Visual Studio Code, Neovim y JetBrains, además de presentar una variedad de opciones que se ajusten a las expectativas necesarias.

Como se mencionó anteriormente OpenAI Codex es un software complementario, es decir, este sistema utiliza datos abiertos y públicos que abarcan diversas fuentes de datos obtenidos de diversos repositorios, abarcando tanto aquellos en GitHub como los relacionados con 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 (pág. 3).

Figura 3.

Con tan solo escribir el comando para crear las tablas, vemos las líneas de código que sugiere el Copilot.

Nota. Impulsa lo que es posible con GitHub Copilot, la herramienta de desarrollo de IA más adoptada en el mundo. Tomado de (GitHub, 2021).

Por otro lado, los siguientes autores argumentan que:

Prácticamente podríamos decir que GitHub Copilot “programa solo” aunque realmente lo que se hace es un análisis de lo que el programador ha escrito hasta el momento para escribir el resto. No obstante, el programador deberá revisar constantemente si la solución sugerida es precisa y adecuada para los requisitos, o si es necesario realizar ajustes o rehacerla de manera distinta. El sistema podría devolver un código que no tenga sentido en su contexto o que no esté correctamente optimizado y, por tanto, la intervención del programador es siempre, y a día de hoy, imprescindible para revisar y probar el código, pero se podrá ahorrar una gran cantidad de tiempo (Pomares, 2021).

En el análisis de García (2022) afirma que este modelo ha sido entrenado para realizar recomendaciones de auto completado de códigos, de forma que, al colocar un comentario en la cabecera, es suficiente como para generar un auto completado en códigos que se tenía pensado,

además GitHub Copilot viene a ser un sorprendente servicio capaz de asistir a los desarrolladores, generando fragmentos de código y ofreciendo recomendaciones que les resulten útiles en sus actividades cotidianas (pág. 47).

Impacto en la colaboración.

El sistema se basa en Codex, un nuevo sistema de inteligencia artificial desarrollado por OpenAI, que indudablemente brindará un apoyo valioso en la educación, tanto para profesores como estudiantes, en diversas tareas que permitan el aprendizaje. Según García (2022) expone que esto supondrá un salto en el proceso de la enseñanza aprendizaje, donde el docente, si tuviera que solicitar una tarea como un ensayo, el estudiante, de acuerdo con la instrucción recibida, podría solicitar a GitHub Copilot, que lo redacte, y así sucedería, inclusive con la referencia bibliográfica incorporaras (pág. 48).

La herramienta tecnológica GitHub Copilot proporciona un inmenso valor a los desarrolladores de software, es esencial utilizarlo de forma responsable y mantener consideraciones éticas. Los desarrolladores deben tener en cuenta que las sugerencias de Copilot se originan a partir de patrones de código que están públicamente disponibles. Es crucial revisar y validar las sugerencias de código, asegurándose de que se ajustan a las normas de seguridad, privacidad y cumplimiento específicas del proyecto (Suárez, 2023).

GitHub Copilot tiene un impacto significativo en la colaboración entre desarrolladores:

1. **Aumento de Eficiencia:** GitHub Copilot agiliza el proceso de codificación al proporcionar sugerencias de código que se adaptan a la tarea en curso. Esto reduce el tiempo necesario para escribir código y permite a los desarrolladores centrarse en desafíos más complejos y creativos (GitHub, 2021).

2. **Facilitación de la Comunicación:** Copilot autocompleta fragmentos de código, sugiere nuevas líneas de código e incluso puede escribir funciones completas según la descripción proporcionada. Según lo compartido en el blog de GitHub, esta herramienta no se limita a ser un algoritmo de generación de lenguaje basado en las entradas del usuario; más bien, actúa como un compañero de programación. Además, tiene la capacidad de aprender y adaptarse a los hábitos de codificación del usuario, analiza el código fuente disponible y genera recomendaciones fundamentadas en enormes cantidades de líneas de código público en las que ha sido entrenado. Así mismo, como cualquier código que se escribe este debe ser evaluado y probado (Fernández., 2021).
3. **Inclusión de Desarrolladores Novatos:** Los desarrolladores menos experimentados tienen una mayor ventaja con herramientas como GitHub Copilot, lo cual es corroborado por otros estudios, incluyendo nuestros propios experimentos anteriores sobre el impacto de la inteligencia artificial en la productividad del desarrollador. A medida que los desarrolladores emplean estas herramientas para perfeccionar sus habilidades, se volverán más competentes en solicitar y colaborar con la inteligencia artificial para potenciar el proceso de desarrollo. Esto en última instancia contribuirá a democratizar la creación de software, a reducir la disparidad en el ámbito laboral y a consolidar las herramientas de codificación asistida por inteligencia artificial como un componente esencial en la experiencia educativa estándar del desarrollador (Griffiths, 2023).

Aprobación que genera GitHub Copilot.

La aprobación que obtuvo la extensión GitHub Copilot, después de estar usando esta aplicación profesionalmente y en proyectos personales durante varios meses, las ventajas que he encontrado en mi experiencia han sido numerosas (Huet, 2022).

- **Establece un recurso eficaz para tareas que se repiten con frecuencia.**

Gracias a su capacidad para comprender el contexto del archivo y del espacio de trabajo en el que se encuentra, se convierte en una herramienta excepcional cuando se requiere llevar a cabo tareas complementarias o completar plantillas de frameworks con menciones recurrentes a variables que están presentes en el archivo.

No solamente eso, sino que también es capaz de identificar la necesidad de utilizar una función que se encuentra en un lugar distinto al archivo actual y que haya sido usada o definida con anterioridad, incluso permitiendo hacer referencia a funciones de la API o auxiliares que se encuentren definidas en otras partes del proyecto.

- **Resulta altamente provechoso para la creación de funciones habituales.**

Copilot resulta muy útil para definir de manera ágil funciones que se emplean en prácticamente todas partes, sin la necesidad de importar extensas bibliotecas con funciones auxiliares. Además, dado que estas funciones suelen ser comunes, el código que genera generalmente es muy eficiente y tiende a requerir una revisión mínima o nula.

- **Es una valiosa fuente de ideas creativas e inspiradoras.**

Otra de las notables ventajas que he experimentado al utilizarlo es la extraordinaria fuente de inspiración, que incluso incluye funciones, atributos del lenguaje y similares, que me ha ofrecido. Dado que el código del que ha aprendido ha sido aportado por desarrolladores, a veces, aunque la sugerencia termine siendo ineficiente o inexacta, En muchos casos, las sugerencias que proporciona resultan sumamente beneficiosas y contribuyen al desarrollo profesional.

- **Nos desafía a ser mejores programadores.**

En conexión con la ventaja previamente mencionada, el hecho de que genere nuevas ideas o que produzca código que requerimos analizar minuciosamente para comprender su lógica o

justificación, Nos motiva en cierta medida a buscar soluciones creativas que empleen los mismos principios y, de alguna forma, nos lleva a reconsiderar el código empleado en funciones similares que hayamos definido con antelación.

Desafíos e Implicaciones que tiene GitHub Copilot.

A pesar de sus ventajas, GitHub Copilot también plantea desafíos.

1. **Seguridad y calidad de código**

Copilot pueda introducir huecos de seguridad en nuestro código, al sugerirnos usar fragmentos de código con vulnerabilidades o que usa a su vez bibliotecas obsoletas e inseguras. Por ejemplo, si Copilot ha sido instruido con segmentos de código que incorporan contraseñas en texto claro, te ofrecerá fragmentos de código similares. Si no comprendes completamente lo que estás haciendo, podrías terminar con la misma vulnerabilidad en tu propio código. Por esta razón, es de gran relevancia que perfeccionemos nuestra habilidad para interpretar y comprender el código de otras personas.

Por lo tanto, es de suma importancia que perfeccionemos nuestra capacidad para leer y comprender el código de otros. Esperamos que más escuelas den cursos de lectura de código, y que más empresas formalicen esta práctica entre sus desarrolladores (Galván, 2021).

2. **Dependencia Excesiva.**

La dependencia excesiva en las sugerencias de GitHub Copilot podría limitar la creatividad y la comprensión profunda del código. Los desarrolladores deben equilibrar el uso de la extensión con su propio conocimiento y experiencia (GitHub, 2021).

MARCO METODOLÓGICO

Durante la etapa de recolección de datos para este caso de estudio. Se utilizó el enfoque de investigación el método inductivo-deductivo, el cual posibilitó la integración entre la teoría y la información empírica al identificar las conexiones existentes entre los conceptos teóricos y los datos recolectados en el estudio.

Adicionalmente, se empleó la metodología de investigación cualitativa, el cual posibilitó una exploración más íntegra y rigurosa del objeto de estudio. Esto condujo a la formulación de conclusiones concretas y robustas en relación al "Análisis de la extensión GitHub Copilot, en el ámbito de colaboración hacia los desarrolladores, para aumentar, la eficacia en el desarrollo de aplicaciones web".

Las técnicas que se utilizaron fue documental y descriptiva, ya que permitió la recopilación de información para enunciar los conceptos que sustentan el estudio de los fenómenos y los procesos de la investigación.

Como instrumento de investigación para la recopilación de datos fue la entrevista, diseñado para obtener una comprensión más profunda, fue aplicado a través de entrevistas en persona. Estas preguntas fueron dirigidas a profesionales involucrados en el campo del desarrollo de software, con el propósito de recopilar información valiosa y lograr una comprensión más completa relacionada con la problemática que enfrentan en su labor.

RESULTADOS

ANÁLISIS DE LA ENTREVISTA.

Esta entrevista directa se pudo obtener como objetivo central conocer la manera en que desempeñan su trabajo los desarrolladores de aplicaciones web, de esta manera recolectar información sobre sus criterios y opiniones propias, cabe recalcar que el personal de programadores que desempeñan un rol importante en la programación en ámbito laboral colaboró con toda la disposición que ameritaba este trabajo investigativo.

Una vez llevada a cabo la entrevista con las personas que tienen conocimiento sobre la extensión GitHub Copilot, en el ámbito de colaboración hacia los desarrolladores, para aumentar, la eficacia en el desarrollo de proyectos, se obtuvieron los siguientes resultados:

En relación a la interrogante de la pregunta **¿Cómo cree usted que la extensión GitHub Copilot podría mejorar el proceso de la programación al momento de desarrollar aplicaciones web?**

El ingeniero Sebastián Coello, opina que la extensión GitHub Copilot es que tiene el potencial de mejorar significativamente el proceso de programación al desarrollar aplicaciones web. Al proporcionar sugerencias relevantes y útiles en tiempo real.

El ingeniero César Lata, opina que la extensión GitHub Copilot tiene todas las capacidades de ser una herramienta revolucionaria en el proceso de programación, especialmente en el desarrollo de aplicaciones web. Al proporcionar sugerencias de código y autocompletar funciones y estructuras de manera inteligente.

El ingeniero Jorge Díaz, opina que la extensión GitHub Copilot tiene el potencial de mejorar la eficiencia en el desarrollo de aplicaciones web al proporcionar sugerencias de código relevantes y reducir la posibilidad de errores.

La ingeniera Mayken Salavarria, opina que la GitHub Copilot mejora el proceso de programación ya que tiene el potencial de ser una herramienta altamente beneficiosa en el proceso de programación para el desarrollo de aplicaciones web.

El licenciado Hernán Quinde, opina que la extensión GitHub Copilot podría revolucionar la forma en que los diseñadores web abordan el desarrollo de aplicaciones. Facilitaría la escritura de código al proporcionar sugerencias en tiempo real, lo que podría agilizar la implementación de elementos de diseño y características interactivas en las aplicaciones web.

En relación a la interrogante de la pregunta **¿Cuál es su opinión sobre la implementación de la extensión GitHub Copilot en la toma de decisiones, tales como la sugerencia o la predicción de código de programación?**

El ingeniero Sebastián Coello, opina que la implementación de GitHub Copilot en la toma de decisiones, como la sugerencia o la predicción de código de programación, es un paso emocionante hacia el futuro de la programación. Esta tecnología puede ser una herramienta valiosa para los desarrolladores al proporcionar recomendaciones basadas en buenas prácticas de programación y patrones establecidos.

El ingeniero César Lata, opina que la implementación de GitHub Copilot en la toma de decisiones es una evolución interesante en la programación. Personalmente, lo veo como una herramienta valiosa para aumentar la productividad y la calidad del código. Sin embargo, es importante recordar que las sugerencias y predicciones de código deben ser evaluadas críticamente por los desarrolladores.

El ingeniero Jorge Díaz, opina que la implementación de GitHub Copilot en la toma de decisiones es una evolución positiva, pero debe ser utilizada con discernimiento. No debe

reemplazar el juicio humano y la experiencia en la toma de decisiones importantes de programación.

La ingeniera Mayken Salavarria, opina que la implementación de GitHub Copilot en la toma de decisiones, como la sugerencia o la predicción de código de programación, es un avance emocionante. Puede servir como una herramienta valiosa para respaldar la toma de decisiones al ofrecer recomendaciones basadas en las mejores prácticas de programación y patrones establecidos.

El licenciado Hernán Quinde, opina que la implementación de GitHub Copilot en la toma de decisiones es intrigante. Puede ofrecer sugerencias inteligentes para la creación de prototipos y la implementación de interacciones, lo que podría agilizar el proceso de diseño web.

En relación a la interrogante de la pregunta **¿Cuál es su opinión acerca del papel que ejercerá la extensión GitHub Copilot en el futuro de la programación para satisfacer mejor las necesidades de los desarrolladores?**

El ingeniero Sebastián Coello, opina que la extensión GitHub Copilot tiene el potencial de desempeñar un papel fundamental en el futuro de la programación al satisfacer mejor las necesidades de los desarrolladores. Al ofrecer sugerencias y código de alta calidad de manera instantánea, puede acelerar el proceso de desarrollo y permitir a los desarrolladores centrarse en la lógica y la creatividad de sus aplicaciones en lugar de perder tiempo en tareas repetitivas.

El ingeniero César Lata, opina el papel futuro de GitHub Copilot en la programación, creo que será una parte integral de nuestro proceso de desarrollo. A medida que la tecnología continúe evolucionando, GitHub Copilot mejorará su capacidad para satisfacer las necesidades de los desarrolladores. Esto significa que podemos esperar un aumento en la velocidad de desarrollo y una reducción en la carga de trabajo repetitiva.

El ingeniero Jorge Díaz, opina que GitHub Copilot jugará un papel esencial en el futuro de la programación al permitir que los desarrolladores se enfoquen en la resolución de problemas y la creatividad, en lugar de tareas repetitivas. Esto mejorará la calidad y eficiencia en el desarrollo de software.

La ingeniera Mayken Salavarria, opina que GitHub Copilot desempeñará un papel fundamental en el futuro de la programación. A medida que la tecnología continúe avanzando, esta herramienta evolucionará para satisfacer de manera más efectiva las necesidades de los desarrolladores.

El licenciado Hernán Quinde, opina que, en el futuro, GitHub Copilot podría desempeñar un papel esencial en el diseño web al satisfacer las necesidades de los profesionales del diseño y desarrollo. Esta herramienta tiene el potencial de aumentar la eficiencia y la calidad del diseño web al reducir el tiempo dedicado a la codificación manual. Los diseñadores web podrían enfocarse en la innovación y la creatividad, lo que podría resultar en experiencias web más atractivas y efectivas.

En relación a la interrogante de la pregunta **¿Cuáles cree usted que podrían ser algunos posibles desafíos o limitaciones al utilizar GitHub Copilot en un entorno de desarrollo?**

El ingeniero Sebastián Coello, opina que la extensión GitHub Copilot tiene muchas ventajas, también presenta desafíos y limitaciones. Uno de los desafíos es garantizar que el código generado sea seguro y cumpla con los estándares de seguridad, ya que la extensión no siempre puede comprender completamente el contexto específico de un proyecto.

El ingeniero César Lata, opina que GitHub Copilot en mi opinión serían los desafíos de garantizar la seguridad del código, ya que la extensión puede no entender completamente el contexto y los requisitos específicos de cada proyecto. Además, debemos abordar

preocupaciones éticas y legales, como la posible infracción de derechos de autor cuando se utiliza código generado automáticamente.

El ingeniero Jorge Díaz, opina que algunos posibles desafíos al utilizar GitHub Copilot incluyen la necesidad de garantizar la seguridad del código generado, la validación de sugerencias automáticas y cuestiones éticas y legales relacionadas con la originalidad del código y los derechos de autor. También podría requerir una supervisión constante para mantener la calidad del código.

La ingeniera Mayken Salavarria, opina que algunos posibles desafíos y limitaciones al utilizar GitHub Copilot podrían incluir la necesidad de validar las sugerencias automáticas para garantizar la seguridad y la coherencia del código generado.

El licenciado Hernán Quinde, opina que al utilizar GitHub Copilot en un entorno de desarrollo, es crucial considerar la necesidad de supervisar y validar las sugerencias automáticas. Los diseñadores web deben asegurarse de que las sugerencias se alineen con los estándares de diseño y la usabilidad. Además, es importante abordar las cuestiones éticas y legales relacionadas con la originalidad del diseño y los derechos de autor al utilizar el código generado automáticamente.

DISCUSIÓN DE RESULTADOS

A lo largo de la investigación se ha determinado en consideración puntos importantes, las cuales son la programación, y la inteligencia artificial. La extensión GitHub Copilot que es una inteligencia artificial programadora que abarca temas que van desde la eficiencia y la calidad del código hasta la ética, la formación y la incorporación en los procesos de trabajo de desarrollo. Su implementación plantea desafíos y oportunidades que la comunidad de desarrolladores debe considerar cuidadosamente.

La extensión GitHub Copilot es una tecnología emocionante que tiene el potencial de mejorar la productividad y la calidad del código, pero también plantea desafíos importantes que deben ser abordados por la comunidad de desarrolladores. La discusión continua sobre su implementación y mejores prácticas es esencial para aprovechar al máximo esta herramienta sin comprometer la calidad del desarrollo de software.

La entrevista realizada a desarrolladores de aplicaciones web proporciona información valiosa sobre la percepción y la potencial influencia de GitHub Copilot en su trabajo diario. Los resultados revelan una serie de puntos clave que merecen atención y discusión.

Mejora en el Proceso de Programación: La mayoría de los entrevistados, concuerdan en que GitHub Copilot tiene el potencial de mejorar significativamente el proceso de programación en el desarrollo de aplicaciones web. La capacidad de proporcionar sugerencias útiles y relevantes en tiempo real se destaca como una ventaja clave. Esto sugiere que los desarrolladores ven a GitHub Copilot como una herramienta que puede acelerar la codificación y aumentar la eficiencia en el desarrollo.

Implementación en la Toma de Decisiones: La implementación de GitHub Copilot en la toma de decisiones, como la sugerencia o la predicción de código de programación, se ve con

entusiasmo por parte de los entrevistados. Sin embargo, existe un consenso en que estas sugerencias deben ser evaluadas críticamente por los desarrolladores. Esto resalta la importancia de no depender completamente de la herramienta y de mantener un papel activo en la toma de decisiones importantes de programación.

Papel en el Futuro de la Programación: Los entrevistados coinciden en que GitHub Copilot tiene el potencial de desempeñar un papel fundamental en el futuro de la programación al satisfacer mejor las necesidades de los desarrolladores. La herramienta podría permitir que los desarrolladores se centren en tareas más creativas y estratégicas al acelerar la generación de código. Esta percepción refleja una visión positiva de cómo GitHub Copilot podría influir en el desarrollo de software en el futuro.

Desafíos y Limitaciones: Los entrevistados señalan varios desafíos y limitaciones potenciales en el uso de GitHub Copilot. Estos incluyen la necesidad de garantizar la seguridad del código generado, la validación de las sugerencias automáticas y preocupaciones éticas y legales relacionadas con la originalidad del código y los derechos de autor. Estos desafíos subrayan la importancia de supervisar y ajustar el uso de la herramienta para mantener la calidad y la integridad del desarrollo de software.

CONCLUSIONES

Con la información que se obtuvo del análisis teórico de la extensión GitHub Copilot es una extensión prometedora en el ámbito de la colaboración para desarrolladores. Sus funcionalidades clave, ventajas y limitaciones han sido identificadas en este estudio. Entre sus ventajas se incluyen la capacidad de generar código de manera rápida y sugerir soluciones, lo que puede aumentar la productividad. Sin embargo, también tiene limitaciones, como la necesidad de una formación adecuada para su uso eficaz y la posibilidad de generar código no óptimo en ciertas situaciones.

La implementación de GitHub Copilot ha demostrado un impacto positivo en la colaboración entre desarrolladores. Se ha observado una reducción en el tiempo de desarrollo y una mayor precisión en la implementación de código, lo que ha mejorado la eficacia de los equipos de desarrollo. Los desarrolladores han encontrado útil la herramienta para agilizar la generación de código, lo que ha liberado tiempo para tareas más creativas y estratégicas.

Las pruebas de concepto ejecutadas en este estudio han demostrado que GitHub Copilot puede tener un impacto positivo en la eficacia de la colaboración y el desarrollo de proyectos. La extensión puede acelerar la escritura de código y facilitar la comprensión del mismo entre los miembros del equipo. Sin embargo, es esencial que los equipos establezcan prácticas de revisión de código sólidas para garantizar la calidad y coherencia del código generado.

RECOMENDACIONES

Basado en el análisis y discusión previos, a continuación, se presentan algunas recomendaciones:

Se recomienda que los programadores nuevos inviertan en programas de formación y capacitación continua. Esto permitirá a los programadores aprovechar al máximo las funcionalidades de la herramienta y comprender sus limitaciones. La formación adecuada es esencial para garantizar un uso eficaz y ético de GitHub Copilot.

Para aprovechar al máximo el impacto positivo de GitHub Copilot en la colaboración entre desarrolladores, se recomienda fomentar una cultura de revisión de código colaborativa y efectiva. Esto implica establecer prácticas sólidas de revisión de código, donde los programadores del equipo se involucren activamente en la validación y mejora de las sugerencias de la herramienta. Además, se debe alentar la comunicación abierta y la compartición de conocimientos entre los desarrolladores.

Para garantizar la eficacia de GitHub Copilot en el desarrollo de proyectos, se recomienda implementar prácticas de revisión de código sólidas. Esto implica que los programadores establezcan procedimientos y estándares claros para evaluar y validar las sugerencias de la herramienta. La revisión de código continua y efectiva es esencial para mantener la calidad y coherencia del código generado por GitHub Copilot.

REFERENCIAS

Osorio, A., & Enerieth, N. (2020). *En La Inteligencia Artificial De Machine Learning*. (n.d.). 327–353.

Felipe, D. (2021). Inteligencia artificial y condición humana: ¿Entidades contrapuestas o fuerzas complementarias? *Revista de Ciencias Sociales*, *XXVII*(2). https://doi.org/10.31876/rcs.v27i2.35937

García, J. (2022). *Implication of Artificial Intelligencein Virtual Classrooms for Higher Education*. *10*(2709-8001.), 31–52.

Ceballos, F. (2004). Enciclopedia del lenguaje C. Mexico: Alfaomega/RaMa.

Cornieles, P. (31 de Octubre de 2018). *IA LATAM*. Obtenido de https://ia-latam.com/2018/10/31/estos-son-los-usos-que-le-damos-a-la-inteligencia-artificial-hoy-en-dia/

Fernández., M. d. (13 de Julio de 2021). *ciiia*. Obtenido de ciiia.mx/noticiasciiia/github-copilot-y-su-impacto-en-la-programacin

Galván, P. (2021). *SG*. Obtenido de https://sg.com.mx/buzz/que-es-github-copilot-y-que-implicaciones-tiene

Gervacio, L. O. (23 de Abril de 2018). *Conogasi*. Obtenido de https://conogasi.org/articulos/lenguaje-de-programacion/

GitHub. (2021). *GitHub*. Obtenido de GitHub Copilot: https://github.com/features/copilot/

Griffiths, A. L. (28 de Junio de 2023). *Dev*. Obtenido de https://dev.to/github/github-copilot-un-ano-revolucionando-la-ia-466b

Hernández, R. R. (2020). *Scielo.* Obtenido de https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S2007-

7890202000800028#:~:text=development%3B%20human%20interactions-,El%20desarrollo%20colaborativo%20se%20refiere%20al%20proceso%20de%20creaci%C3%B3n%20de,interacci%C3%B3n%20se%20lleve%20a%20c

Huet, P. (04 de Noviembre de 2022). *OpenWebinars*. Obtenido de https://openwebinars.net/blog/experiencia-con-copilot-tras-8-meses-de-uso/

Pomares, D. M. (14 de Julio de 2021). *VassCompany*. Obtenido de https://vasscompany.com/github-copilot/

Suárez, I. (21 de Junio de 2023). *Apiumhub*. Obtenido de https://apiumhub.com/es/tech-blog-barcelona/github-copilot-como-la-ia-esta-transformando-el-desarrollo-de-software/#:~:text=GitHub%20Copilot%20ayuda%20a%20los,fomenta%20la%20legibilidad%20del%20c%C3%B3digo.

López Gil, A. (2018). Estudio comparativo de metodologías tradicionales y ágiles para proyectos de Desarrollo de Software. *Universidad de Valladolid*, 139. https://uvadoc.uva.es/handle/10324/32875%0Ahttps://uvadoc.uva.es/handle/10324/32875%0Ahttps://agileexperience.es/wp-content/uploads/2020/06/TFG-I-1015.pdf%0Ahttp://uvadoc.uva.es/handle/10324/32875

Ospino, S. G., Villegas, L. E. M., & Hurtado, G. P. G. (2022). Transferencia de conocimiento en equipos de desarrollo de software: una estrategia de gamificación para fomentar la colaboración. *Revista Ingenierías Universidad de Medellín*, *21*(41), 1–27. https://revistas.udem.edu.co/index.php/ingenierias/article/view/3956

Patricia Layedra Larrea, N. I., Alejandro Salazar Cazco III, S., Vinicio Ramos Valencia, M. I., & Alexander Baldeón Hermida, B. I. (2022). *Ciencias de la Educación Artículo de Investigación Escuela Superior Politécnica de Chimborazo, Ecuador*. *8*(3), 1601–1625.

http://dominiodelasciencias.com/ojs/index.php/es/index

Prieto, B., & Te, M. (2018). *on Cu ´ antico Indice general*. *September*.

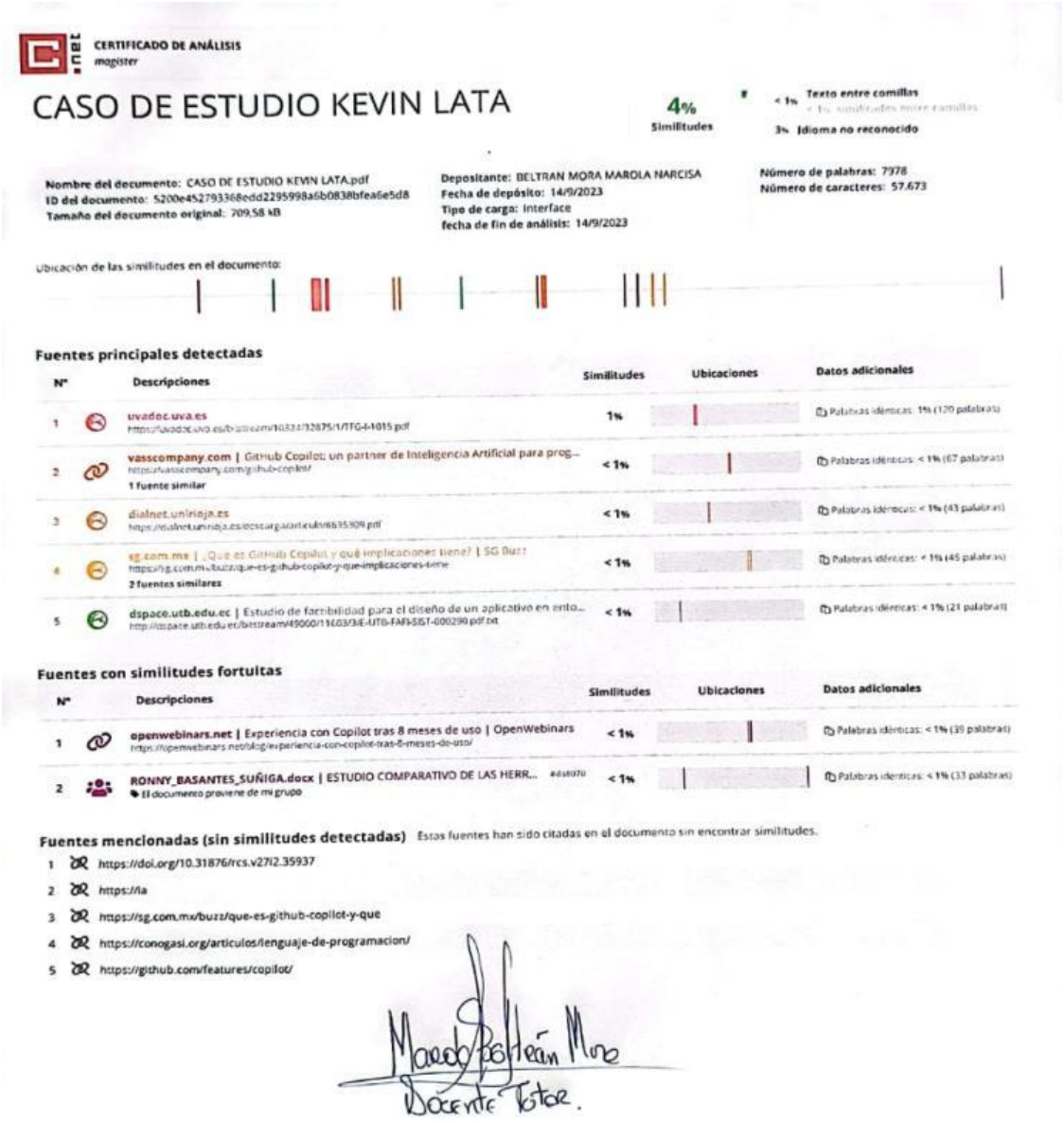
Rojas, E. M. (2020). Machine Learning: análisis de lenguajes de programación y herramientas para desarrollo. *Revista Ibérica de Sistemas e Tecnologias de Informação*, *E28*, 586–599.

Rouhiainen, L. (2018). Inteligencia artificial 101 cosas que debes saber hoy sobre nuestro futuro. *Alienta Editorial*, 22. https://planetadelibrosar0.cdnstatics.com/libros_contenido_extra/40/39307_Inteligencia_artificial.pdf

Torrres Ovalle, B. S. (2022). *GitHub Copilot*. 1–11. https://doi.org/10.26507/paper.2300

ANEXOS

RESULTADOS DEL CERTIFICADO DE ANÁLISIS DE PLAGIO



CERTIFICADO DE ANÁLISIS
magister

CASO DE ESTUDIO KEVIN LATA

4% Similitudes

< 1% Texto entre comillas

3% Idioma no reconocido

Nombre del documento: CASO DE ESTUDIO KEVIN LATA.pdf
ID del documento: 5200e452793368edd2295998a6b0838bfea6e5d8
Tamaño del documento original: 709,58 kB

Depositante: BELTRAN MORA MAROLA NARCISA
Fecha de depósito: 14/9/2023
Tipo de carga: interface
fecha de fin de análisis: 14/9/2023

Número de palabras: 7978
Número de caracteres: 57.673

Ubicación de las similitudes en el documento:

Fuentes principales detectadas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	uvadoc.uva.es	1%		Palabras idénticas: 1% (120 palabras)
2	vasscompany.com | GitHub Copilot: un partner de Inteligencia Artificial para prog... 1 fuente similar	< 1%		Palabras idénticas: < 1% (67 palabras)
3	dialnet.unirioja.es	< 1%		Palabras idénticas: < 1% (43 palabras)
4	sg.com.mx | ¿Qué es GitHub Copilot y qué implicaciones tiene? | SG Buzz 2 fuentes similares	< 1%		Palabras idénticas: < 1% (45 palabras)
5	dspace.utb.edu.ec | Estudio de factibilidad para el diseño de un aplicativo en ento...	< 1%		Palabras idénticas: < 1% (21 palabras)

Fuentes con similitudes fortuitas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	openwebinars.net | Experiencia con Copilot tras 8 meses de uso | OpenWebinars	< 1%		Palabras idénticas: < 1% (39 palabras)
2	RONNY_BASANTES_SUÑIGA.docx | ESTUDIO COMPARATIVO DE LAS HERR... El documento proviene de mi grupo	< 1%		Palabras idénticas: < 1% (33 palabras)

Fuentes mencionadas (sin similitudes detectadas) Estas fuentes han sido citadas en el documento sin encontrar similitudes.

1 https://doi.org/10.31876/rcs.v27i2.35937
2 https://ia
3 https://sg.com.mx/buzz/que-es-github-copilot-y-que
4 https://conogasi.org/articulos/lenguaje-de-programacion/
5 https://github.com/features/copilot/

Docente Tutor.

ANEXO ENTREVISTA. 1

TEMA

ANÁLISIS DE LA EXTENSIÓN GITHUB COPILOT, EN EL AMBITO DE COLABORACIÓN HACIA LOS DESARROLLADORES, PARA AUMENTAR, LA EFICACIA EN EL DESARROLLO DE PROYECTOS.

ENTREVISTA

Nombre: Sebastián Salvador Coello Quezada.

Profesión: Ingeniero en Sistemas Computacionales.

Lugar de trabajo: VAOSGROUP S.A.

¿Cómo cree usted que la extensión GitHub Copilot podría mejorar el proceso de la programación al momento de desarrollar aplicaciones web?

Mi opinión acerca de la extensión GitHub Copilot es que tiene el potencial de mejorar significativamente el proceso de programación al desarrollar aplicaciones web. Al proporcionar sugerencias relevantes y útiles en tiempo real.

¿Cuál es su opinión sobre la implementación de la extensión GitHub Copilot en la toma de decisiones, tales como la sugerencia o la predicción de código de programación?

Como ingeniero en sistemas computacionales, creo que la implementación de GitHub Copilot en la toma de decisiones, como la sugerencia o la predicción de código de programación, es un paso emocionante hacia el futuro de la programación. Esta tecnología puede ser una herramienta valiosa para los desarrolladores al proporcionar recomendaciones basadas en buenas prácticas de programación y patrones establecidos.

¿Cuál es su opinión acerca del papel que ejercerá la extensión GitHub Copilot en el futuro de la programación para satisfacer mejor las necesidades de los desarrolladores?

La extensión GitHub Copilot tiene el potencial de desempeñar un papel fundamental en el futuro de la programación al satisfacer mejor las necesidades de los desarrolladores. Al ofrecer sugerencias y código de alta calidad de manera instantánea, puede acelerar el proceso de desarrollo y permitir a los desarrolladores centrarse en la lógica y la creatividad de sus aplicaciones en lugar de perder tiempo en tareas repetitivas.

¿Cuáles cree usted que podrían ser algunos posibles desafíos o limitaciones al utilizar GitHub Copilot en un entorno de desarrollo?

Aunque GitHub Copilot tiene muchas ventajas, también presenta desafíos y limitaciones. Uno de los desafíos es garantizar que el código generado sea seguro y cumpla con los estándares de seguridad, ya que la extensión no siempre puede comprender completamente el contexto específico de un proyecto.

Sebastian Coello

Ing. Sebastián Salvador Coello Quezada.

ANEXO ENTREVISTA. 2

TEMA

ANÁLISIS DE LA EXTENSIÓN GITHUB COPILOT, EN EL AMBITO DE COLABORACIÓN HACIA LOS DESARROLLADORES, PARA AUMENTAR, LA EFICACIA EN EL DESARROLLO DE PROYECTOS.

ENTREVISTA

Nombre: César Santos Lata Jácome.

Profesión: Ingeniero en Sistemas.

Lugar de trabajo: VAOSGROUP S.A.

¿Cómo cree usted que la extensión GitHub Copilot podría mejorar el proceso de la programación al momento de desarrollar aplicaciones web?

Creo que la extensión GitHub Copilot tiene todas las capacidades de ser una herramienta revolucionaria en el proceso de programación, especialmente en el desarrollo de aplicaciones web. Al proporcionar sugerencias de código y autocompletar funciones y estructuras de manera inteligente.

¿Cuál es su opinión sobre la implementación de la extensión GitHub Copilot en la toma de decisiones, tales como la sugerencia o la predicción de código de programación?

La implementación de GitHub Copilot en la toma de decisiones es una evolución interesante en la programación. Personalmente, lo veo como una herramienta valiosa para aumentar la productividad y la calidad del código. Sin embargo, es importante recordar que las sugerencias y predicciones de código deben ser evaluadas críticamente por los desarrolladores.

¿Cuál es su opinión acerca del papel que ejercerá la extensión GitHub Copilot en el futuro de la programación para satisfacer mejor las necesidades de los desarrolladores?

En mi opinión en cuanto al papel futuro de GitHub Copilot en la programación, creo que será una parte integral de nuestro proceso de desarrollo. A medida que la tecnología continúe evolucionando, GitHub Copilot mejorará su capacidad para satisfacer las necesidades de los desarrolladores. Esto significa que podemos esperar un aumento en la velocidad de desarrollo y una reducción en la carga de trabajo repetitiva.

¿Cuáles cree usted que podrían ser algunos posibles desafíos o limitaciones al utilizar GitHub Copilot en un entorno de desarrollo?

Los posibles desafíos y limitaciones que tendría GitHub Copilot en mi opinión serían los desafíos de garantizar la seguridad del código, ya que la extensión puede no entender completamente el contexto y los requisitos específicos de cada proyecto. Además, debemos abordar preocupaciones éticas y legales, como la posible infracción de derechos de autor cuando se utiliza código generado automáticamente.

Ing. César Santos Lata Jácome.

ANEXO ENTREVISTA. 3

TEMA

ANÁLISIS DE LA EXTENSIÓN GITHUB COPILOT, EN EL AMBITO DE COLABORACIÓN HACIA LOS DESARROLLADORES, PARA AUMENTAR, LA EFICACIA EN EL DESARROLLO DE PROYECTOS.

ENTREVISTA

Nombre: Jorge Isrrael Díaz Montoya.

Profesión: Ingeniero en Sistemas.

Lugar de trabajo: VAOSGROUP S.A.

¿Cómo cree usted que la extensión GitHub Copilot podría mejorar el proceso de la programación al momento de desarrollar aplicaciones web?

En mi opinión acerca de la extensión GitHub Copilot tiene el potencial de mejorar la eficiencia en el desarrollo de aplicaciones web al proporcionar sugerencias de código relevantes y reducir la posibilidad de errores.

¿Cuál es su opinión sobre la implementación de la extensión GitHub Copilot en la toma de decisiones, tales como la sugerencia o la predicción de código de programación?

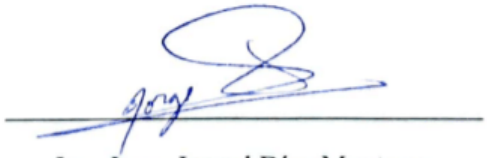
Como ingeniero en sistemas, mi opinión sobre la implementación de GitHub Copilot en la toma de decisiones es una evolución positiva, pero debe ser utilizada con discernimiento. No debe reemplazar el juicio humano y la experiencia en la toma de decisiones importantes de programación.

¿Cuál es su opinión acerca del papel que ejercerá la extensión GitHub Copilot en el futuro de la programación para satisfacer mejor las necesidades de los desarrolladores?

Creo que GitHub Copilot jugará un papel esencial en el futuro de la programación al permitir que los desarrolladores se enfoquen en la resolución de problemas y la creatividad, en lugar de tareas repetitivas. Esto mejorará la calidad y eficiencia en el desarrollo de software.

¿Cuáles cree usted que podrían ser algunos posibles desafíos o limitaciones al utilizar GitHub Copilot en un entorno de desarrollo?

En mi opinión como ingeniero en sistemas es que algunos posibles desafíos al utilizar GitHub Copilot incluyen la necesidad de garantizar la seguridad del código generado, la validación de sugerencias automáticas y cuestiones éticas y legales relacionadas con la originalidad del código y los derechos de autor. También podría requerir una supervisión constante para mantener la calidad del código.



Ing. Jorge Isrrael Díaz Montoya.

EVIDENCIA DE LA ENTREVISTA REALIZADA

EVIDENCIA DEL USO DE LA EXTENSIÓN GITHUB COPILOT, EN EL AMBITO LABORAL