Uso de herramientas de IA generativa para la automatización del desarrollo de software: un caso de estudio

RESUMEN: La Inteligencia Artificial Generativa (IA generativa) está demostrando tener aplicaciones transversales en diversas áreas y particularmente en el campo educativo está teniendo un impacto significativo. Esta rama de la IA permite desarrollar nuevas herramientas y técnicas para transformar y mejorar la experiencia de aprendizaje. El presente artículo presenta dos usos concretos de la IA generativa, aplicables tanto para el desarrollo del software como para el entorno educativo: 1) agilizar la creación de una plataforma web para diseño, aplicación y calificación de exámenes sobre diferentes materias, y 2) generar de forma automatizada los reactivos que servirán para evaluar dichas asignaturas. El propósito fue demostrar que la IA generativa, que emplea modelos de lenguaje entrenados (LLM), puede acelerar significativamente el desarrollo de software mediante la automatización de código, dado que en contraste con técnicas tradicionales se lograron reducciones del 91.5% en tiempos de desarrollo y del 85.6% en iteraciones de prueba y error. Si bien la IA generativa es una herramienta prometedora, se requiere supervisión humana para entregar instrucciones precisas, considerando las limitaciones actuales e implicaciones éticas de estas tecnologías emergentes.

PALABRAS CLAVE: Inteligencia Artificial, IA generativa, aplicaciones en educación, desarrollo de software, tecnología y educación.

Colaboración
Armando Roman Gallardo; Jose Roman Herrera Morales; Sara Sandoval Carrillo; Omar Alvarez Cardenas, Universidad de Colima

Fecha de recepción: 14 de marzo 2024
Fecha de aceptación: 10 de junio de 2024

ABSTRACT: Generative Artificial Intelligence (generative AI) is proving to have transversal applications in various areas and particularly in the educational field it is having a significant impact. This branch of AI allows us to develop new tools and techniques to transform and improve the learning experience. This article presents two specific uses of generative AI, applicable to both software development and the educational environment: 1) expedite the creation of a web platform for the design, application and grading of exams on different subjects, and 2) generate in an automated manner the reagents that will be used to evaluate these subjects. The purpose was to demonstrate that generative AI, which uses trained language models (LLM), can significantly accelerate software development through code automation, given that in contrast to traditional techniques, reductions of 91.5% in development times and 85.6% in trial and error iterations. While generative AI is a promising tool, human supervision is required to deliver accurate instructions, considering the current limitations and ethical implications of these emerging technologies.

KEYWORDS: Artificial Intelligence, Generative AI, Education applications, Software development, technology and education.

INTRODUCCIÓN

La Inteligencia Artificial (IA) es una rama de la informática que estudia la creación de agentes inteligentes, que son sistemas que pueden razonar, aprender y actuar de forma autónoma. La IA se puede dividir en dos grandes ramas: la IA cognitiva, que se centra en la creación de sistemas que pueden pensar y actuar como los humanos, y la IA conductual, que se centra en la creación de sistemas que pueden aprender y adaptarse a su entorno [1].

Existen también diversas especializaciones de la IA que se pueden utilizar para crear aplicaciones que beneficien a las personas, entre ellas la Inteligencia Artificial Generativa (IA generativa) y la Inteligencia Artificial Aumentada (IAA). La IA generativa se encuentra en la rama

cognitiva de la IA y utiliza algoritmos y redes neuronales avanzadas para aprender de información textual e imágenes para posteriormente generar contenido nuevo y único [2]. Este contenido puede ser texto, imágenes, música, sonido, o cualquier otro tipo de contenido creativo. Mientras que la IAA se encuentra en la rama conductual de la IA y utiliza la inteligencia artificial para mejorar las capacidades humanas. Por ejemplo, la IAA se utiliza para crear asistentes virtuales que pueden ayudar a las personas con tareas cotidianas, o para generar sistemas de aprendizaje automático que pueden ayudar a las personas a aprender nuevas habilidades [3].

Los modelos de lenguaje grande (LLM) son una forma de IA generativa que se utiliza para crear contenido nuevo a partir de datos existentes, mejorar las capacidades humanas, o ambas cosas. Estos LLM son la base de herramientas populares como ChatGPT y Google Bard (ahora Gemini) porque se emplean para entrenar a estos chatbots y modelos de lenguaje para que sean capaces de generar texto, traducir idiomas, escribir diferentes tipos de contenido creativo y responder a preguntas de forma informativa [4].

El uso combinado de estas tecnologías y herramientas de IA generativa tiene el potencial de revolucionar la forma en que se desarrollan y utilizan las aplicaciones inteligentes ya que pueden utilizarse para crear aplicaciones más personalizadas, eficientes y eficaces, en una amplia gama de áreas, por ejemplo para mejorar procesos de desarrollo de software, donde de forma particular, herramientas como el aprendizaje y la generación automática de código están demostrando tener un gran potencial para acelerar y mejorar el desarrollo de aplicaciones [5],[6].

La hipótesis central es que el uso de la IA generativa en el diseño, la codificación y las pruebas de aplicaciones, puede acelerar y mejorar la eficiencia del desarrollo en comparación con los métodos tradicionales. Esto se debe a que las herramientas de IA pueden automatizar tareas repetitivas, como la generación de código, el análisis de errores y la evaluación de rendimiento. Esto libera tiempo valioso para que los desarrolladores se concentren en aspectos más estratégicos, como la planificación del diseño, la resolución de problemas y la mejora de la experiencia del usuario, además de que permite iterar y probar nuevas funcionalidades con mayor velocidad [5].

El potencial del desarrollo de software aumentado por IA está respaldado por un cuerpo creciente de investigación y evidencia:
- Bruneliere et al. [5] proponen el marco AIDOaRt, que utiliza la IA para tareas de DevOps automatizadas y desarrollo continuo en sistemas ciberfísicos, demostrando su eficacia para mejorar la eficiencia y la calidad en el desarrollo de sistemas ciberfísicos.
- Pham et al. [7] destacan el potencial de las herramientas impulsadas por IA para automatizar los casos de prueba de software de extremo a extremo, ofreciendo importantes ahorros de tiempo y costo.
- Eager y Brunton [10] exploran el uso de la IA en la educación, demostrando su potencial para personalizar y mejorar las prácticas de enseñanza y aprendizaje, lo que puede traducirse bien en el dominio del desarrollo de software.
- Ozkaya [6] enfatiza el desarrollo de software con IAA (software aumentado) como la próxima frontera en el campo, prometiendo avances significativos en la velocidad, la calidad y la eficiencia general.

Mirando hacia el futuro, varias áreas clave ofrecen posibilidades para la investigación y el desarrollo continuo:
- Integración de la IA con herramientas y metodologías de desarrollo existentes: La integración transparente de las herramientas de IA con los flujos de trabajo de desarrollo existentes será crucial para la adopción generalizada [8].
- Estandarización y mejores prácticas: Establecer mejores prácticas y estándares para el desarrollo de software con IAA garantizará la consistencia y la calidad.
- Consideraciones éticas y desarrollo responsable: Abordar los posibles sesgos y preocupaciones éticas en torno al uso de la IA en el desarrollo de software [9].

El objetivo principal de esta investigación aplicada es demostrar los beneficios concretos de integrar herramientas de IA generativa durante el ciclo de desarrollo de la aplicación web, primordialmente en la generación de exámenes rápidos. De esta manera se pudieron cuantificar los ahorros de tiempo alcanzados, evaluar la calidad del código generado y medir la velocidad de iteración en comparación con un enfoque tradicional. Para ello, se describe cómo emplear la IA generativa para automatizar tareas repetitivas en las diferentes etapas de desarrollo de software, qué van desde la creación base del proyecto, el desarrollo de las interfaces web, la integración del servicio de preguntas, la lógica de selección de preguntas, el módulo de almacenamiento de datos y finalmente, la etapa de pruebas y corrección de errores.

MATERIAL Y MÉTODOS

El caso de estudio analizado consiste en utilizar herramientas de IA generativa para desarrollar una aplicación web, por ello, uno de los primeros pasos es realizar un proceso de exploración inicial donde se consideraron y evaluaron diversas herramientas de IA generativa como el popular chatGPT de OpenAI, Github Copilot, Google Bard (ahora Gemini), Claude.ai, entre otras. En la Tabla 1 se muestra un listado de plataformas que nos facilitan diversas tareas de generación de contenido y particularmente de código para desarrollo de software. Las mejores facilidades las ofrecieron Claude.ai y Google Bard por lo que fueron las seleccionadas para este trabajo.

Tabla 1. Herramientas de IA generativa para facilitar el desarrollo de software.

Plataforma	Empresa fabricante	Año de Liberación
Gemini (antes Bard)	Google	2023
Claude.ai	Anthropic	2022
Llama-2	Meta	2023
GitHub Copilot	OpenAI, Microsoft	2021
chatGPT	OpenAI	2022
OpenAI Codex	OpenAI	2021
TabNine	Codota	2021
DeepCode	Snyk (la adquirió en 2020)	2021
Amazon Bedrock	Amazon	2023

Fuente: Elaboración propia.

La aplicación web a desarrollar debe crear exámenes rápidos con base en temas seleccionados por un profesor y proporcionar las interfaces de software tanto para qué por un lado el profesor genere, asigne y evalúe el examen, y por otro el alumno pueda resolver el examen asignado. Para ello se definen dos perfiles de usuarios para la aplicación web: Usuario Profesor y Usuario Estudiante. Para cada uno de estos usuarios se desarrolló el código de programación en diferentes niveles considerando una arquitectura característica de 3 capas: Capa de Presentación (Front-End), Componentes de lógica de Negocio y Capa de Servicios (Back-End). En el diagrama a bloques de la Figura 1 se muestra cómo se integran los diferentes elementos, componentes y servicios, y las tareas qué realizan cada uno de ellos:

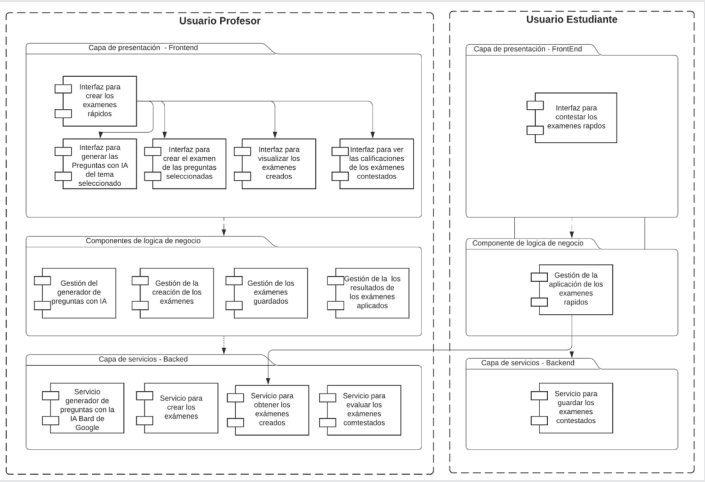

Figura 1. Diagrama a bloques de la aplicación web para exámenes rápidos con IA generativa.
Fuente: Elaboración propia.

De manera general, estas son las principales tareas qué se llevaron a cabo para el desarrollo de la aplicación web para generar exámenes con el uso de herramientas de IA generativa:

1. Programar el servicio generador de preguntas. Con el uso de Python y Flask, junto con la asistencia de la IA generativa, se implementa un servicio web para interactuar con la API de Google Bard y obtener preguntas sobre un tema específico, el cual se especificaba como parámetro. El resultado fue generar un banco de preguntas y respuestas para posteriormente integrar exámenes rápidos basados en esos temas.
2. Programar los servicios del BackEnd y de la Lógica del Negocio. Utilizando el método GET de las APIs con NodeJS, Express y Serialize, junto con la asistencia de iteraciones con IA generativa para incorporar los métodos adicionales. Posteriormente, se probaron utilizando Postman para almacenar tanto los exámenes como las respuestas proporcionadas por los alumnos durante la evaluación.
3. Programar las interfaces de la aplicación web de los usuarios de tipo Profesor y Estudiante. Se generó el código de la aplicación principal para que el profesor pueda crear los exámenes, guardarlos, y aplicarlos a los estudiantes. Se utilizó HTML, CSS, JavaScript con Material Design UI, y con la ayuda de la IA generativa se fue generando el código para incorporar la API, probarla y mejorarla hasta obtener la versión final.

Para una mejor comprensión sobre el desarrollo de la aplicación web para la generación de exámenes, las tareas principales se explicarán con mayor detalle.

Servicio de generación de preguntas según un tema seleccionado

Para obtener las preguntas de evaluación de forma dinámica según el tema seleccionado, se desarrolló un servicio API REST utilizando el framework Flask [11] de Python. Este servicio expone un endpoint GET que recibe en el cuerpo de la solicitud el tema sobre el cual se requiere generar preguntas. Internamente, se realiza una llamada a la API para Python de Google Bard [12] enviando una prompt diseñada para solicitar 10 preguntas de opción múltiple sobre el tema de interés.

Las respuestas retornadas por la IA de Google Bard vienen en formato de texto plano, por lo que se implementó un procesamiento con expresiones regulares y lógica de parseo, para extraer el enunciado de cada pregunta, tres opciones de respuesta posibles y la opción que es la correcta. De esta manera, a partir de un tema simple como “sumas y restas” se pueden obtener preguntas con opciones múltiples como:

Enunciado: ¿Cuánto es 5 + 3?
Respuestas posibles: a) 7 b) 8 c) 9
Respuesta correcta: b)

Adicionalmente, se convirtió la representación de datos de las preguntas a matrices de JavaScript para facilitar la manipulación y renderizada visual en la aplicación web destino. El servicio entrega un arreglo de estas

preguntas procesadas, encapsulando completamente la lógica de comunicación con la API de Google Bard y tratamiento de las respuestas.

Desarrollo de interfaces web

Las interfaces web son la cara visible de la aplicación web y conforman lo qué se conoce como la capa del Front-End y se tiene una para el usuario profesor y otra que sería para el usuario estudiante. Cada interfaz web consume o accede a través de una serie de componentes intermedios (de lógica del negocio) a los servicios que se programaron en la parte del Back-End. En la Figura 1 se aprecian todos los servicios de la capa del Backend.

Para el caso del servicio de definición de las preguntas de los exámenes, este fue generado en su totalidad utilizando modelos de lenguaje entrenados (LLM) de Anthropic [13] y el servicio de IA generativa Claude.ai [14]. La generación partió de una breve descripción en lenguaje natural de los requerimientos y funcionalidades necesarias. A partir de esto se genera de forma automática un proyecto completo con el siguiente diseño y características:

- Desarrollado en HTML, CSS, JavaScript y Material Design como bundle.
- Diseño adaptable a dispositivos móviles
- Página de inicio con campo para ingresar tema y botón de búsqueda.
- Vista de preguntas retornadas como tarjetas con opción múltiple.
- Botones para marcar una pregunta como seleccionada/deseleccionada.
- Confirmación con preguntas seleccionadas y formulario para ingresar nombre del examen.
- Integración con servicio API para guardar preguntas seleccionadas.

Tras la generación inicial se realizaron pequeños ajustes en estilos y nombres de variables.

De esta manera, la IA permitió completar en minutos la base del proyecto web que hubiera tomado días desarrollar manualmente y se agiliza en gran medida la implementación de interfaces responsivas y funciones de interacción web. En la Figura 2 se aprecia un ejemplo de una pantalla de la aplicación web destinada al usuario de tipo profesor, donde se hace la selección de preguntas para conformar un nuevo examen.

Almacenamiento de las preguntas y respuestas

Para persistir las preguntas de los exámenes generados, así como las respuestas y resultados de los alumnos al resolverlos, se desarrolló un servicio API REST con Node.js [15] y Express [16].

Este servicio expone los siguientes endpoints:

- GET /examenes recibe array de preguntas en formato matriz, nombre del examen, tema y guarda registro en base de datos. Devuelve ID único del examen generado
- GET /agregarresuelto recibe ID de examen, nombre del alumno, array de respuestas y puntaje obtenido. Registra resultado vinculado al examen.
- GET /verexamen/{id} retorna preguntas almacenadas para el ID de examen solicitado. Permite aplicar el examen guardado previamente.

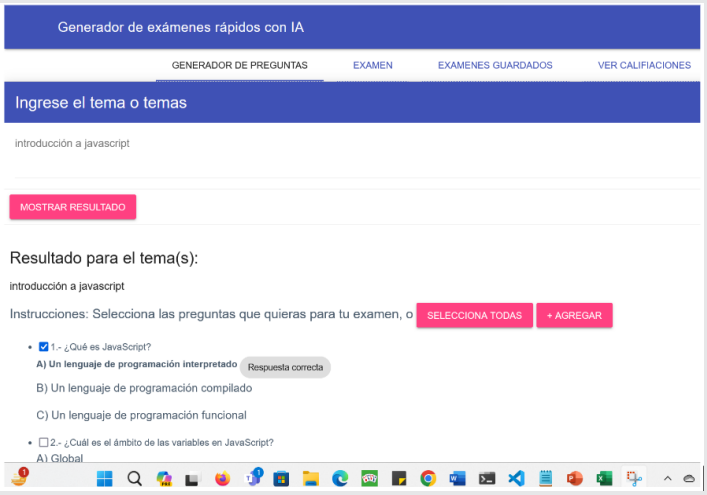

Figura 2: Ejemplo de interfaz de la aplicación web del profesor para selección de preguntas.
Fuente: Elaboración propia.

Para el almacenamiento y recuperación de información de los exámenes generados, aplicados y resueltos, se utilizó una base de datos relacional en MySQL Server y mediante el ORM Sequelize de NodeJS se mapean los registros de datos a objetos JavaScript para facilitar la manipulación de estos. De esta forma, todas las preguntas, exámenes, respuestas y calificaciones quedan almacenadas para su posterior consulta y procesamiento. La integración con este servicio permite a la aplicación web del usuario de tipo profesor guardar y reconstituir los exámenes generados.

RESULTADOS

Con el objetivo de demostrar los beneficios del uso de IA generativa, todo el código de los servicios y aplicación web fue generado de forma automática a partir de instrucciones en lenguaje natural, lo que comúnmente se le conoce como “prompts” o diálogos con la entidad inteligente, realizadas mediante la ayuda de Google Bard (ahora Gemini).

Cada una de las diferentes etapas del desarrollo de software tanto para la generación de los componentes de la lógica de negocio, así como el desarrollo de las interfaces web y los servicios de backend fueron programados utilizando los servicios de estas herramientas de IA generativa.

Esto generó los siguientes resultados en comparación con un desarrollo tradicional (Tabla 2). El uso de IA generativa permitió reducir el tiempo de desarrollo total de 34 horas a solo 4 horas con 5 minutos, lo que representa una

disminución de un 91.5% del tiempo destinado a las diferentes etapas del desarrollo de la aplicación (Figura 3).

Por otro lado, en la Tabla 3 se evidencia que el proceso de iteración y corrección de errores fue más ágil en comparación con el enfoque tradicional de depuración y corrección de la aplicación.

Tabla 2. Comparación de tiempos de desarrollo de la aplicación.

Actividad	desarrollo manual	desarrollo con IA generativa
Creación base del proyecto	3 horas	30 min
Desarrollo de interfaces web	10 horas	30 min
Integración del servicio de preguntas (Google Bard)	2 horas	1 hora
Lógica de selección de preguntas	5 horas	5 min
Módulo de almacenamiento	8 horas	1 hora
Pruebas y correcciones	6 horas	1 hora
Tiempo total	**34 horas**	**4 horas con 5 minutos**

Fuente: Elaboración propia.

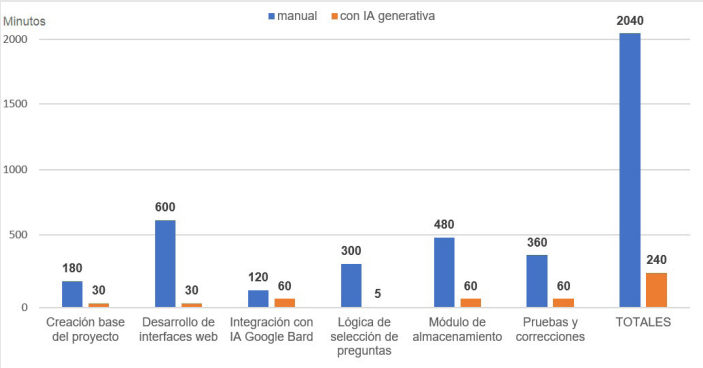

Figura 3: Comparativa de tiempos para tareas de desarrollo web: enfoque manual vs con IA generativa.
Fuente: Elaboración propia.

Asimismo, en la Figura 4 se puede apreciar esta diferencia significativa de ahorro en tiempos, dado que pasó de un total de 900 minutos a sólo 130 minutos lográndose un 85.6% de tiempo menos en el acumulado de los diferentes casos de corrección de errores.

Cabe mencionar que el cálculo de los tiempos para llevar a cabo las diversas actividades de desarrollo web, así como de las tareas de depuración y corrección de errores, fueron estimados basados en el juicio de un programador senior con amplia experiencia por haber participado en el desarrollo de una gran cantidad de proyectos web similares. Así mismo, fue este mismo programador experto quien llevó a cabo las tareas de diseño y programación de la aplicación web auxiliado de las herramientas de IA generativa.

Tabla 3. Comparación de tiempos de corrección de errores: Manual vs IA Generativa.

Tipo de error	Corrección Manual	Corrección con IA generativa
Formato incorrecto de los reactivos de las preguntas	240 minutos (4 horas) Requiere cambios en múltiples partes del código.	5 minutos Requiere únicamente modificar el prompt de generación de código
Formato incorrecto de código de almacenamiento de datos.	300 minutos (5 horas) Requiere cambios en múltiples partes del código	5 minutos Requiere únicamente modificar el prompt de generación para indicar revise el código y se corrijan los errores
Formato incorrecto en código de aplicación para la visualización web	360 minutos (6 horas) Requiere cambios en múltiples partes del código	120 minutos (2 horas) Requiere modificar el prompt de generación de código para indicar con precisión las adecuaciones en la interfaz web para el correcto acomodo, colores, desplegado de datos, llamadas a APIs de almacenamiento y obtención de preguntas, así como ligeros ajustes manuales.
Tiempo total	**900 minutos (15 horas)**	**130 minutos (2 horas y 10 min)**

Fuente: Elaboración propia.

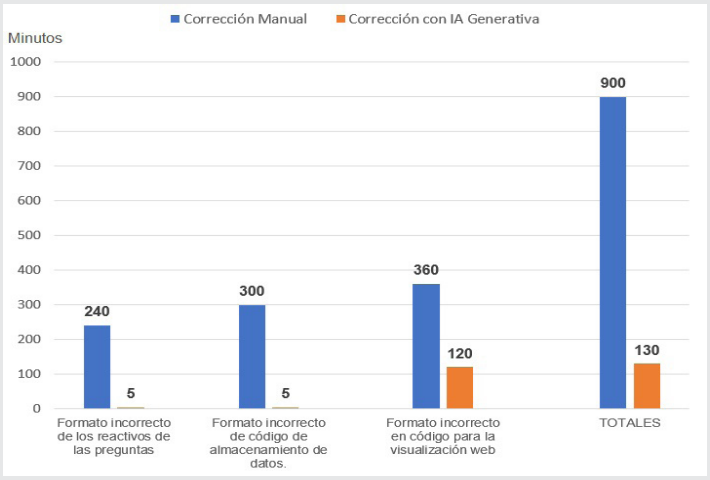

Figura 4: Comparativa de tiempos para tareas de corrección de errores: enfoque manual vs con IA generativa.
Fuente: Elaboración propia.

CONCLUSIONES

Los resultados de esta investigación aplicada indican que la implementación de herramientas de IA generativa en el desarrollo de la aplicación web para la generación de exámenes resultó en mejoras sustanciales en la velocidad y eficiencia del proceso. Esto se logró al automatizar las tareas de programación y simplificar considerablemente la detección y corrección de errores, todo ello sin requerir modificaciones extensas en distintas secciones del código.

En particular, el tiempo de desarrollo total se redujo en un 91.5%, pasando de 34 horas en un enfoque tradicional a solo 4 horas con IA generativa. Esto se debe a que la IA automatizó tareas repetitivas, como la generación de código, el diseño de interfaces y la integración de servicios.

Además, la iteración y corrección de errores resultó 85.6% más rápida. Esto se debe a que la IA permite modificar el código de forma rápida y sencilla, sin necesidad de realizar cambios en múltiples partes del código.

En general, los resultados de esta investigación sugieren que el uso de IA generativa puede ser una herramienta valiosa para los desarrolladores de aplicaciones web. Al automatizar tareas repetitivas, la IA puede ayudar a los desarrolladores a concentrarse en aspectos más estratégicos del proceso de desarrollo, lo que puede conducir a un desarrollo más rápido, eficiente y de mayor calidad.

Con la experiencia obtenida y con los resultados de esta investigación, se pueden hacer las siguientes recomendaciones para el uso de IA generativa en el desarrollo de aplicaciones web:

- La IA generativa puede ser una herramienta valiosa para automatizar tareas repetitivas, como la generación de código, el diseño de interfaces y la integración de servicios.
- Los desarrolladores deben estar familiarizados con las limitaciones de la IA generativa, como la posibilidad de generar código incorrecto o ineficiente.
- Los desarrolladores deben realizar pruebas exhaustivas del código generado por la IA para garantizar su calidad.

En el futuro, se espera que las herramientas de IA generativa continúen mejorando, lo que podría llevar a mejorar en la velocidad y eficiencia del desarrollo de aplicaciones web que tenemos actualmente.

BIBLIOGRAFÍA

[1] Ruiz-Rojas, LI., Acosta-Vargas, P., De-Moreta-Llovet, J., y Gonzalez-Rodriguez, M. (2023). Empowering Education with Generative Artificial Intelligence Tools: Approach with an Instructional Design Matrix. Sustainability, 15(15):11524. DOI: 10.3390/su151511524.

[2] Kar, S., Roy, C., Das, M., Mullick, S., & Saha, R. (2023). AI Horizons: Unveiling the Future of Generative Intelligence. International Journal of Advanced Research in Science, Communication and Technology. https://doi.org/10.48175/ijarsct-12969.

[3] Yu, H. and Guo, Y. (2023). Generative artificial intelligence empowers educational reform: current status, issues, and prospects. Front. Educ. 8:1183162. DOI: 10.3389/feduc.2023.1183162.

[4] Höppner, T., Streatfeild, L. (2023). ChatGPT, Bard & Co.: An Introduction to AI for Competition and Regulatory Lawyers. En SSRN: 9 Hausfeld Competition Bulletin (1, 2003, article 1). DOI: 10.2139/ssrn.4371681. Disponible en: https://ssrn.com/abstract=4371681.

[5] Bruneliere, H., Muttillo, V., Eramo, R., Berardinelli, L., Gomez, A., Bagnato, A., ... & Cicchetti, A. (2022). AIDOaRt: AI-augmented Automation for DevOps, a model-based framework for continuous development in Cyber–Physical Systems. En: Microprocessors and Microsystems, 94, 104672. DOI: 10.1016/j.micpro.2022.104672.

[6] Ozkaya, I. (2023). The next frontier in software development: AI-augmented software development processes. En: IEEE Software, 40(4), 4-9. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10176194.

[7] Pham, P., Nguyen, V., & Nguyen, T. (2022, October). A Review of AI-augmented End-to-End Test Automation Tools. En: Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering (pp. 1-4). https://dl.acm.org/doi/abs/10.1145/3551349.3563240.

[8] Russo, D. (2023). Navigating the Complexity of Generative AI Adoption in Software Engineering. ACM Transactions on Software Engineering and Methodology, Vol 33, Issue 5. Article No.: 135, pp: 1-50. DOI: https://doi.org/10.1145/3652154.

[9] Kuck, K. (2023). Generative Artificial Intelligence: A Double-Edged Sword. 2023 World Engineering Education Forum - Global Engineering Deans Council (WEEF-GEDC), 1-10. DOI: 10.1109/WEEF-GEDC59520.2023.10343638.

[10] Eager, B., & Brunton, R. (2023). Prompting higher education towards AI-augmented teaching and learning practice. En: Journal of University Teaching & Learning Practice, 20(5), 02. DOI: 10.53761/1.20.5.02

[11] The Pallet Projects. (2023). Welcome to Flask Flask Documentation (3.0.x). En Palletsprojects.

com. Disponible en: https://flask.palletsprojects.com/en/3.0.x/.

[12] PyPI.org. (2023). Google Bard API para Python. Sitio oficial para descarga de la librería para Python en PyPI. Disponible en: https://pypi.org/project/bardapi/.

[13] Yiu, E. et al. (2023). Transmission Versus Truth, Imitation Versus Innovation: What Children Can Do That Large Language and Language-and-Vision Models Cannot (Yet). En: Perspectives on Psychological Science. https://journals.sagepub.com/doi/full/10.1177/17456916231201401.

[14] Anthropic. (2023). Claude - Asistente de IA Generativa. En línea. Disponible en: http://www.claude.ai.

[15] OpenJS-Foundation (2023) Node.js. Disponible en: https://nodejs.org/en.

[16] ExpressJS.com (2017) Express JS -Node.js web application framework. Sitio official. Disponible en: https://expressjs.com/