**UNIVERSIDAD MAYOR REAL Y PONTIFICIA DE SAN FRANCISCO XAVIER DE CHUQUISACA**

FACULTAD DE CIENCIAS Y TECNOLOGIA

**CARRERA DE INGENIERÍA EN CIENCIAS DE LA COMPUTACION**

PERFIL DE PROYECTO DE GRADO



**“MODELO GENERADOR DE ESTILOS CSS EN BASE A ETIQUETAS HTML PARA DESARROLLADORES WEB”**

TUMIRI HUANCA ALEX

Sucre, 11 de Abril del 2023

1. **TITULO**

**MODELO GENERADOR DE ESTILOS CSS EN BASE A ETIQUETAS HTML PARA DESARROLLADORES WEB**

1. **INTRODUCCIÓN Y/O ANTECEDENTES**

En la actualidad el lenguaje de diseño y programación CSS tiene muchas ventajas, características, librerías, API’s y es utilizado en todos los navegadores y plataformas web del mundo, siendo así un estándar en el desarrollo de interfaces de usuario y en el desarrollo de aplicaciones web, el problema no reside en que la tecnología CSS sea una tecnología ineficiente para el desarrollo de software, o que tenga problemas de compatibilidad, o una ruta de aprendizaje compleja, todo lo contrario.

Los desarrolladores de software sobre todo especializados en la implementación de diseños e interfaces gráficas durante mucho tiempo han intentado hacer su trabajo más eficiente y productivo en esta exhaustiva búsqueda de eficiencia y productividad los mismos programadores he ingenieros de software, han desarrollado diversas técnicas y herramientas las cuales facilitan su trabajo a la hora de crear software.

Existen muchas librerías y herramientas de CSS que pueden ayudarnos a generar plantillas y diseños web, estas herramientas aumentan la productividad y eficiencia siendo algunas de estas Bootstrap, Materialize CSS, Tailwind CSS, Semantic UI, etc. Sin embargo también aumentan la curva de aprendizaje y el tiempo para aprender las diferentes herramientas que constantemente se van actualizando día a día, sin mencionar que el proceso de desarrollo de estilos CSS no solo depende, que el desarrollador de software conozca muy bien la tecnología y los conceptos de la misma si no que tenga como vulgarmente se dice “un muy buen gusto”, el cual es importante para tener éxito con el desarrollo de la Interfaz de Usuario (UI) y Experiencia de usuario (UX), mientras que uno hace referencia a la experiencia y sensación del usuario, el otro está dirigido hacia un lado más racional de la navegación, también se deben considerar otros desafíos mayores como el buen diseño.

Según el sitio web BuiltWith, que rastrea el uso de tecnologías web en todo el mundo, Bootstrap es la librería de CSS más utilizada en la actualidad, con un 16% de participación de mercado. Le sigue Materialize CSS con un 0,9%, Foundation con un 0,8%, Tailwind CSS con un 0,6%, Semantic UI con un 0,4%, y Bulma con un 0,2%.

Bootstrap es la librería de CSS más utilizada debido a varias razones:

* Amplia comunidad de usuarios y soporte: Bootstrap fue desarrollado por Twitter, y ha sido mantenido y mejorado por una gran comunidad de desarrolladores. Esta comunidad ha creado una gran cantidad de recursos, documentación, tutoriales y foros para ayudar a los usuarios de Bootstrap en todo el mundo.
* Diseño y funcionalidad responsivos: Bootstrap se enfoca en ofrecer una experiencia de usuario consistente y fluida en dispositivos móviles, tabletas y computadoras de escritorio, lo que lo hace muy popular entre los desarrolladores que buscan crear diseños responsivos y adaptativos.
* Conjunto completo de componentes y estilos predefinidos: Bootstrap ofrece un conjunto completo de componentes y estilos predefinidos que permiten a los desarrolladores crear diseños y aplicaciones web de alta calidad de manera rápida y sencilla. Esto ayuda a reducir el tiempo de desarrollo y a mejorar la eficiencia.
* Personalización: A pesar de que Bootstrap ofrece una gran cantidad de componentes y estilos predefinidos, también permite personalizar el aspecto y la funcionalidad de los mismos. Esto hace que Bootstrap sea muy versátil y pueda ser utilizado en proyectos de cualquier tamaño y complejidad.

Bootstrap es muy popular debido a su amplia comunidad de usuarios, su diseño y funcionalidad responsivos, su conjunto completo de componentes y estilos predefinidos, y su capacidad de personalización. Estas características hacen que sea una herramienta muy útil para los desarrolladores que buscan crear diseños y aplicaciones web modernas y adaptativas.

Las herramientas anteriormente mencionadas también brindan un diseño estándar el cual es fácilmente reconocible siendo este el caso de Bootstrap y no es de muy buen gusto por lo cual no agrega valor al diseño de la interfaz.

1. **SITUACION PROBLEMÁTICA**

Existe poca productividad por parte de los ingenieros de software a la hora de desarrollar interfaces de usuario, diseño y estilos con el lenguaje de diseño gráfico CSS. Debido a la complejidad de la implementación de un diseño web efectivo y la gran cantidad de líneas de código que esto puede requerir, muchos desarrolladores pueden enfrentarse a un proceso de desarrollo lento y tedioso este proceso consiste en escribir código repetitivo constantemente.

Estas librerías, frameworks y APIs anteriormente mencionadas comúnmente carecen de flexibilidad a la hora de desarrollar ofreciendo muchas veces estándares de diseños con colores predeterminados que no son de buen gusto y que para lograr el resultado deseado por el desarrollador se debe escribir más código e incluso tener más conocimientos sobre la librería que se esté utilizando, muchas veces el trabajo solo se incrementa tanto en líneas de código como en tiempo de desarrollo, reduciendo la productividad y aumentando la complejidad del desarrollo y obviamente aumentando también el tiempo de desarrollo, es cierto que el uso excesivo de estas herramientas puede llevar a una falta de flexibilidad y creatividad en el diseño y la implementación de una aplicación o sitio web. Por lo tanto, es importante que los desarrolladores encuentren el equilibrio adecuado entre el uso de estas herramientas y la implementación de soluciones personalizadas y creativas.

En algunas situaciones, el desarrollador de software puede tener la capacidad y conocimiento técnico para desarrollar interfaces de muy buena calidad con CSS, pero resulta que carece del buen gusto y habilidades de diseño necesarias para crear una interfaz de usuario atractiva y funcional. El buen gusto y la habilidad de diseño son esenciales para crear una experiencia de usuario efectiva y agradable, y esto a menudo requiere de una combinación de habilidades técnicas y creativas.

Si un desarrollador de software carece de estas habilidades, puede resultar en una interfaz de usuario poco atractiva, difícil de usar y menos efectiva.

Por lo general las diferentes herramientas para CSS que existen en la actualidad son muy diversas y muchas veces obtienes con estas el mismo resultado, también tienen una curva de aprendizaje compleja sin mencionar que debes conocer muy bien los conceptos y sintaxis del lenguaje CSS, por esta razón no son alternativas factibles para desarrolladores sin experiencia con CSS o que se estén iniciando en el desarrollo de interfaces de usuario.

Existen varias herramientas de CSS que pueden tener una curva de aprendizaje larga e ineficiente para algunos desarrolladores, dependiendo de su nivel de experiencia y habilidades técnicas. Algunas de estas herramientas son:

* Sass y Less: Sass (Syntactically Awesome Style Sheets) y Less (Leaner CSS) son preprocesadores de CSS que permiten a los desarrolladores escribir CSS de manera más eficiente y organizada. Sin embargo, su sintaxis y estructura pueden ser más complejas que el CSS convencional, lo que puede requerir tiempo y esfuerzo para aprender.
* CSS Grid y Flexbox: CSS Grid y Flexbox son herramientas de diseño de diseño web que permiten a los desarrolladores crear diseños de página más complejos y flexibles. Sin embargo, la curva de aprendizaje para estas herramientas puede ser larga y requiere un conocimiento detallado de las propiedades de CSS y cómo se aplican.
* Animaciones CSS: Las animaciones CSS permiten a los desarrolladores agregar efectos y animaciones a las páginas web, pero pueden requerir un conocimiento detallado de las propiedades de animación y transición de CSS.
* CSS preprocessors: Además de Sass y Less, existen otros preprocesadores de CSS como Stylus y PostCSS. Cada uno de estos preprocesadores tiene su propia sintaxis y estructura, lo que puede requerir un tiempo adicional para aprender.

Cada herramienta de CSS también tiene sus propios pros y contras, y la elección de la herramienta adecuada dependerá de las necesidades y objetivos específicos del proyecto. A continuación, se presentan algunos de los pros y contras de las herramientas de CSS más populares:

* Bootstrap:

Pros: Bootstrap es una de las herramientas de CSS más populares y ampliamente utilizadas, lo que significa que hay una gran comunidad de desarrolladores y recursos disponibles en línea. Bootstrap es fácil de usar y tiene una curva de aprendizaje corta, lo que lo hace ideal para proyectos de desarrollo web rápidos. Además, Bootstrap ofrece una gran cantidad de componentes, como botones, formularios y carruseles, que se pueden implementar fácilmente en cualquier proyecto.

Contras: Debido a su popularidad, Bootstrap a menudo se utiliza para crear sitios web que se ven muy similares. Además, Bootstrap tiene una gran cantidad de clases de estilo, lo que puede aumentar el tamaño de los archivos CSS y ralentizar el rendimiento de la página.

* Materialize CSS:

Pros: Materialize CSS es una herramienta de CSS basada en el diseño de material de Google, lo que significa que ofrece una estética moderna y atractiva. Materialize CSS es fácil de usar y tiene una amplia variedad de componentes y funciones, lo que hace que sea ideal para proyectos de desarrollo web rápidos. Además, Materialize CSS tiene una buena documentación y una gran comunidad de desarrolladores.

Contras: Materialize CSS es menos conocido que otras herramientas de CSS, lo que puede dificultar la búsqueda de recursos y apoyo en línea haciendo que su desarrollo demore mucho más tiempo. Además, al igual que Bootstrap, Materialize CSS tiene una gran cantidad de clases de estilo, lo que puede aumentar el tamaño de los archivos CSS y ralentizar el rendimiento de la página.

* Tailwind CSS:

Pros: Tailwind CSS es una herramienta de CSS altamente personalizable y modular, lo que permite a los desarrolladores crear diseños únicos y específicos para sus proyectos. Tailwind CSS utiliza clases de estilo que describen exactamente lo que hacen, lo que facilita la lectura y el mantenimiento del código. Además, Tailwind CSS tiene una buena documentación y una gran comunidad de desarrolladores.

Contras: La curva de aprendizaje de Tailwind CSS puede ser larga, ya que los desarrolladores deben aprender las clases de estilo específicas que se utilizan en esta herramienta. Además, Tailwind CSS puede resultar en una gran cantidad de clases de estilo en el código, lo que puede dificultar la lectura y el mantenimiento.

* Semantic UI:

Pros: Semantic UI es una herramienta de CSS altamente personalizable que utiliza un lenguaje claro y semántico para describir los componentes de la interfaz de usuario. Semantic UI es fácil de usar y tiene una gran cantidad de componentes, lo que facilita la creación de diseños complejos. Además, Semantic UI tiene una buena documentación y una gran comunidad de desarrolladores.

Contras: Semantic UI puede ser menos conocido que otras herramientas de CSS, lo que puede dificultar la búsqueda de recursos y apoyo en línea. Además, al igual que con otras herramientas de CSS, Semantic UI puede resultar en una gran cantidad de clases de estilo en el código, lo que puede dificultar la lectura y el mantenimiento.

El desarrollo de software con la tecnología CSS como con otras tecnologías tiende a ser repetitivo, poco intuitivo, ineficiente y en su mayoría reduce la productividad del ingeniero de software a causa de las malas prácticas, desconocimiento de la tecnología CSS y otros factores que hacen del ingeniero de software sea improductivo a la hora de desarrollar software de buena calidad.

.

1. **PROBLEMA CENTRAL**

La complejidad en la implementación de diseños web efectivos con CSS y la gran cantidad de líneas de código necesarias disminuyen la productividad en el desarrollo de interfaces de usuario, diseño y estilos. Aunque se han utilizado librerías, frameworks y APIs de CSS estándar predefinidas para solucionarlo, estas limitan la flexibilidad y creatividad, lo que aumenta la complejidad del proceso y el tiempo requerido. Además, la falta de habilidades de diseño en los desarrolladores de software puede generar interfaces poco atractivas e ineficaces, lo que resulta en un problema crítico en el desarrollo de interfaces. En este sentido, la complejidad de aprendizaje de las herramientas de CSS puede ser un obstáculo para aquellos que se inician en el desarrollo de interfaces de usuario.

1. **ABORDAJE DE SOLUCION**

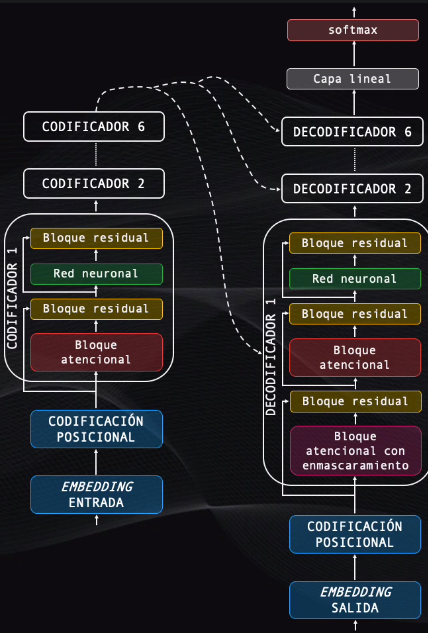
El desarrollo de un modelo de inteligencia artificial generará estilos CSS predefinidos y con características específicas, reconociendo el código HTML introducido por el desarrollador de software en una aplicación web intuitiva y fácil de usar que tendrá como motor de generación de código el modelo anteriormente mencionado, para reducir de manera notable el tiempo de desarrollo, aumentando la productividad y eficiencia del ingeniero de software, evitando que el ingeniero en cuestión escriba código repetitivo reduciendo así el tiempo de desarrollo con el cual el desarrollador de software podrá gestionar de manera más productiva su tiempo para optimizar y desarrollar un código de buena calidad y por consiguiente un mejor producto.

Se recopilará un gran volumen de datos de estilos CSS para el preprocesamiento de las diferentes características de la sintaxis del lenguaje CSS y clasificación de datos obtenidos para crear dos diferentes sets de datos que serán utilizados para el entrenamiento y pruebas de nuestro modelo generador de código CSS.

Como alternativa tecnológica para el desarrollo del modelo generador de código CSS se considera, luego de un análisis profundo e investigación sobre el estado actual del desarrollo de modelos de inteligencia artificial, es difícil ignorar el avance que se realizó en los últimos años en el área del Procesamiento del lenguaje natural (NLP) es una tecnología que brinda a las computadoras la capacidad interpretar, manipular, y comprender el lenguaje humano esta tecnología será de vital utilidad para reconocer, clasificar y extraer texto para que el modelo pueda aprender la sintaxis del lenguaje de programación CSS.

La arquitectura Transformer de redes neuronales será utilizada para el entrenamiento y optimización del modelo, esta arquitectura brinda la innovadora técnica del procesamiento en paralelo de la secuencia de parámetros introducidos a diferencia de las redes neuronales recurrentes donde lo parámetros se procesan de uno a uno de forma serial, a continuación, un gráfico de la arquitectura Transformer.

**Diagrama de la Arquitectura Transformers:**



**Fuente:** CODIFICANDOBITS Redes Transformer (... o el fin de las Redes Recurrentes). [En línea]. Disponible en Internet: <https://www.codificandobits.com/blog/redes-transformer/ > [Consulta: 03 de marzo 2023].

En el anterior grafico vemos todos los módulos de codificación y decodificación, también se utiliza las técnicas de embedding y los bloques atencionales toda esta arquitectura es muy eficiente e innovadora en el desarrollo de modelos de procesamiento de lenguaje natural y generación de texto específicamente, optimizando y ajustando la arquitectura Transformer a nuestro caso en específico obtendremos como resultado un eficiente y potente modelo de generación de código CSS

1. **OBJETIVO GENERAL**

Desarrollar un modelo de inteligencia artificial para la generación de código CSS predefinido y con buen gusto que no solo facilite el trabajo de los desarrolladores de software, sino que también reduzca el tiempo de desarrollo y aumente la productividad a la hora generar rápidamente diseños predefinidos y personalizables sin tener que escribir manualmente cada línea de código. En última instancia, el uso del generador de código CSS puede ayudar a reducir la complejidad del desarrollo y permitir a los desarrolladores centrarse en la creatividad y la innovación en lugar de tener que dedicar tiempo a tareas repetitivas.

1. **OBJETIVOS ESPECIFICOS**

* Realizar un modelo de inteligencia artificial que genere código CSS con un 90% de efectividad.
* Desarrollar una aplicación web para generar código CSS en base a etiquetas HTML.
* Reducir el tiempo de desarrollo del ingeniero de software.
* Reducir el desarrollo de código repetitivo.
* Creación de una Herramienta para el desarrollo de software ágil.

1. **JUSTIFICACIÓN**
   1. **Justificación Económica**

El desarrollo de un modelo de inteligencia artificial para la generación de código CSS predefinido y con buen gusto puede tener un impacto positivo en la economía de la empresa. Al reducir el tiempo de desarrollo y aumentar la productividad, los desarrolladores de software pueden trabajar más eficientemente y dedicar más tiempo a tareas de mayor valor añadido. Esto puede permitir que la empresa pueda lanzar productos y servicios al mercado más rápido, lo que puede aumentar su competitividad y su cuota de mercado. Además, al reducir la complejidad del desarrollo, el generador de código CSS puede ayudar a reducir los errores de codificación y los costos asociados a su corrección. En última instancia, el uso del generador de código CSS puede aumentar la eficiencia y reducir los costos, lo que puede traducirse en mayores beneficios para la empresa y un retorno de inversión significativo en el corto y largo plazo.

* 1. **Justificación Tecnológica**

En el desarrollo del proyecto, se abarcarán los conocimientos de:

Entrenamiento de modelos de Inteligencia artificial generadores de texto aplicando NLP (Procesamiento del Lenguaje Natural), tanto en el software como en la generación de código CSS.

La arquitectura Transformers de redes neuronales que se empleara para la optimización y entrenamiento de nuestro modelo.

Sistema de generación de código CSS mediante la entrada de código HTML.

**MARCO TEÓRICO**

El lenguaje de hojas de estilo en cascada (CSS) es un componente clave en el desarrollo de interfaces de usuario en la web. CSS es una herramienta poderosa que permite a los desarrolladores crear diseños personalizados y atractivos para mejorar la experiencia del usuario en la web. Además, CSS es importante para la accesibilidad web, ya que permite a los desarrolladores crear diseños que sean legibles y fáciles de usar para todos los usuarios.

En la actualidad, existen varias librerías, frameworks y APIs de CSS disponibles en el mercado que pueden ayudar en el desarrollo de interfaces de usuario. Estas soluciones predefinidas se han utilizado para abordar la complejidad de la implementación de diseños web efectivos. Sin embargo, estas soluciones pueden limitar la creatividad y la flexibilidad en el diseño de interfaces de usuario, lo que puede afectar negativamente la experiencia del usuario.

La creatividad y la flexibilidad son fundamentales en el diseño de interfaces de usuario para mejorar la experiencia del usuario. Las soluciones predefinidas de CSS pueden limitar la creatividad y la flexibilidad en el diseño de interfaces de usuario, lo que puede resultar en diseños repetitivos e ineficaces. Es importante que los desarrolladores tengan la libertad de experimentar y crear diseños únicos y personalizados para mejorar la experiencia del usuario.

Es fundamental que los desarrolladores de software tengan habilidades de diseño para crear interfaces de usuario efectivas y atractivas. La falta de habilidades de diseño puede generar interfaces poco atractivas e ineficaces, lo que resulta en una experiencia de usuario deficiente.

Los desarrolladores deben tener habilidades de diseño para poder crear diseños que sean intuitivos y fáciles de usar para todos los usuarios. Las herramientas de CSS pueden tener una curva de aprendizaje compleja, lo que puede ser un obstáculo para aquellos que se inician en el desarrollo de interfaces de usuario.

Es importante que las herramientas de CSS sean intuitivas y fáciles de usar para facilitar el proceso de desarrollo de interfaces de usuario. Además, las herramientas de CSS deben ser capaces de adaptarse a las necesidades cambiantes de los desarrolladores para mejorar su productividad y eficiencia en el proceso de desarrollo de interfaces de usuario.

La historia de la inteligencia artificial (IA) se remonta a la década de 1940, cuando se empezó a explorar la posibilidad de crear máquinas capaces de simular el pensamiento humano. El término "inteligencia artificial" fue acuñado por John McCarthy en 1956, durante la Conferencia de Dartmouth.

En las décadas siguientes, se desarrollaron varios enfoques para la IA, incluyendo el uso de lógica, la simulación de redes neuronales y el aprendizaje automático. En 1959, Arthur Samuel desarrolló un programa que podía jugar al ajedrez, lo que se considera uno de los primeros ejemplos de aprendizaje automático.

En las décadas de 1960 y 1970, se produjeron avances significativos en la IA, incluyendo el desarrollo del lenguaje de programación LISP y la creación de sistemas expertos, que utilizaban bases de conocimientos para tomar decisiones. Sin embargo, también se produjo un período de desilusión en la IA, conocido como "invierno de la IA", debido a la falta de progreso y los costos elevados de los proyectos de investigación.

En las décadas de 1980 y 1990, se produjo un resurgimiento de la IA, impulsado por avances en el aprendizaje automático y la creación de algoritmos más eficientes. En 1997, Deep Blue de IBM venció al campeón mundial de ajedrez Garry Kasparov, lo que se considera un hito en la historia de la IA. En las últimas décadas, la IA ha experimentado un rápido avance, impulsado por el aumento de la potencia de procesamiento y la disponibilidad de grandes conjuntos de datos.

La IA se ha aplicado en una amplia gama de campos, desde la visión por computadora hasta la traducción automática y la robótica. A medida que la IA se vuelve más avanzada, también ha surgido preocupación sobre su impacto en la sociedad y la necesidad de un enfoque ético en su desarrollo y aplicación.

Los modelos de inteligencia artificial generadores de texto que usan NLP (procesamiento del lenguaje natural, por sus siglas en inglés) son una rama de la inteligencia artificial que se enfoca en la creación de sistemas capaces de comprender y generar texto de manera similar a como lo haría un ser humano.

Estos modelos utilizan técnicas de aprendizaje profundo, que les permiten analizar grandes cantidades de datos y aprender patrones y relaciones en el lenguaje natural. Uno de los modelos más conocidos en esta área es el modelo de lenguaje GPT (Generative Pre-trained Transformer, por sus siglas en inglés), desarrollado por OpenAI. Este modelo utiliza una arquitectura de red neuronal llamada Transformer, que es capaz de procesar grandes cantidades de texto y generar texto de alta calidad.

Los modelos de inteligencia artificial generadores de texto que usan NLP tienen una amplia gama de aplicaciones, desde la generación de contenido de marketing y publicidad hasta la creación de diálogos de chatbot y la producción de textos para la literatura y la poesía y también aplicado a nuestro caso código. Estos modelos también se están utilizando en la creación de noticias y artículos de opinión automatizados, lo que ha generado un debate sobre la ética y la precisión en la generación de contenido generado por inteligencia artificial.

Aunque los modelos de inteligencia artificial generadores de texto que usan NLP han avanzado significativamente en los últimos años, todavía hay muchos desafíos por resolver. Por ejemplo, estos modelos a menudo tienen dificultades para generar texto coherente y relevante en contextos específicos, y pueden generar textos con prejuicios y estereotipos culturales. Además, también hay preocupaciones entorno a la capacidad de estos modelos para crear textos falsificados y engañosos, lo que puede tener consecuencias importantes en áreas como la política y la seguridad nacional. Por lo tanto, es importante que se sigan investigando en esta área y se desarrollen técnicas para abordar estos problemas éticos y prácticos asociados con la generación de texto por inteligencia artificial.

Los modelos generadores de código son una tecnología de inteligencia artificial que tiene como objetivo automatizar la tarea de escribir código. Estos modelos utilizan técnicas de aprendizaje automático para analizar grandes conjuntos de datos de código y generar código de forma autónoma.

Si bien esta tecnología puede tener ventajas en términos de eficiencia y productividad, también plantea desafíos y preocupaciones significativas en términos de su impacto en la sociedad actual. En primer lugar, la automatización de la tarea de escribir código podría tener un impacto en el mercado laboral de los programadores y desarrolladores de software.

Si bien esta tecnología podría permitir a los desarrolladores completar más tareas en menos tiempo, también podría reducir la demanda de mano de obra en la industria del software. Esto podría tener consecuencias significativas para los trabajadores en el campo, así como para la economía en general. Además, existe la preocupación de que los modelos generadores de código puedan aumentar la brecha de habilidades en la industria del software.

Si bien esta tecnología puede ayudar a los desarrolladores a crear aplicaciones más rápidamente, también podría requerir un nivel de conocimiento técnico avanzado para entrenar y utilizar estos modelos de manera efectiva. Esto podría exacerbar las desigualdades existentes en la industria del software y limitar el acceso a esta tecnología para aquellos que no tienen los recursos o habilidades necesarias.

También existe el riesgo de que los modelos generadores de código puedan producir código defectuoso o incluso malicioso. Si bien la automatización de la tarea de escribir código puede mejorar la eficiencia, también podría reducir la calidad del código generado si los modelos no se entrenan adecuadamente o si se les proporcionan datos inadecuados. Esto podría tener consecuencias graves en términos de seguridad y estabilidad de las aplicaciones de software.

En conclusión, los modelos generadores de código tienen el potencial de mejorar significativamente la eficiencia y productividad en la industria del software, pero también plantean preocupaciones y desafíos importantes en términos de su impacto en la sociedad actual. Es importante que se tomen medidas para garantizar que esta tecnología se utilice de manera responsable y ética para maximizar sus beneficios y minimizar sus riesgos y consecuencias negativas.

**CAPITULO III: INGENIERÍA DEL PROYECTO**

**CONCLUSIONES**

**RECOMENDACIONES**

**BIBLIOGRAFÍA**

**ANEXOS**

**PLANOS**

1. **CRONOGRAMA**
2. **COSTO DE ELABORACION DEL PROYECTO**
3. **BIBLIOGRAFIA**