

ELABORACIÓN DE PROGRAMAS EN PYTHON MEDIANTE LA INTELIGENCIA ARTIFICIAL

Luis Antonio Bertorelli Arias
Programa de Ingeniería Mecánica
Escuela Colombiana de Ingeniería Julio Garavito
luis.bertorelli@mail.escuelaing.edu.co

Resumen — Este informe se hace con el fin de observar y evidenciar el cómo la inteligencia artificial utilizada de la manera adecuada puede facilitar todo tipo de trabajo académico, o incluso cualquier aspecto de la vida rutinaria, este informe cumple con el propósito de evidenciar como la inteligencia artificial, en este caso ChatGPT, puede crear programas de gran aporte académico o rutinario.

Palabras Claves — *Inteligencia Artificial, ChatGPT, Programas.*

I. INTRODUCCIÓN

En este informe se deja en evidencia como la inteligencia artificial puede llegar a crear programas de cualquier tipo, y demostrar que con las peticiones y preguntas adecuadas esta inteligencia puede resolver todo tipo de problema, en esta ocasión se busca crear dos códigos que se ven en la cotidianidad sin entender realmente el trabajo interno que manejan; el primero es un sistema de cambio de contraseña y el segundo es el sistema de una máquina expendedora adicionalmente, se pretende demostrar que la inteligencia artificial puede ser de gran utilidad para muchos problemas.

II. OBJETIVO

El objetivo de este informe es adquirir destrezas y habilidades en el manejo de la inteligencia artificial como ayuda y soporte para diferentes proyectos, comprender e interpretar los resultados obtenidos por la inteligencia artificial y de esta manera ser capaz de seleccionar la información realmente útil y hallar errores de ser necesario, identificando que esta herramienta es hecha para una buena y responsable utilización.

III. METODOLOGIA

En este informe se presentarán dos códigos realizados por ChatGPT, teniendo un conocimiento base de cómo debería ser la estructura de un código en Python, y de esta manera con las peticiones adecuadas, hacer que la inteligencia artificial (ChatGPT) nos ayude con casi toda la elaboración de los

programas. Estos programas tienen relación en lo que es la vida universitaria para un estudiante de la Escuela Colombiana de Ingeniería Julio Garavito.

1er Código

Como primer código se tomó la idea de hacer un sistema de cambio de contraseña, en el cual la nueva contraseña tendrá unas restricciones por seguridad. Se tomo esta idea para el código debido a que el estar cambiando constantemente las contraseña es algo común en muchas plataformas, sin ir muy lejos, la plataforma de la Escuela Colombiana de Ingeniería Julio Garavito exige un cambio de contraseña semestralmente, y esto fue lo que llevo a la realización de este programa.

Se desea crear un código en Python para cambiar la contraseña actual, este código debe aceptar como nueva contraseña una que sea diferente a la anterior, debe contener una mayúscula, y un carácter especial, de lo contrario, la contraseña no se podrá cambiar debido a que no se cumplen los parámetros de cambio. Adicionalmente, la contraseña debe tener un mínimo de 8 y máximo de 12 caracteres.

Se creo un chat con ChatGPT con el fin de que el sistema se vaya relacionando con el tema, para que en las próximas preguntas las respuestas recibidas por esta inteligencia artificial fueran más precisas, debido a que, con esta pregunta introductoria, ya tuvo que realizar una investigación y recopilación de información acerca del tema.

Como proyecto de una clase quisiera hacer un código que permita hacer el cambio de contraseñas, quisiera saber cual es el sistema para guardar usuarios y contraseñas en una plataforma, y que parámetros hay que tener en cuenta a la hora de diseñar un código que me permita realizar lo anterior.

Para crear un sistema de cambio de contraseñas, primero debes tener un método para almacenar usuarios y sus contraseñas de manera segura. Aquí hay una guía básica para que tengas en cuenta al diseñar tu código:

- Almacenamiento Seguro de Contraseñas:**
 - Nunca almacenes las contraseñas en texto claro. Utiliza un algoritmo de hash seguro como bcrypt o argon2 para almacenar las contraseñas de forma segura.
 - No implementes tu propio algoritmo de hashing, utiliza bibliotecas o módulos confiables.
- Base de Datos:**
 - Puedes utilizar una base de datos para almacenar la información de usuarios y contraseñas. Algunas opciones comunes son MySQL, PostgreSQL o MongoDB.
- Creación de Usuarios:**
 - Debes tener una funcionalidad para registrar nuevos usuarios. Pide la información necesaria (nombre de usuario, correo electrónico, contraseña) y asegúrate de validarla adecuadamente.
- Inicio de Sesión:**
 - Crea un sistema de inicio de sesión donde el usuario debe proporcionar su nombre de usuario o correo electrónico y su contraseña. Verifica que las credenciales coincidan con lo que está almacenado en la base de datos.
- Restablecimiento de Contraseñas:**
 - Permite a los usuarios restablecer sus contraseñas si las olvidan. Esto generalmente...

Send a message

ChatGPT can make mistakes. Consider checking important information.

Imagen 1. Pregunta #1 – Primer Código.

Se inicio con la realización del código, por lo cual, primero le pedí que me creara un código en el que la nueva contraseña no debe coincidir con la anterior.

Deseo crear un código en Python para cambiar la contraseña actual, este código debe aceptar como nueva contraseña una que sea diferente a la anterior.

Para crear un código en Python que permita cambiar la contraseña actual, puedes seguir estos pasos:

- Almacenamiento de Contraseñas:**
 - Asegúrate de que las contraseñas estén almacenadas de manera segura utilizando un algoritmo de hash seguro como bcrypt o argon2.
- Validación de la Nueva Contraseña:**
 - Verifica que la nueva contraseña cumple con los requisitos establecidos (longitud mínima, caracteres especiales, etc.).
- Verificación de la Contraseña Anterior:**
 - Antes de cambiar la contraseña, verifica que la contraseña actual proporcionada por el usuario sea correcta.
- Verificación de Contraseña Diferente:**
 - Asegúrate de que la nueva contraseña sea diferente a la anterior.

Aquí hay un ejemplo de código Python que te puede servir como punto de partida:

```
python
import bcrypt

# Supongamos que 'hashed_password' contiene la contraseña almacenada previamente
hashed_password = b'$2b$12$VSKRwLsWh5qI26puyAbKXuB0KsY0/BgUfK0mN.N.G0pB6dbb13'

def verificar_contraseña_actual(hashed_password, contraseña_actual):
    return bcrypt.checkpw(contraseña_actual.encode('utf-8'), hashed_password)
```

Imagen 2. Pregunta #2 – Primer Código.

Como respuesta, el código que nos dio ChatGPT es demasiado básico, así como dice en su respuesta, sirve como base, de esta manera se tomara este como un inicio de lo que puede ser el código final, y a partir de otras preguntas se intentara seguir complementando el código.

La siguiente petición que se le hizo a la inteligencia artificial fue que el código debe pedirle al usuario la contraseña anterior, para que de esta manera almacene esta información en su base de datos.

A partir del código anterior, deseo que se le añada que al usuario se le debe pedir la contraseña anterior para así almacenarla en la base de datos, también el usuario debe escribir su nueva contraseña y el código debe decirle al usuario si esta contraseña fue aceptada como su nueva contraseña. Por favor no suponer ninguna contraseña almacenada previamente

Entiendo tu solicitud. En este caso, vamos a modificar el código para que el usuario proporcione su contraseña actual, y luego se le pedirá que ingrese su nueva contraseña. Luego, el código verificará si la nueva contraseña es diferente a la anterior y, si todo es correcto, la almacenará y mostrará un mensaje de confirmación.

Aquí tienes el código modificado:

```
python
import bcrypt

def verificar_contraseña_actual(hashed_password, contraseña_actual):
    return bcrypt.checkpw(contraseña_actual.encode('utf-8'), hashed_password)

def cambiar_contraseña(hashed_password, contraseña_actual, nueva_contraseña):
    if not verificar_contraseña_actual(hashed_password, contraseña_actual):
        return "La contraseña actual es incorrecta"
    elif contraseña_actual == nueva_contraseña:
        return "La nueva contraseña debe ser diferente a la actual"
    else:
        nueva_contraseña_hash = bcrypt.hashpw(nueva_contraseña.encode('utf-8'), bcrypt.gensalt())
        # Aquí deberías actualizar la contraseña almacenada en tu base de datos
        return "Contraseña cambiada exitosamente"

# Simulación: Supongamos que 'hashed_password' es la contraseña almacenada previamente
hashed_password = None

# El usuario proporciona la contraseña actual y la nueva contraseña
contraseña_actual = input("Ingresa tu contraseña actual: ")
nueva_contraseña = input("Ingresa tu nueva contraseña: ")

# Llamada a la función para cambiar la contraseña
resultado = cambiar_contraseña(hashed_password, contraseña_actual, nueva_contraseña)

# Se imprime el resultado
print(resultado)
```

En este código, el usuario primero debe proporcionar su contraseña actual y luego su nueva contraseña. Luego, la función 'cambiar_contraseña' verifica si la contraseña actual es correcta y si la nueva contraseña es diferente. Si ambos criterios se cumplen, la función devuelve un mensaje de confirmación. Ten en cuenta que en lugar de imprimir el mensaje,

Imagen 3. Pregunta #3 – Primer Código.

El código que ChatGPT nos daba como resultado estaba usando una biblioteca diferente a la de los códigos utilizados en el código, así que se le dijo que la biblioteca utilizada era la incorrecta.

pip install bcrypt este codigo esta mal

Mis disculpas por la confusión. Si no tienes la biblioteca 'bcrypt' instalada, puedes utilizar la biblioteca estándar de Python 'hashlib' para realizar el hash de las contraseñas. Aquí está el código modificado:

```
python
import hashlib

def hash_contraseña(contraseña):
    # Utilizamos un salt aleatorio para mayor seguridad
    salt = b'$2b$12$VSKRwLsWh5qI26puyAbKXuB0KsY0/BgUfK0mN.N.G0pB6dbb13'
    return hashlib.sha256(salt + contraseña.encode('utf-8')).hexdigest()

def verificar_contraseña_actual(hashed_password, contraseña_actual):
    return hashed_password == hash_contraseña(contraseña_actual)

def cambiar_contraseña(hashed_password, contraseña_actual, nueva_contraseña):
    if not verificar_contraseña_actual(hashed_password, contraseña_actual):
        return "La contraseña actual es incorrecta"
    elif contraseña_actual == nueva_contraseña:
        return "La nueva contraseña debe ser diferente a la actual"
    else:
        nueva_contraseña_hash = hash_contraseña(nueva_contraseña)
        # Aquí deberías actualizar la contraseña almacenada en tu base de datos
        return "Contraseña cambiada exitosamente"

# Simulación: Supongamos que 'hashed_password' es la contraseña almacenada previamente
hashed_password = None

# El usuario proporciona la contraseña actual y la nueva contraseña
contraseña_actual = input("Ingresa tu contraseña actual: ")
nueva_contraseña = input("Ingresa tu nueva contraseña: ")

# Llamada a la función para cambiar la contraseña
resultado = cambiar_contraseña(hashed_password, contraseña_actual, nueva_contraseña)

# Se imprime el resultado
print(resultado)
```

Imagen 4. Pregunta #4 – Primer Código.

Evidenciando que tenía un error en la biblioteca, se decidió por volver a empezar y exigirle un nuevo código con la biblioteca nueva, con las mismas especificaciones que ya debería tener el código.

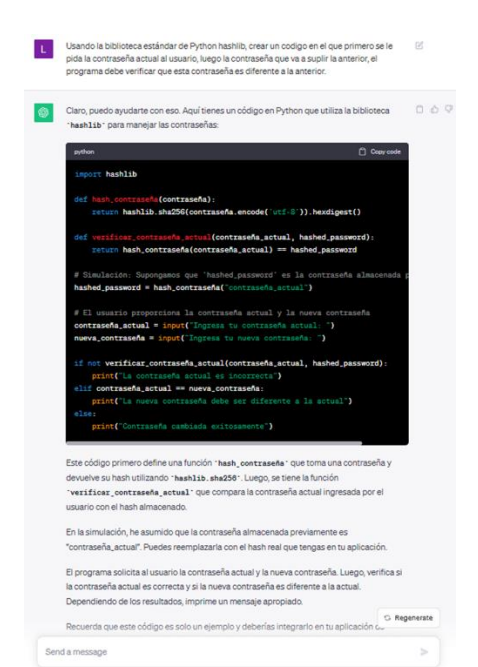


Imagen 5. Pregunta #5 – Primer Código.

Con este código como respuesta se decidió seguir con la elaboración del código, en el que ahora se le colocó una restricción, se tiene que cumplir el rango de caracteres para el cambio de contraseña.

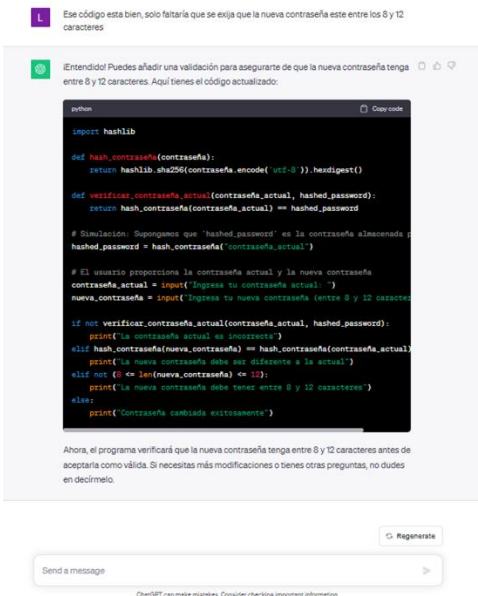


Imagen 6. Pregunta #6 – Primer Código.

Con este código como respuesta se decidió colocar la restricción de contener al menos una mayúscula para el cambio

de contraseña, con el propósito de tener un código más parecido a la realidad en lo que sería un sistema de actualización de contraseña.

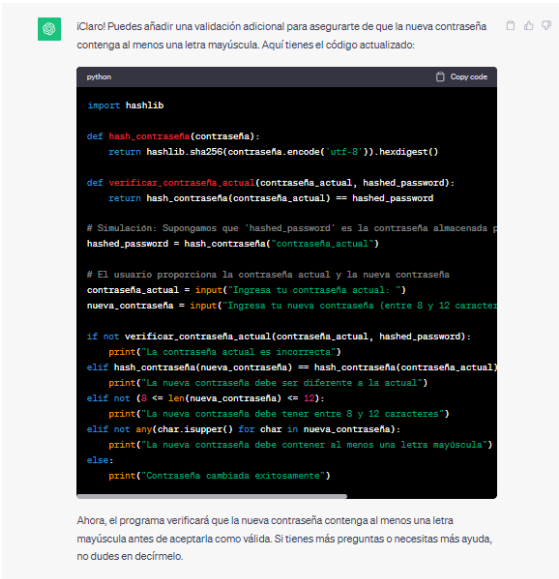


Imagen 7. Pregunta #7 – Primer Código.

Este fue el código final que funcionó de manera correcta en Python, a continuación, se mostrara el resultado del código y el buen funcionamiento del programa.

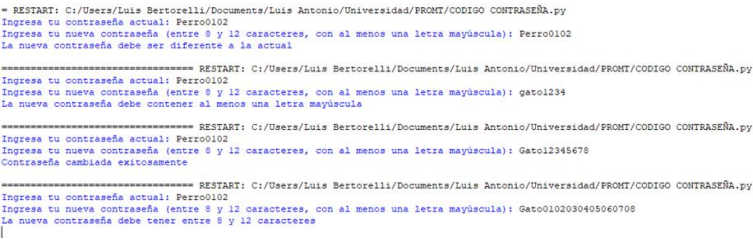


Imagen 8. Resultado en Python – Primer Código.

2do Código

Como segundo código se tomó la idea de hacer un programa parecido al sistema que usan las máquinas dispensadoras de productos, en las cuales se le ingresa dinero, se compra el producto y este devuelve el dinero que sobra si es que existe. Este ejemplo lo tome debido a que en la Escuela Colombiana de Ingeniería Julio Garavito hay muchas de estas máquinas y tienen un uso frecuente en la comunidad estudiantil.

Para este programa se desea que el usuario ingrese un billete, este le permita comprar unos productos con respectivo valor, y la máquina le devuelva el dinero, el código también debe decir que si el dinero no es suficiente la máquina le avise que el dinero no es suficiente.

Para este código como para el anterior se debió crear un chat con ChatGPT, para que vaya familiarizándose con el tema.

Quisiera saber como funciona el sistema de las maquinas dispensadoras de comida, y que parámetros son importantes en este tema

Las máquinas dispensadoras de comida, también conocidas como máquinas expendedoras o vending, son dispositivos automáticos diseñados para proporcionar una variedad de productos, que van desde alimentos y bebidas hasta artículos no perecederos y suministros diversos. Estas máquinas operan mediante un sistema automatizado que permite a los clientes seleccionar y adquirir productos sin necesidad de la asistencia de un empleado.

Aquí hay una descripción básica del funcionamiento y los parámetros importantes en el tema de las máquinas dispensadoras de comida:

1. Sistema de selección y pago:

- **Panel de selección:** Las máquinas expendedoras están equipadas con un panel de selección que incluye botones, teclas numéricas o una pantalla táctil que permite

Imagen 9. Pregunta #1 – Segundo Código

Una vez ya puesto el tema en el chat, se inicia con una petición específica de que realice un código de Python que cumpla con el objetivo

Quisiera hacer un código en python en el cual se tiene una maquina dispensadora de productos, esta tenga seis productos con precios de 1000, 1500, 2000, 2500, 3000, 3500 pesos colombianos, la maquina solo acepta billetes de 2000, 5000 y de 10000, y la maquina debe devolver el dinero sobrante de la compra del producto.

¡Por supuesto! Puedes crear un programa en Python que simule el funcionamiento de una máquina dispensadora de productos con los requisitos que mencionaste. Aquí tienes un ejemplo de código que puedes utilizar como punto de partida:

```
python
# Definir productos y sus precios
productos = {
    1: {"nombre": "Producto 1", "precio": 1000},
    2: {"nombre": "Producto 2", "precio": 1500},
    3: {"nombre": "Producto 3", "precio": 2000},
    4: {"nombre": "Producto 4", "precio": 2500},
    5: {"nombre": "Producto 5", "precio": 3000},
    6: {"nombre": "Producto 6", "precio": 3500}
}

# Definir billetes aceptados
billetes_aceptados = [10000, 5000, 2000]

# Función para mostrar productos
def mostrar_productos():
    for id, producto in productos.items():
        print(f"{id}. {producto['nombre']} - Precio: {producto['precio']} pesos")
```

Imagen 10. Pregunta #2 – Segundo Código

El código respuesta que ChatGPT dio, fue bastante bueno, sin embargo, la intención de este código es que la opción a comprar se pueda elegir en pantalla, así que la siguiente petición fue esa misma.

Me gustaría que los productos se pudieran elegir mediante botones

Para permitir que los productos se elijan mediante botones en la consola, puedes utilizar la librería `keyboard` de Python. Esta librería te permitirá detectar las teclas presionadas por el usuario. Aquí te muestro cómo puedes modificar el código para implementar esta funcionalidad:

Primero, asegúrate de tener la librería `keyboard` instalada. Si no la tienes instalada, puedes hacerlo usando `pip`:

```
bash
pip install keyboard
```

Luego, puedes usar el siguiente código modificado:

```
python
import keyboard

# ... (código anterior)

# Función para obtener la selección del usuario usando teclas
def obtener_seleccion():
    while True:
        try:
            evento = keyboard.read_event(suppress=True)
            if evento.event_type == keyboard.KEY_DOWN:
                tecla = evento.name
```

Imagen 11. Pregunta #3 – Segundo Código

Esta biblioteca no era la que se esperaba, así que se volvió a hacer la petición, pero con mucha mayor especificación.

Esta bien, pero me gustaría que esos botones aparecieran en la pantalla

Para crear una interfaz gráfica en Python, puedes utilizar la librería `tkinter`, que es una biblioteca estándar de Python para crear interfaces de usuario. Puedes utilizarla para crear una ventana con botones para seleccionar los productos. Aquí te muestro cómo modificar el código:

```
python
import tkinter as tk

# ... (código anterior)

def crear_interfaz():
    # Crear ventana
    ventana = tk.Tk()
    ventana.title("Máquina Dispensadora")

    # Función para realizar la compra
    def comprar():
        producto_id = seleccion.get()
        if producto_id in productos:
            monto_insertado = int(monto_entry.get())
            if monto_insertado in billetes_aceptados:
                comprar_producto(producto_id, monto_insertado)
            else:
                resultado.set("Billete no válido.")
        else:
            resultado.set("Producto no válido.")

    # Mostrar productos
```

Imagen 12. Pregunta #4 – Segundo Código

Esta vez el código si cumplió con las expectativas, ya que este código crea una interfaz en donde el usuario puede interactuar, sin embargo, el código tenía una falla, ya que no se decía que cuando el dinero no alcanzaba no se podría comprar ese producto, si no que se hacia la compra con el saldo insuficiente, así que se le dijo que corrigiera ese error.

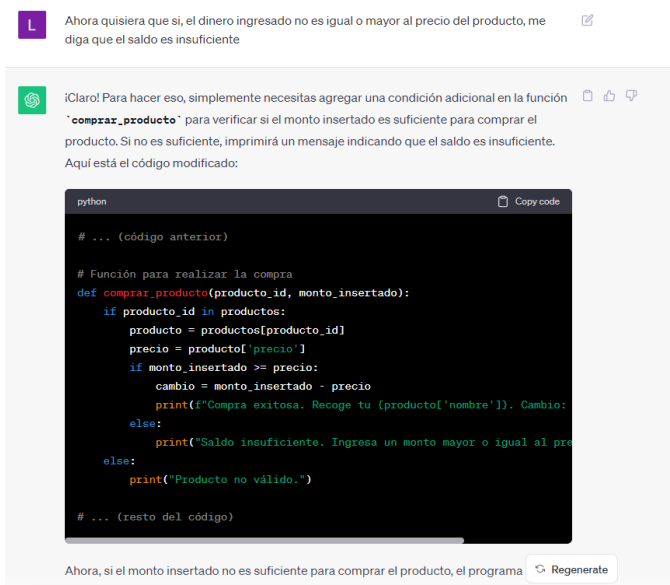


Imagen 13. Pregunta #5 – Segundo Código

Finalmente, el código quedó terminado, y se le hicieron unas modificaciones al código general que ChatGPT realizó, resultando de la siguiente manera

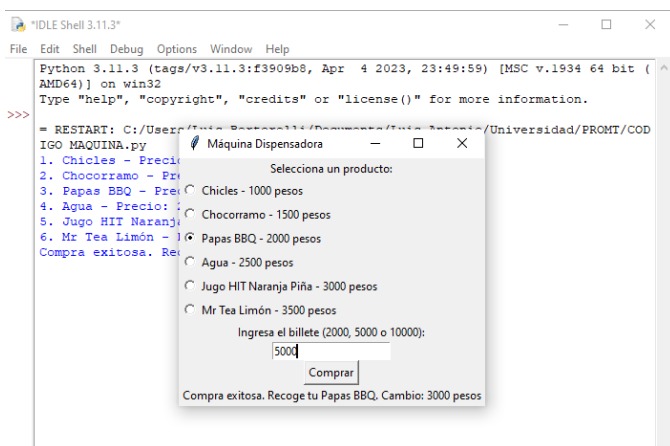


Imagen 14. Resultado en Python – Segundo Código

IV. RESULTADOS Y ANÁLISIS

Como análisis de resultados se evidencia para el primer programa se realizó perfectamente el código, sin embargo, se le debió hacer un ajuste en el orden de las líneas de código y en unas comillas que hacían que el código no funcionara adecuadamente, presentando un error en este. Por el contrario, para la elaboración del segundo programa ChatGPT tuvo más precisión, y esto se debe a que la conversación y las preguntas llevaban un hilo, y el código solo se tuvo que modificar en su apariencia mas no en su funcionamiento. Lo cual permite analizar que el error está en saber cómo llevar la conversación y en cómo hacerle las peticiones o las preguntas a la inteligencia artificial, ya que de esto depende a precisión de la respuesta.

V. CONCLUSIÓN

Como se propuso en los objetivos, se adquirió destreza y habilidad en el manejo de la inteligencia artificial, en donde fue necesario realizar las preguntas e instrucciones apropiadas para que la herramienta generara los códigos con todos sus requerimientos y funcionalidades, teniendo en cuenta que se necesitó hacer correcciones y aclaraciones en algunas instrucciones, seleccionando así la información realmente útil y la que no era necesaria, además de las adecuadas demostraciones que ponen en evidencia que el código final cumplió con todo.