Documentación del Proyecto de Generación de Ideas para Regalos

Descripción General

Este proyecto es una herramienta automatizada diseñada para generar ideas de regalos personalizados. Utiliza datos de intereses personales extraídos de un archivo CSV y la API de OpenAI para sugerir regalos que se alinean con las preferencias individuales. Además, genera enlaces de Amazon para estos regalos y proporciona explicaciones sobre por qué cada regalo es adecuado.

Características Principales

- **Extracción de Datos:** Lee un archivo CSV para obtener intereses específicos.
- **Generación de Regalos:** Utiliza la API de OpenAI para sugerir regalos basados en los datos extraídos.
- **Enlaces de Amazon:** Crea enlaces directos a Amazon para facilitar la compra de los regalos sugeridos.
- **Explicaciones Personalizadas:** Proporciona razones personalizadas de por qué un regalo es adecuado, utilizando la inteligencia artificial.
- **Visualización Creativa:** Genera una imagen temática navideña usando DALL·E 3, mostrando los regalos sugeridos.

Requerimientos

- **Python:** La herramienta está escrita en Python, por lo que se requiere una instalación de Python.
- **Bibliotecas de Python:** Es necesario instalar `csv` y `re` para el procesamiento de datos, y `openai y os` para la integración con las APIs de OpenAI.
- **Clave API de OpenAI:** Se requiere una clave API válida de OpenAI para acceder a sus servicios de inteligencia artificial.
- **Archivo CSV:** Se necesita un archivo CSV que contenga los datos de los intereses personales, generado por la extensión descrita en el documento README.

Uso

1. **Instalación de Dependencias:** Instalar las bibliotecas necesarias mediante `pip`.
2. **Configuración de la Clave API:** Proporcionar la clave API de OpenAI al programa.
3. **Carga de Datos:** Ingresar la ubicación y detalles del archivo CSV que genero la extensión de chrome.
4. **Generación de Ideas para Regalos:** El programa procesará los datos y utilizará la API de OpenAI para sugerir regalos.

5. **Enlaces de Compra y Explicaciones:** El programa proporcionará enlaces de Amazon para los regalos sugeridos y explicará por qué son adecuados.
6. **Visualización de Regalos:** Se generará una imagen navideña con los regalos utilizando DALL·E 3.
7. **Calculo de costo:** Se generará un cálculo de costos dado los tokens utilizados en la ejecución.

Seguridad y Privacidad

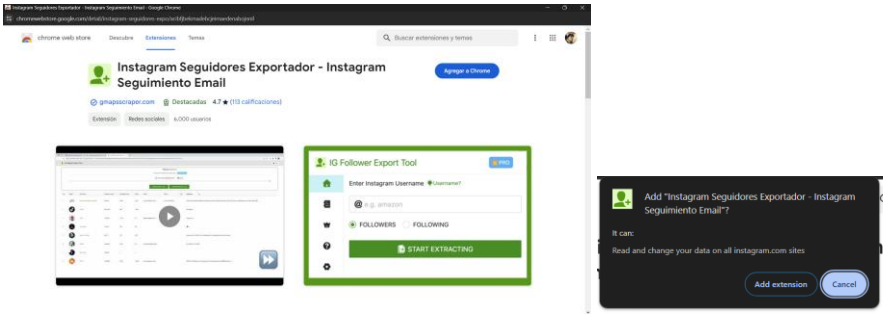
- La clave API de OpenAI debe ser manejada de forma segura y no debe exponerse en el código fuente. Por lo que en el propio código se solicita al usuario.

Contribuciones y Soporte

- Las contribuciones son bienvenidas a través de GitHub.
- De momento no existe un soporte a largo plazo.

Instalación de la extensión de Chrome

https://ig-follower-extractor.gmapsscraper.com/



Imágenes del programa en ejecución

Germán Andrés Magallón Corona

```
[1] import csv
    import re
    import openai
    import os
    import unicodedata

    # Pedir al usuario información relevante
    nombre_archivo = input("Ingresa el nombre del archivo CSV (incluyendo la extensión .csv): ")
    nombre_persona = input("Ingresa el nombre de la persona a la que le vas a dar el regalo: ")
    genero_persona = input("Ingresa el genero de la persona: ")
    edad_persona = input("Ingresa la edad aproximada de la persona: ")
    datos_persona = "La persona tiene"+ edad_persona + " años y su genero es "+ genero_persona
```

```
Ingresa el nombre del archivo CSV (incluyendo la extensión .csv): IG-Follower-Email-Scraper_1702104246421.csv
Ingresa el nombre de la persona a la que le vas a dar el regalo: Paulina
Ingresa el genero de la persona: Femenino
Ingresa la edad aproximada de la persona: 22
```

```
[2] # Pedir al usuario que ingrese la clave API
    openai_api_key = input("Ingresa tu clave API de OpenAI: ")
    os.environ["OPENAI_API_KEY"] = openai_api_key
```

```
Ingresa tu clave API de OpenAI: [redacted]
```

```
[3] #Lee el archivo csv, lo limita a 1000 usuarios como maximo y limpia los datos para que sean faciles de leer para la IA
    def leer_y_limpiar_csv(nombre_archivo, limite=1000):
        seguidos = []
        with open(nombre_archivo, mode='r', encoding='utf-8') as csvfile:
            reader = csv.reader(csvfile)
            for row in reader:
                if row and len(seguidos) < limite:  # Verifica si la fila no está vacía y si no se ha alcanzado el límite
                    usuario = row[4]  # Asumiendo que los nombres de usuario están en la quinta columna (índice 4)
                    usuario_limpio = re.sub(r'[0-9]', '', usuario)  # Eliminar números
                    usuario_limpio = usuario_limpio.replace('_', ' ').replace('.', ' ')  # Reemplazar '_' y '.' con espacios
                    seguidos.append(usuario_limpio)
                else:
                    break  # Romper el bucle si se alcanza el límite
        return seguidos
```

```
[4] pip install openai==0.28
```

```
[8] seguidos = leer_y_limpiar_csv(nombre_archivo)
    def generar_recomendaciones_de_regalos(seguidos, openai_api_key):
        openai.api_key = openai_api_key

        # Crear un prompt que incluya los seguidos y pida recomendaciones de regalos
        prompt = "Basado en los siguientes intereses y personas seguidas: {}\n".format(", ".join(seguidos))
        prompt += "Por favor, sugiere 3 regalos baratos y fáciles de encontrar para Navidad. Solo requiero titulos cortos, porque con esos titulos voy a buscar los productos en Amazon."
        prompt += "Por favor ten encuenta también los siguientes factores de genero y edad aproximada: {}\n".format(", ".join(datos_persona))

        try:
            response = openai.ChatCompletion.create(
                model="gpt-4",  # Asegúrate de que este es el modelo correcto
                messages=[
                    {"role": "system", "content": "Tu prompt de sistema aquí"},
                    {"role": "user", "content": prompt}
                ]
            )
            return response.choices[0].message['content'].strip().split('\n')
        except Exception as e:
            print(f"Error al generar recomendaciones: {e}")
            return []

    # Guardar las recomendaciones de regalos
    regalos = generar_recomendaciones_de_regalos(seguidos, openai_api_key)
```

Germán Andrés Magallón Corona

```
[16] #Quitar los acentos para generar correctamente los links
     import unicodedata
     def eliminar_acentos(texto):
         return unicodedata.normalize('NFKD', texto).encode('ASCII', 'ignore').decode('ASCII')

     # Generar los links de búsqueda de Amazon México
     def generar_enlace_amazon(busqueda):
         # Eliminar comillas y otros caracteres no deseados
         busqueda_limpia = eliminar_acentos(busqueda.replace('"', '').replace('.', '').replace(',', ''))
         # Separar las palabras y unirlas con '+'
         palabras = busqueda_limpia.split()
         url_amazon = "https://www.amazon.com.mx/s?k=" + '+'.join(palabras)
         return url_amazon

     # Guardar los enlaces
     enlaces_amazon = [generar_enlace_amazon(regalo) for regalo in regalos]

     # Texto introductorio para la lista de regalos
     print(f"Dada la lista de seguidos que me diste de {nombre_persona}, pienso que estos regalos pueden gustarle: ")

     for regalo, enlace in zip(regalos, enlaces_amazon):
         print(f"Regalo: {regalo}, Enlace de Amazon: {enlace}")
```

```
Dada la lista de seguidos que me diste de Paulina, pienso que estos regalos pueden gustarle:
Regalo: 1. "Libro de cocina saludable", Enlace de Amazon: https://www.amazon.com.mx/s?k=1+Libro+de+cocina+saludable
Regalo: 2. "Set de joyería minimalista", Enlace de Amazon: https://www.amazon.com.mx/s?k=2+Set+de+joyeria+minimalista
Regalo: 3. "Kit de cuidado para plantas", Enlace de Amazon: https://www.amazon.com.mx/s?k=3+Kit+de+cuidado+para+plantas
```

```
[27] def generar_explicaciones(regalos, nombre_persona):
         # Simplificar el prompt para evitar exceder el límite de caracteres
         prompt = f"Explica por qué crees que los siguientes regalos son adecuados para {nombre_persona} dado los datos que analisaste basandote en su lista de seguidos de instagram, quien
         for regalo in regalos:
             prompt += f"- {regalo}\n"
         prompt += "Explicación:"

         try:
             response = openai.ChatCompletion.create(
                 model="gpt-4",
                 messages=[
                     {"role": "system", "content": "You are a helpful assistant."},
                     {"role": "user", "content": prompt}
                 ]
             )
             return response.choices[0].message["content"]
         except Exception as e:
             print(f"Error al generar explicaciones: {e}")
             return ""

     # Ejemplo de uso de la función
     explicaciones = generar_explicaciones(regalos, nombre_persona)
     print("La explicación para estos regalos es:\n", explicaciones)
```

```
La explicación para estos regalos es:
 1. "Libro de cocina saludable": Paulina sigue varias páginas relacionadas con el fitness y la alimentación saludable en Instagram. Este interés en la comida sana y su deseo aparente d

 2. "Set de joyería minimalista": Al analizar a quienes sigue Paulina en Instagram, notamos que sigue a varias marcas y personas influyentes en el campo de la moda, especialmente en lo

 3. "Kit de cuidado para plantas": Paulina sigue a algunas páginas de jardinería y plantas en Instagram, lo que sugiere que tiene interés en las plantas y posiblemente ya tenga algunas
```

```
[28] #Generar la imagen con Dalle3
     def generar_imagen_dalle(descripcion, openai_api_key):
         openai.api_key = openai_api_key

         try:
             response = openai.Image.create(
                 model="dall-e-3",  #modelo de DALL·E 3
                 prompt=descripcion,
                 n=1,  # Número de imágenes a generar
                 size="1024x1024"  # Tamaño de la imagen
             )
             return response.data[0]['url']  # URL de la imagen generada
         except Exception as e:
             print(f"Error al generar la imagen: {e}")
             return ""

     # Imprimir la imagen
     descripcion = f"Una escena navideña con un árbol decorado y los tres regalos sugeridos. Que son: {regalos}"
     url_imagen = generar_imagen_dalle(descripcion, openai_api_key)
     print("URL de la imagen generada:", url_imagen)
```

```
URL de la imagen generada: https://oaidalleapiprodscus.blob.core.windows.net/private/org-xiWjKf2e54NggRLrsf1h5JFQ/user-cfoGGhomQX8mh3hOlZhgp0dr/img-TQMBccsugj2dkCVokW2WTL5n.png?st=2023
```

Germán Andrés Magallón Corona

```
[29] # Ejemplo de cálculo de costo considerando la longitud de las respuestas de ChatGPT y una imagen de DALL-E

     # Tarifas estimadas
     PRECIO_CHATGPT_POR_TOKEN = 0.00002  # Precio estimado por token para ChatGPT
     PRECIO_DALLE_POR_IMAGEN = 0.02      # Precio por imagen generada por DALL·E

     # Estimación de tokens: Asumiendo que cada token tiene aproximadamente 4 caracteres
     num_tokens_regalos = sum(len(regalo) for regalo in regalos) / 4
     num_tokens_amazon = sum(len(enlace) for enlace in enlaces_amazon) / 4
     num_tokens_explicacion = len("La explicación para estos regalos es: " + explicaciones) / 4

     # Calcular el costo total de los tokens
     costo_tokens = (num_tokens_regalos + num_tokens_amazon + num_tokens_explicacion) * PRECIO_CHATGPT_POR_TOKEN

     # Calcular el costo total incluyendo una imagen de DALL·E
     costo_total = costo_tokens + PRECIO_DALLE_POR_IMAGEN

     print(f"El costo total en dolares americanos de esta ejecución fue de: ${costo_total:.5f}")

     El costo total en dolares americanos de esta ejecución fue de: $0.02708
```