

# **CREACIÓN DE UN SISTEMA DE ATENCIÓN AL CLIENTE POR MEDIO DE UN CHATBOT BASADO EN LENGUAJE NATURAL**

Cristian Leonardo Giraldo Pabón

**PROMPT-2023-2**

Escuela Colombiana de Ingeniería Julio Garavito

10 de noviembre de 2023

## **1. Contexto**

En el mundo actual, la interacción entre humanos y tecnología está evolucionando rápidamente. Una de las áreas clave en esta evolución es el Procesamiento del Lenguaje Natural (NLP), que permite a las máquinas entender y responder al lenguaje humano de una manera más inteligente y natural. En este contexto, he trabajado en la creación de un chatbot, una aplicación que utiliza NLP para interactuar con los usuarios de una manera conversacional.

## **2. Objetivo General**

El objetivo principal de nuestro chatbot es proporcionar respuestas rápidas y efectivas a las preguntas frecuentes de los usuarios, así como manejar consultas más específicas mediante el procesamiento inteligente del lenguaje natural. En esencia, queríamos crear una interfaz amigable que brinde asistencia y resuelva dudas de manera eficiente.

## **3. Uso de chatGPT para el desarrollo de un chatbot de atención al cliente para un servicio de operador móvil**

Se dió uso de la herramienta chatGPT como una estrategia de aprendizaje para facilitar la creación del objetivo específico, por lo que inicialmente se le

solicitó información sobre este y se obtuvo un desarrollo paso a paso del proceso para llevar a cabo el objetivo, a continuación se presentan los prompts desarrollados por chatGPT:

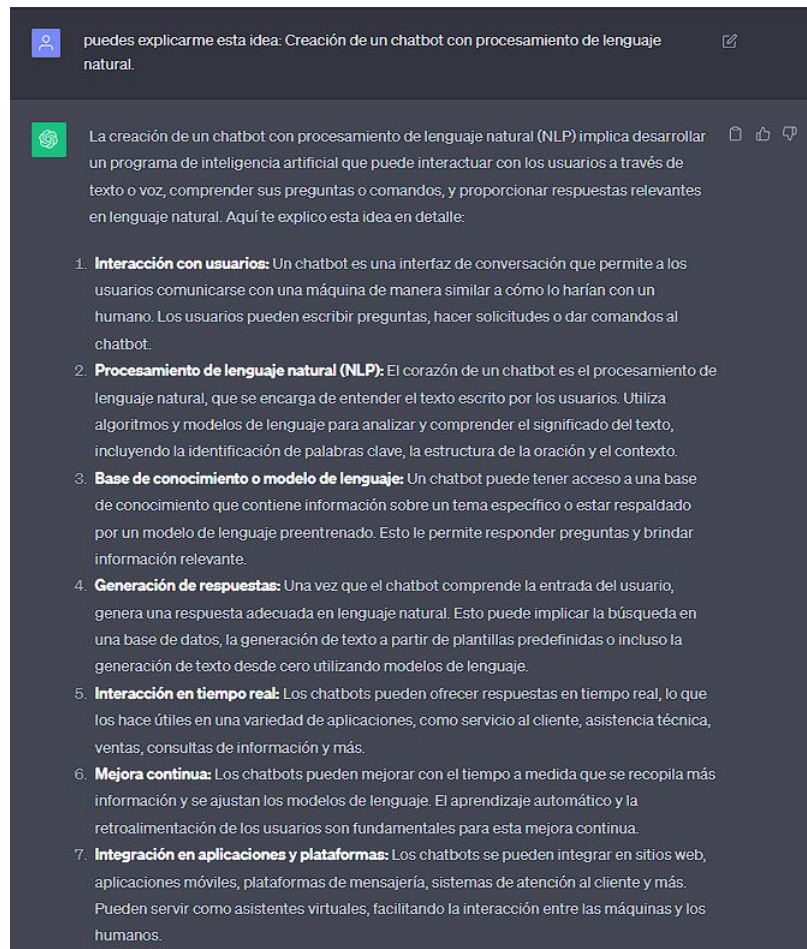
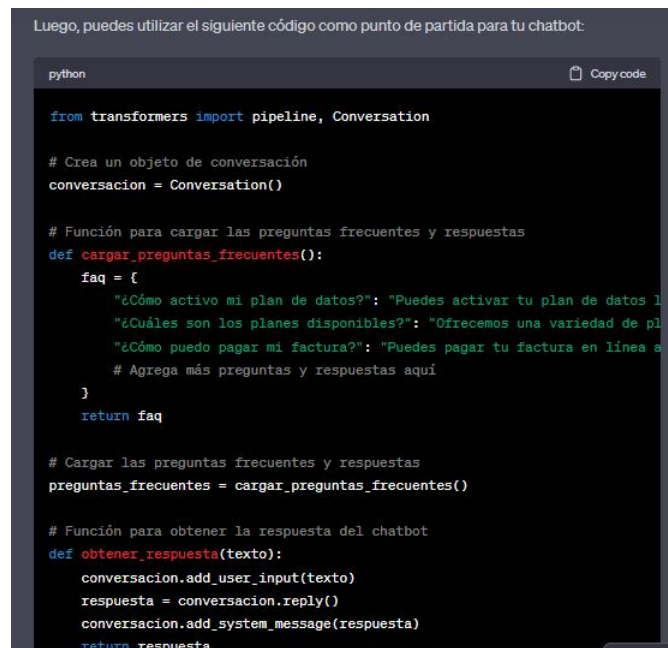
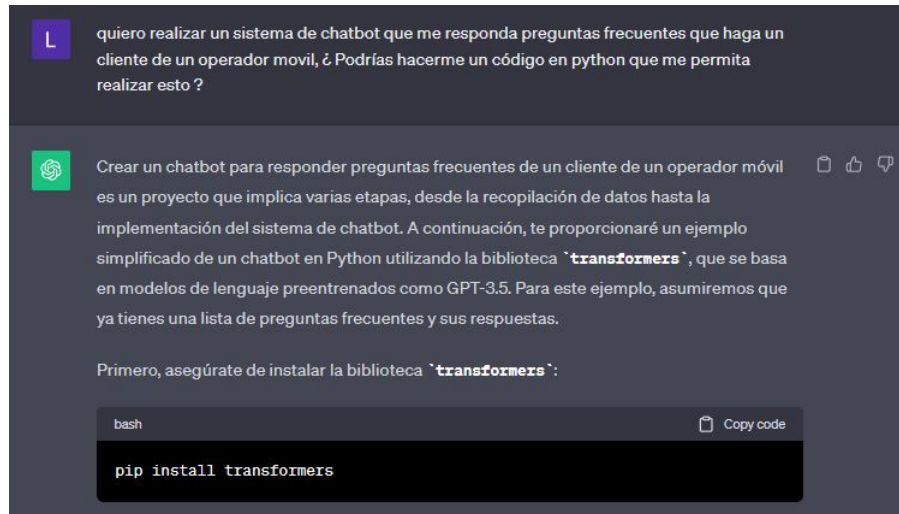


Figura 1: Explicación del objetivo a cumplir

A partir de esta explicación detallada, se empezó a desarrollar la idea para crear un chatBot principalmente de atención al cliente para un servicio de operador móvil en el que se realizan preguntas frecuentes y se quieren acortar tiempos resolviendo estas preguntas de manera sencilla, rápida y eficaz.

### 3.1. Desarrollo de los códigos en lenguaje Python

A partir de la idea general planteada se realizó la consulta a chatGPT para el desarrollo de los códigos y la implementación del programa:



```

# Función para responder preguntas frecuentes
def responder_pregunta_frecuente(pregunta):
    if pregunta in preguntas_frecuentes:
        respuesta = preguntas_frecuentes[pregunta]
    else:
        respuesta = "Lo siento, no tengo información sobre esa pregunta en e
    return respuesta

# Loop principal para interactuar con el chatbot
while True:
    entrada_usuario = input("Usuario: ")

    if entrada_usuario.lower() == "salir":
        print("Chatbot: Adiós. Hasta luego.")
        break

    # Intentar encontrar una respuesta en las preguntas frecuentes
    respuesta = responder_pregunta_frecuente(entrada_usuario)

    # Si la respuesta no está en las preguntas frecuentes, utilizar el chatb
    if respuesta == "Lo siento, no tengo información sobre esa pregunta en e
        respuesta = obtener_respuesta(entrada_usuario)

    print("Chatbot:", respuesta)

```

Este código implementa un chatbot que responde preguntas frecuentes utilizando una lista predefinida de preguntas y respuestas. Si el chatbot no puede encontrar una respuesta en las preguntas frecuentes, utiliza el modelo de lenguaje GPT-3.5 para generar una respuesta.

A partir del código dado por ChatGPT se realizaron las pruebas en Google Colab para comprobar su funcionamiento, además fue necesario agregar al código algunas preguntas frecuentes que queremos que el chatBot nos responda, de esta forma serán reconocidas y se podrán usar para enviar respuestas coherentes.

se obtuvieron resultados adecuados a la función solicitada, a continuación se muestra el resultado del código ejecutado:

```

Usuario: como puedo activar mi plan de datos
Chatbot: Puede activar su plan de datos llamando al servicio de atención al cliente al 1234 o visitando nuestro sitio web.

```

Figura 2: ChatBot inicialmente generado

A partir de esto se podría decir que el código funciona sin ningún error, sin embargo, realizando más pruebas; al ingresar una pregunta que no se encontraba dentro de las preguntas frecuentes, el programa arrojaba un error, ya que no encontraba respuesta a la pregunta realizada, por lo que nuevamente se le solicitó a ChatGPT ayuda para solucionar el error:

```
Usuario: ¿ Cuánto vale el plan de datos más sencillo ?
-----
AttributeError                                Traceback (most recent call last)
<ipython-input-2-5a7375deffb1> in <cell line: 35>()
    45     # Si la respuesta no está en las preguntas frecuentes, utilizar el chatbot de GPT-3.5
    46     if respuesta == "to lento, no tengo información sobre esa pregunta en este momento.":
----> 47         respuesta = obtener_respuesta(entrada_usuario)
    48
    49     print("Chatbot:", respuesta)

<ipython-input-2-5a7375deffb1> in obtener_respuesta(texto)
    20 def obtener_respuesta(texto):
    21     conversacion.add_user_input(texto)
----> 22     respuesta = conversacion.reply()
    23     conversacion.add_system_message(respuesta)
    24     return respuesta

AttributeError: 'Conversation' object has no attribute 'reply'

[BUSCAR EN STACK OVERFLOW]
```

Figura 3: Error encontrado al ejecutar el código

L Al hacer la prueba e ingresar una pregunta que no está dentro de las preguntas frecuentes no obtengo la respuesta esperada, si no obtengo el siguiente error:  
AttributeError: 'Conversation' object has no attribute 'reply'

Mis disculpas por la confusión. Parece que el error proviene de la biblioteca "transformers", que no admite la función "reply" en versiones más recientes. Para simplificar el código y hacerlo más efectivo, puedes utilizar directamente el modelo de lenguaje GPT-3.5 a través de la biblioteca "openai". Asegúrate de tener instalada la biblioteca "openai".

```
bash
pip install openai
```

Como se puede observar en la imagen es necesario ser muy específico con la pregunta que se le realiza a ChatGPT, ya que este tendrá más herramientas para poder darnos la solución correcta al problema

```
Luego, puedes utilizar el siguiente código como punto de partida:

python
import openai

# Define tu clave de API de OpenAI
api_key = "tu_clave_de_api_aqui"

# Función para responder preguntas
def responder_pregunta(pregunta):
    try:
        respuesta = openai.Completion.create(
            engine="text-davinci-002",
            prompt=pregunta,
            max_tokens=100,
            api_key=api_key
        )
        return respuesta.choices[0].text.strip()
    except Exception as e:
        return str(e)

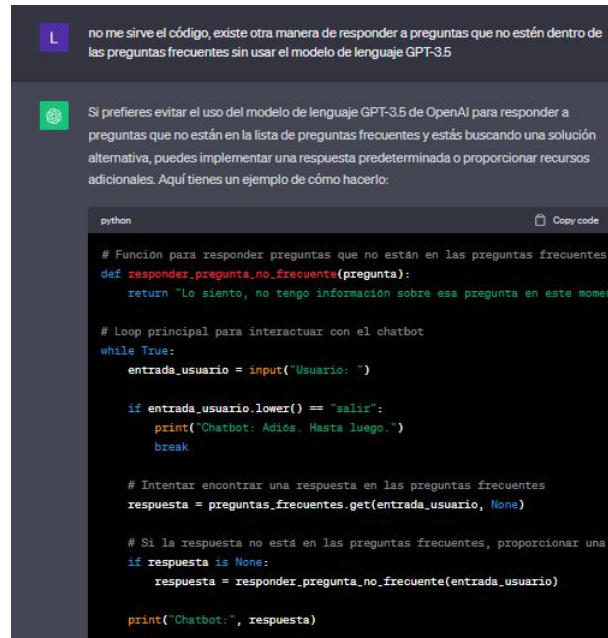
# Loop principal para interactuar con el chatbot
while True:
    entrada_usuario = input("Usuario: ")

    if entrada_usuario.lower() == "salir":
        print("Chatbot: Adiós. Hasta luego.")
        break

    respuesta = responder_pregunta(entrada_usuario)

    print("Chatbot:", respuesta)
```

Al observar y analizar las soluciones dadas por ChatGPT fue posible ver que se utilizaba directamente el modelo de lenguaje GPT-3.5 a través de la biblioteca openai para responder las preguntas, para esto era necesario la clave API suministrada por OpenAI por lo que no se logró realizar la conexión, sin embargo se optó por una solución más sencilla por lo que también se realizó la consulta con chatGpt.



The screenshot shows a chat window with a dark background. At the top, a user icon (a purple square with a white 'L') is followed by the text: "no me sirve el código, existe otra manera de responder a preguntas que no estén dentro de las preguntas frecuentes sin usar el modelo de lenguaje GPT-3.5". Below this, a bot icon (a green circle with a white gear) is followed by the text: "Si prefieres evitar el uso del modelo de lenguaje GPT-3.5 de OpenAI para responder a preguntas que no están en la lista de preguntas frecuentes y estás buscando una solución alternativa, puedes implementar una respuesta predeterminada o proporcionar recursos adicionales. Aquí tienes un ejemplo de cómo hacerlo:". Below the text is a code block with a "python" label and a "Copy code" button. The code is a Python script for a chatbot that uses a predefined response for questions not in the frequent questions list.

```
python Copy code

# Función para responder preguntas que no están en las preguntas frecuentes
def responder_pregunta_no_frecuente(pregunta):
    return "Lo siento, no tengo información sobre esa pregunta en este momento."

# Loop principal para interactuar con el chatbot
while True:
    entrada_usuario = input("Usuario: ")

    if entrada_usuario.lower() == "salir":
        print("Chatbot: Adiós. Hasta luego.")
        break

    # Intentar encontrar una respuesta en las preguntas frecuentes
    respuesta = preguntas_frecuentes.get(entrada_usuario, None)

    # Si la respuesta no está en las preguntas frecuentes, proporcionar una
    if respuesta is None:
        respuesta = responder_pregunta_no_frecuente(entrada_usuario)

    print("Chatbot:", respuesta)
```

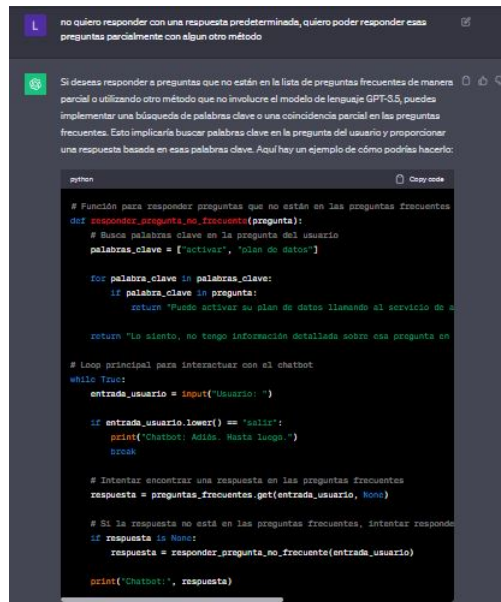
En este código, hemos agregado la función `responder_pregunta_no_frecuente` que proporciona una respuesta predeterminada cuando no se encuentra una pregunta en la lista de preguntas frecuentes. Esto permite ofrecer una respuesta estándar a preguntas que no están en la lista.

Además, puedes ajustar la respuesta predeterminada según tus necesidades o incluso proporcionar enlaces a recursos adicionales para que los usuarios busquen más información.

Esta solución es más simple y no requiere el uso de la API de OpenAI. Puedes personalizar la respuesta predeterminada y los recursos adicionales de acuerdo con tu caso de uso y las necesidades de tus usuarios.

Sin embargo, no se quería obtener una respuesta estandar, es por esto que se implementó un sistema de identificación de palabras claves y al obtener estas, respondía con el conjunto de respuestas que más se acomodaba a la pregun-

ta, para realizar esto se le solicitó a ChatGPT específicamente la tarea que se requería conseguir:



The screenshot shows a ChatGPT interface with a user prompt in Spanish asking for a way to handle questions not in a frequent questions list. The AI response explains a keyword search method and provides a Python code snippet. The code defines a function `responder_pregunta_no_frecuente` that searches for keywords in a user's question and returns a response based on the results. It also includes a main loop for interacting with the chatbot.

```
python
# Función para responder preguntas que no están en las preguntas frecuentes
def responder_pregunta_no_frecuente(pregunta):
    # Busca palabras clave en la pregunta del usuario
    palabras_clave = ["activar", "plan de datos"]

    for palabra_clave in palabras_clave:
        if palabra_clave in pregunta:
            return "Puede activar su plan de datos llamando al servicio de a"

    return "Lo siento, no tengo información detallada sobre esa pregunta en"

# Loop principal para interactuar con el chatbot
while True:
    entrada_usuario = input("Usuario: ")

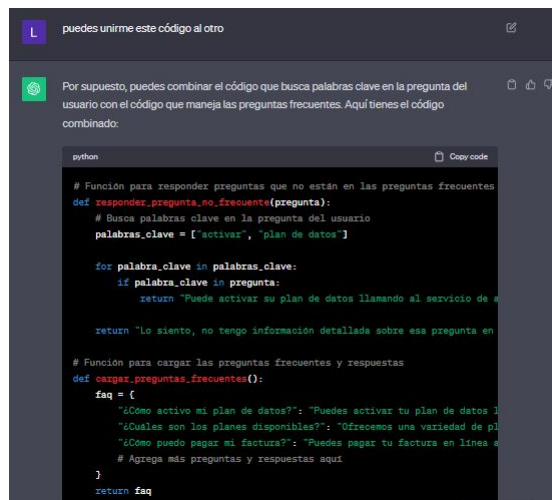
    if entrada_usuario.lower() == "salir":
        print("Chatbot: Adios. Hasta luego.")
        break

    # Intentar encontrar una respuesta en las preguntas frecuentes
    respuesta = preguntas_frecuentes.get(entrada_usuario, None)

    # Si la respuesta no está en las preguntas frecuentes, intentar responder
    if respuesta is None:
        respuesta = responder_pregunta_no_frecuente(entrada_usuario)

    print("Chatbot:", respuesta)
```

Finalmente se unieron los códigos dados, y se le suministró al código un conjunto de preguntas frecuentes y respuestas establecidas mucho más amplio, así como un conjunto de palabras clave más grande, para que pueda identificar mayor cantidad de preguntas.



The screenshot shows a ChatGPT interface with a user prompt asking to combine two code snippets. The AI response provides a combined Python code snippet. The code includes a function `responder_pregunta_no_frecuente` for handling questions not in the frequent questions list, and a function `cargar_preguntas_frecuentes` for loading the frequent questions and answers. The `cargar_preguntas_frecuentes` function uses a dictionary to store the frequent questions and their corresponding answers.

```
python
# Función para responder preguntas que no están en las preguntas frecuentes
def responder_pregunta_no_frecuente(pregunta):
    # Busca palabras clave en la pregunta del usuario
    palabras_clave = ["activar", "plan de datos"]

    for palabra_clave in palabras_clave:
        if palabra_clave in pregunta:
            return "Puede activar su plan de datos llamando al servicio de a"

    return "Lo siento, no tengo información detallada sobre esa pregunta en"

# Función para cargar las preguntas frecuentes y respuestas
def cargar_preguntas_frecuentes():
    faq = {
        "¿Como activo mi plan de datos?": "Puedes activar tu plan de datos l",
        "¿Cuáles son los planes disponibles?": "Ofrecemos una variedad de pl",
        "¿Como puedo pagar mi factura?": "Puedes pagar tu factura en línea a"
    }
    # Agrega más preguntas y respuestas aquí

    return faq
```



## 4. Análisis y conclusiones

El chatBot desarrollado es un ejemplo de cómo el procesamiento de lenguaje natural puede mejorar la experiencia del usuario y brindar respuestas personalizadas de manera eficiente. A medida que continuamos explorando nuevas tecnologías y técnicas, estamos comprometidos a ofrecer un servicio que sea cada vez más intuitivo y valioso para los usuarios. Como el lenguaje natural al chatbot comprender y responder de manera más inteligente a preguntas complejas.

El desarrollo del chatbot ha sido un proceso integral que ha logrado satisfacer las necesidades de los usuarios de manera efectiva. La implementación de preguntas frecuentes proporciona respuestas rápidas y precisas, mientras que la integración del procesamiento de lenguaje natural (NLP) permite al chatbot abordar preguntas más complejas.

### 4.1. Aprendizaje por medio de ChatGPT

Aunque los resultados iniciales son positivos, se reconoce la importancia de la retroalimentación continua de los usuarios para ajustar y mejorar el chatbot. ChatGPT ha sido fundamental para facilitar el aprendizaje y la generación de respuestas contextuales, aunque la supervisión humana y la experimentación constante son necesarias para mantener la eficacia del modelo en evolución. En el futuro, la mejora continua del modelo, la diversificación de fuentes de información y la experimentación iterativa serán esenciales para asegurar que el chatbot siga siendo una herramienta valiosa y eficiente.