# CHATBOT USING PYTHON

## PHASE 3: DEVELOPMENT PART 1

## INTRODUCTION

In this phase, we embark on building our chatbot. The primary objective is to load and preprocess the dataset, set up the chatbot environment, and implement basic user interactions. Additionally, we will create a public URL to access the chatbot for testing and development purposes. This phase marks a significant step towards the development of a fully functional chatbot that can provide users with assistance and information.

## SETTING UP THE ENVIRONMENT

To begin, we are working in a Google Colab environment. Google Colab provides a cloud-based Python environment with access to various resources and simplifies tasks like mounting Google Drive, where we store our project files.

***Environment Preparation***

**Google Drive Connection**: We initiate the process by mounting our Google Drive to facilitate access to the essential files and datasets. This step is imperative when working in the Google Colab environment.

```
from google.colab import drive
drive.mount('/content/drive')
```

**Library Installation:** We are installing necessary Python libraries: 'transformers' for GPT-3 integration, 'flask' for web app development, and 'pyngrok' to create public URLs. These libraries are crucial for developing the chatbot.

```
!pip install transformers flask pyngrok
import os
import pandas as pd
from transformers import pipeline
from flask import Flask, request, jsonify
from pyngrok import ngrok
```

## DATASET LOADING AND PREPROCESSING

***Dataset Handling***

Our chatbot's success hinges on the quality of the training dataset. We load and preprocess a dataset comprising dialogues with questions and answers. This dataset acts as the foundation for training and fine-tuning our GPT-3 model to understand context and generate contextually relevant responses.

Our dataset is stored in a .txt file and consists of dialogues with two columns: questions and answers. We use the 'pandas' library to read the file.

```
file_path = '/content/drive/My Drive/colab/dialogs.txt'

df = pd.read_csv(file_path, sep='\t', names=['Question', 'Answer'])
```

At this stage, we are focused on the basic loading and organization of the dataset. More advanced preprocessing will be applied in the subsequent phases as needed.

### *GPT-3 Pipeline Initialization*

We initialize the GPT-3 pipeline, a critical component for our chatbot's language understanding and response generation. For this phase, we employ the 'EleutherAI/gpt-neo-2.7B' model from the Transformers library, known for its vast pre-trained knowledge and natural language understanding capabilities.

```
generator = pipeline('text-generation', model='EleutherAI/gpt-neo-2.7B')
```

## CHATBOT APPLICATION

### *API Key Configuration*

A pivotal element in our chatbot's functionality is the secure integration with the OpenAI GPT-3 API. To ensure this integration is seamless and secure, we set the GPT-3 API key as an environment variable. This approach safeguards the API key from accidental exposure in the codebase and ensures a higher level of security in our chatbot implementation.

```
os.environ['API_KEY'] = 'YOUR_API_KEY'
```

### *Flask Web Application Creation*

For users to interact with the chatbot, we need a web interface. The creation of our chatbot application begins with the selection of Flask as the foundational framework. Flask's simplicity, lightweight nature, and modularity make it an ideal choice for creating a microservice responsible for chatot interactions.

```
from flask import Flask, request, jsonify

app = Flask(__name__)
```

### *User Interaction Handling*

We initialize a Flask app using the following code: We implement a dedicated route within our Flask application to manage user interactions. This route processes user messages, analyzes them forcontext, and triggers the GPT-3 model to generate relevant responses. The Flask application is designed to provide a seamless and responsive user experience.

```
@app.route('/chat', methods=['POST'])
def chat():
    if 'user_message' in request.form:
        user_message = request.form['user_message']
        # Generate a bot response using the GPT-3 model
        bot_response = generator(user_message, max_length=50, do_sample=True)[0]['generated_text']
        return jsonify({'bot_response': bot_response})
    else:
        return jsonify({'error': 'Invalid request'})
```

## CREATING A PUBLIC URL FOR CHATBOT DEPLOYMENT:

In this section of the code, we set up Ngrok, a tool that allows us to create a public URL for our Flask web application, which hosts the chatbot. By using **ngrok.connect**, we generate a unique URL that redirects to our local development environment. This public URL allows users to interact with the chatbot from anywhere on the internet.

```
if __name__ == '__main':
    public_url = ngrok.connect(addr='5000')
    print(' * ngrok tunnel "' + public_url.public_url + '" -> "http://127.0.0.1:5000"')
    app.run(host='0.0.0.0', port=5000)
```

## CHATBOT TESTING

```
import requests
# Define the URL for chatbot interactions
chatbot_url = 'https://a3f3-34-168-145-174.ngrok.io/chat'
# Create a user message (change this as needed)
user_message = 'Hello, chatbot!'
# Send a POST request to chat with the chatbot
response = requests.post(chatbot_url, data={'user_message': user_message})
# Print the chatbot's response
print(response.json())
```

## *External Script for Testing*

To validate our chatbot's functionality, we employ an external script for user interaction simulations. This script sends user messages, captures chatbot responses, and facilitates testing to ensure coherent and contextually relevant answers. Thorough testing is a critical step in refining the chatbot's performance and ensuring that it delivers high-quality responses.

## CONCLUSION

In this part of the development process, we successfully set up the chatbot environment, loaded and preprocessed the dataset, created a Flask web application, and made the chatbot accessible online via a public URL. This sets the stage for more advanced functionality and interactions in the upcoming phases.

**BY:**

SHWETHA V

ROOBINI M

GOVINDALALITHA T M

HARINI PRIYAA S