



```
pip install googletrans==4.0.0-rc1
```

 Show hidden output

```
pip install gtts
```

 Collecting gtts
 Downloading gTTS-2.5.4-py3-none-any.whl.metadata (4.1 kB)
Requirement already satisfied: requests<3,>=2.27 in /usr/local/lib/python3.11/dist-packages (from gtts) (2.32.3)
Requirement already satisfied: click<8.2,>=7.1 in /usr/local/lib/python3.11/dist-packages (from gtts) (8.1.8)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.27->gtts) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.27->gtts) (2.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.27->gtts) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.27->gtts) (2025.1.31)
Downloading gTTS-2.5.4-py3-none-any.whl (29 kB)
Installing collected packages: gtts
Successfully installed gtts-2.5.4

```
import requests
from bs4 import BeautifulSoup
import nltk
nltk.download('punkt')
nltk.download('punkt_tab')
nltk.download('stopwords')
from nltk.corpus import stopwords
from googletrans import Translator
import spacy
from gtts import gTTS
```

 [nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data] Package punkt_tab is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Package stopwords is already up-to-date!

1. Write a well-structured paragraph describing any chatbot of your choice. Include details such as its purpose, features, working mechanism, and real-world applications. Save this content as a text file named "chatbot.txt".

```
url = "https://en.wikipedia.org/wiki/ChatGPT" # Replace with the actual URL
response = requests.get(url)


soup = BeautifulSoup(response.text, "html.parser")

body_content = soup.find(class="mw-content-ltr mw-parser-output")
text = body_content.get_text(strip=True, separator="\n") if body_content else "Content not found."

with open("chatbot.txt", "w", encoding="utf-8") as f:
    f.write(text)
```


2. Using Python, process the text extracted from the file by splitting it into individual sentences and words. Display the extracted sentences and words separately to observe the structure of the content.

```
f = open('chatbot.txt')
lines = f.read()
sentences = nltk.sent_tokenize(lines)
print(sentences)
```

 ['Chatbot developed by OpenAI\nChatGPT\nDeveloper(s)\nOpenAI\nInitial release\nNovember\xa030, 2022\n(2 years ago)\n(\n2022-11-30\n)\nStable release\nJanuary\xa027, 2025\n(14 months ago)\n\n']

```
total = []
for sentence in sentences:
    words = nltk.word_tokenize(sentence)
    total+=words


print(total)
```

 ['Chatbot', 'developed', 'by', 'OpenAI', 'ChatGPT', 'Developer', '(', 's', ')', 'OpenAI', 'Initial', 'release', 'November', '30', ', ', '2022', '(', '2', 'years', 'ago', ')', ' ', '\n']


3. Identify any stop words (common words such as "the," "is," "in," etc.) within the extracted words and remove them from the text. You can use the Natural Language Toolkit (NLTK) or any other suitable library to filter out these stop words.

```
nonstop = [i for i in total if i not in stopwords.words('english')]

print(total)
```

 ['Chatbot', 'developed', 'by', 'OpenAI', 'ChatGPT', 'Developer', '(', 's', ')', 'OpenAI', 'Initial', 'release', 'November', '30', ', ', '2022', '(', '2', 'years', 'ago', ')', ' ', '\n']

```
stop = [i for i in total if i in stopwords.words('english')]
print("stopwords are :", stop)
```

 stopwords are : ['by', 's', 'is', 'a', 'by', 'and', 'in', 'is', 'on', 'the', 'can', 'and', 'to', 'and', 'a', 'a', 'of', 'and', 'is', 'with', 'the', 'which', 'has', 'to', 'in']

4. Implement a Python script to count and display the number of stop words and non-stop words in the text. Provide a summary of the word count, indicating how many words were classified as stop words and how many were considered meaningful content words.

```
print("length of non-stopwords",len(nonstop))
print("length of stopwords", len(stop))
```

length of non-stopwords 20914
length of stopwords 4406

5. Select any ten words from the processed text (excluding stop words) and translate them into your native language using Python. You may use an external translation library such as Google Translate API for this task. Display the original words along with their translated equivalents.

```
data = nonstop[95:106]
s = " ".join(data)
print("words to be translated : ", s)

translator = Translator()

translated_text = translator.translate(s, dest="ml")
print("Translated text : ",translated_text.text)
```

words to be translated : generate human-like conversational responses enables users refine steer conversation towards desired
Translated text : മനുഷ്യ പോലുള്ള സംഭാഷണ പ്രതികരണങ്ങൾ സൃഷ്ടിക്കുക ഉപയോക്താക്കളെ ആവശ്യമുള്ള സംഭാഷണം പരിഷ്കരിക്കുന്നത് പ്രാപ്തമാക്കുന്നു

```
for i in nonstop[95:106]:
    print(f"{i} : {translator.translate(str(i), dest = 'ml').text}")
```

generate : ഉത്പാദിപ്പിക്കുക
human-like : മനുഷ്യൻ സമാനമാണ്
conversational : സംഭാഷണം
responses : പ്രതികരണങ്ങൾ
enables : പ്രവർത്തനക്ഷമമാക്കുന്നു
users : ഉപയോക്താക്കൾ
refine : ശുദ്ധീകരിക്കുക
steer : നയിക്കുക
conversation : സംഭാഷണം
towards : നേരെ
desired : ആഗ്രഹിച്ചു

6. Identify and extract all named entities (such as names of chatbot models, company names, and technologies) from the text using Named Entity Recognition (NER) techniques.

```
nlp = spacy.load("en_core_web_sm")

s = " ".join(nonstop[:100])
doc = nlp(s)

print("\nNamed Entities:")
for ent in doc.ents:
    print(f"{ent.text}: {ent.label}")
```

/usr/local/lib/python3.11/dist-packages/spacy/util.py:1740: UserWarning: [W111] Jupyter notebook detected: if using `prefer_gpu()` or `require_gpu()`, include it in the same cell
warnings.warn(Warnings.W111)

Named Entities:
OpenAI ChatGPT Developer: ORG
OpenAI Initial: ORG
November 30 , 2022: DATE
2 years ago: DATE
2022-11-30: DATE
January 27 , 2025: DATE
14 days ago: DATE
2025-01-27: DATE
1: CARDINAL
OpenAI: GPE
Platform Cloud: PERSON
Type Chatbot Large: PERSON
Generative: PERSON
License Proprietary: ORG
2: CARDINAL
3: CARDINAL
OpenAI: GPE
2022: DATE

7. Convert the chatbot description into speech using a TexttoSpeech (TTS) library such as pyttsx3 or gTTS and save the audio file as "chatbotspeech.mp3".

```
tts = gTTS(text=s, lang='en')
tts.save("chatbot_speech.mp3")
print("\nAudio file saved as 'chatbot_speech.mp3'")
```

Audio file saved as 'chatbot_speech.mp3'

