

BUILDING SMART JULIE WITH PYTHON AND OPENAI API



Contextualization

It seems that the title of this article has piqued your curiosity. Yes, this article was created for this purpose. I think the first question for you is who is smart **Julie**? That's exactly what I wanted you to ask because our main goal is to answer what smart **Julie** is. However, in a nutshell, smart **Julie** is a friend who, in addition to being kind, is also very intelligent, capable of answering all your questions and even giving you advice if you ask for it.

Technically, smart Julie will be a **virtual assistant** that we will build with **python** and **openai API**. So, we will be able to answer any question that is put to her, both academically and socially. However, the question has to be well formulated and the pronunciation of the words sufficient in the language that she intends to use. Therefore, in this project we will use English but any internationally recognized language can be used as well.

What do we need to complete this project?

The first step for this project is to know how to program with **python**, because the knowledge of installing libraries and connecting to an **API** will be essential. But don't worry because we will explain it step by step. So, you just need to follow all the steps and at the end we will provide the link from GitHub with source code.

1. Necessary tools

The tools needed for this project are **Python** programming language and following libraries:

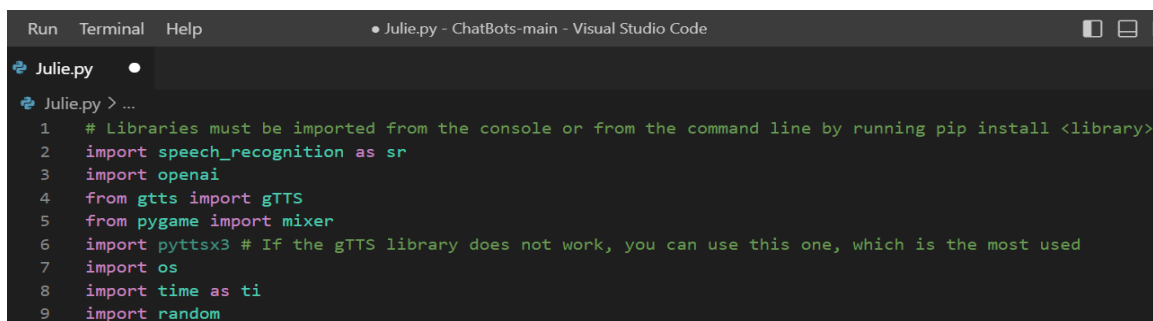
- **SpeechRecognition** is a **Python** library that allows you to recognize human speech and convert it into text. In particular, this can be used in various applications, such as **virtual assistants** and audio transcription. To use the library, you need to have a microphone connected to your computer and install the library using a package manager like pip.
- **OpenAI API** is an application programming interface (**API**) made available by OpenAI. This way developers can use the artificial intelligence models developed by the company in their own applications.
- **GTTS** stands for "Google Text-to-Speech" and is a free **API** made available by Google to convert text to speech. The **API** allows developers to integrate speech synthesis functionality into their own applications, using a variety of voices and languages.
- **Pygame** is a set of **Python** libraries aimed at game and multimedia development. Specifically, libraries include modules for window and event management, 2D graphics, image manipulation, audio, keyboard and mouse input, collision detection, and much more. In our case, we will use it for audio manipulation.
- **Pytttsx3** is based on different speech synthesis engines like SAPI5 (Speech **API** 5) on Windows and NSSpeechSynthesizer on macOS. In addition, it supports multiple languages and voices, as well as real-time audio playback.
- **OS** : the **os** module is a standard **Python** library that provides a way to interact with the operating system. Actually, it allows you to work with the file system, get information from the environment, manage processes, change environment variables and much more.
- **TIME** the time module is a **Python** standard library that allows you to work with functions related to time, such as measuring durations. This is waiting for a certain period before executing a certain action, formatting dates and times, among other features.
- **Random**: the random module is a standard **Python** module that provides functions for generating pseudorandom numbers. In fact, these functions are useful for various applications such as games, simulations, and cryptography.

- **Text editor** is where we write our code in this case, we use vscode but you can choose your preferences

2. Installation and import of libraries

Before importing the libraries make sure that you have python installed on your computer and that your microphone is operational because some libraries require it. Regarding the procedures for installing **python** and libraries, find in this [article](#) where we present all the details for the purpose, below we present the imports.

3. Coding



```
Run Terminal Help • Julie.py - ChatBots-main - Visual Studio Code
Julie.py
Julie.py > ...
1 # Libraries must be imported from the console or from the command line by running pip install <library>
2 import speech_recognition as sr
3 import openai
4 from gtts import gTTS
5 from pygame import mixer
6 import pytsx3 # If the gTTS library does not work, you can use this one, which is the most used
7 import os
8 import time as ti
9 import random
```

Next we call our **API** key in the form of a string

```
openai.api_key = "paste your api key here"
```

We define the function that transforms the voice captured in the mic to text

```

15 # We define the function that transforms the voice captured in the mic to text
16 def transform_audio_in_text():
17
18     r = sr.Recognizer()
19     with sr.Microphone() as origen:
20         r.pause_threshold = 0.9
21         print("Speak something please!")
22         audio = r.listen(origen)
23         try:
24             # The language with which the voice is recognized must be specified
25             request = r.recognize_google(audio, language="en-EN")
26             print("You: " + request)
27             return request
28
29         except sr.UnknownValueError:
30             print("Ups, Sorry I didn't understand!")
31             return "Please can repeat?"
32
33         except sr.RequestError:
34             print("Ups, no service!")
35             return "waiting"
36
37         except:
38             print("Ups, something are wrong!")
39             return "waiting"

```

Afterwards, we define the function that will transform the text (message) into audio, I leave both for gTTS library and for pyttsx3.

```

# We define the function that will transform the text (message) into audio, I leave both for gTTS library and for pyt
def speak (mensaje):
#This section of code is for gTTS library **
    volume = 0.7
    tts = gTTS(mensaje, lang="en", slow=False)
    ran = random.randint(0,9999)
    filename = 'Temp' + format(ran) + '.mp3'
    tts.save(filename)
    mixer.init()
    mixer.music.load(filename)
    mixer.music.set_volume(volume)
    mixer.music.play()

    while mixer.music.get_busy():
        ti.sleep(0.3)

    mixer.quit()
    os.remove(filename)

```

***** This section of code is for pyttsx3 library, just use one removing the numeral

In case the gTTS library doesn't work

Start the pyttsx3 engine

#engine = pyttsx3.init()

Slow down playback speed, default is 200

#engine.setProperty('rate', 150)

speak message

#engine.say(message)

engine.runAndWait()

```
71 def despedida():
72     speak("See you, have a great day!")
73     exit()
74 def main():
75     conversation = ""
76     despedida_phrases = ["tchau", "thank you by", "see you later", "see you soon", "i have to go", "end", "see off"]
77     speak ("Hello I'm Julie, welcome, I'm willing to answer all of exiyour questions, please let's start the conversa
78     while True:
79         question = transform_audio_in_text().lower()
80         conversation += "\nYou: " + question + "\nJulie:"
81         response = openai.Completion.create(
82             model="text-davinci-003",
83             prompt=conversation,
84             temperature=0.5,
85             max_tokens=3500,
86             top_p=0.5,
87             frequency_penalty=0.7,
88             presence_penalty=0.0,
89             stop=["\n", " You:", " Julie:"]
90         )
91         answer = response.choices[0].text.strip()
92         conversation += answer
93         print("Julie: " + answer + "\n")
94         speak(answer)
95         # Check if any of the despedida phrases was said to stop the conversation
96         if any(frase in question for frase in despedida_phrases):
97             despedida()
98     main()
```

FULL CODE

```
Go Run Terminal Help • Julie.py - Smart_Julie - Visual Studio Code
test.py Julie.py Los.py ls.py testes.py 2
Julie.py > main
1 # Libraries must be imported from the console or from the command line by running pip install <library>
2 import speech_recognition as sr
3 import openai
4 from gtts import gTTS
5 from pygame import mixer
6 import pyttsx3 # If the gTTS library does not work, you can use this one, which is the most used
7 import os
8 import time as ti
9 import random
10
11
12 openai.api_key = "sk-8XDtCbMOL5ZEbg14Q1YvT3B1bkFJN5zsZoNuqtBXNQcaSdeA"
13
14
15 # We define the function that transforms the voice captured in the mic to text
16 def transform_audio_in_text():
17
18     r = sr.Recognizer()
19     with sr.Microphone() as origen:
20         r.pause_threshold = 0.9
21         print("Speak something please!")
22         audio = r.listen(origen)
23         try:
24             # The language with which the voice is recognized must be specified
25             request = r.recognize_google(audio, language="en-EN")
26             print("You: " + request)
27             return request
28
29         except sr.UnknownValueError:
30             print("Ups, Sorry I didn't understand!")
31
32             return "Please can repeat?"
33
34         except sr.RequestError:
35             print("Ups, no service!")
36             return "waiting"
37
38         except:
39             print("Ups, something are wrong!")
40             return "waiting"
41
42 # We define the function that will transform the text (message) into audio, I leave both for gTTS library and for pyt
43 def speak (mensaje):
44     #This section of code is for gTTS library **
45     volume = 0.7
46     tts = gTTS(mensaje, lang="en", slow=False)
47     ran = random.randint(0,9999)
48     filename = 'Temp' + format(ran) + '.mp3'
49     tts.save(filename)
50     mixer.init()
51     mixer.music.load(filename)
52     mixer.music.set_volume(volume)
53     mixer.music.play()
54
55     while mixer.music.get_busy():
56         ti.sleep(0.3)
57
58     mixer.quit()
59     os.remove(filename)
```

```

71 def despedida():
72     speak("See you, have a great day!")
73     exit()
74 def main():
75     conversation = ""
76     despedida_phrases = ["tchau", "thank you by", "see you later", "see you soon", "i have to go", "end", "see off"]
77     speak ("Hello I'm Julie, welcome, I'm willing to answer all of exiyour questions, please let's start the conversa
78     while True:
79         question = transform_audio_in_text().lower()
80         conversation += "\nYou: " + question + "\nJulie:"
81         response = openai.Completion.create(
82             model="text-davinci-003",
83             prompt=conversation,
84             temperature=0.5,
85             max_tokens=3500,
86             top_p=0.5,
87             frequency_penalty=0.7,
88             presence_penalty=0.0,
89             stop=["\n", " You:", " Julie:"]
90         )
91         answer = response.choices[0].text.strip()
92         conversation += answer
93         print("Julie: " + answer + "\n")
94         speak(answer)
95         # Check if any of the despedida phrases was said to stop the conversation
96         if any(frase in question for frase in despedida_phrases):
97             despedida()
98     main()

```

Final considerations

Once you have completed all the steps, you can talk to our assistant **Julie** and she is already in a position to answer your questions. So, don't lose time! Find the source code [here](#) and test it on your own computer and have fun. Remember that the next article will be much more interesting because we are going to convert Julie into an installable setup on your computer or smartphone don't miss it out!

Keywords

[Julie](#), [openAI](#), [python](#), [API](#), [Virtual assistant](#)

Reference List

1. <https://pypi.org/project/SpeechRecognition/>
2. <https://openai.com/research/overview>
3. <https://gtts.readthedocs.io/en/latest/>
4. <https://www.pygame.org/docs/>
5. <https://pytttsx3.readthedocs.io/en/latest/>
6. <https://stackoverflow.com/questions/4813238/difference-between-subprocess-popen-and-os-system>
7. <https://docs.python.org/3/library/time.html>
8. <https://docs.python.org/3/library/random.html>