**Create an AI voice assistant using python, whisper AI and openai API.**

[[Sangeeth Joseph - The AI dev](https://medium.com/@sangeeth123sj?source=post_page-----b9814bbfd8e6--------------------------------)](https://medium.com/@sangeeth123sj?source=post_page-----b9814bbfd8e6--------------------------------)

[Sangeeth Joseph - The AI dev](https://medium.com/@sangeeth123sj?source=post_page-----b9814bbfd8e6--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F34f2ddc7dbd6&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40sangeeth123sj%2Fcreate-an-ai-voice-assistant-using-python-whisper-ai-and-openai-api-b9814bbfd8e6&user=Sangeeth+Joseph+-+The+AI+dev&userId=34f2ddc7dbd6&source=post_page-34f2ddc7dbd6----b9814bbfd8e6---------------------post_header-----------)

5 min read

·

Aug 26

12

1



With the rise of ChatGPT, I’ve always been chatting with it to quench my thirst for more knowledge in terms of webdevelopment and other things.

I’ve been chatting too long, now I wanna talk in voice (just like we expect with real people we chat with ;) )

**Let’s call this new voice assistant that we creating as AIVA: Artificial Intelligence Voice Assistant**

**The Plan:**

1. Our speech to text input — WhisperAI
2. Text input to AI asistant — OpenAI
3. Text output from AI assistant to speech — Pyttsx3 (free python library though not the best voices)

**Steps:**

1. Setup work folder and virtual environment, install dependencies.
2. create main.py with all 3 parts of the workflow and a .env file for the openai key.
3. Run and debug!

So let’s start!

**1. Setup work folder and virtual environment, install dependencies.**

I use terminal for this purpose.

Go to terminal create new folder using command :

mkdir <name of project>

enter the folder

cd <name of project>

create and activate virtual environment for installing dependencies.

#create environment  
python3 -m venv env  
#activate environment  
source env/bin/activate

install dependencies of openaAI, pyttsx3 and whiserAI using pip:

pip install openai  
pip install pyttsx3  
pip install python-dotenv  
pip install python-sounddevice

**Dotenv** is required to access the apikey in the main script from the .env file where we store the key as environment variable (helps in security of the api key)

**Sounddevice** is for recording the voice input into wav file to pass to gpt 3.5

**Pyttsx3** is for converting text to speech to get output voice.

open the folder now in vs code using

code .

**2. Create main.py with the code**

We have already seen the main.py code for creation of a python text input and output assistant like Chatgpt using a simple python script in part 1.

You can refer to the part 1 here: <https://python.plainenglish.io/create-an-ai-assistant-using-python-and-openai-api-bc00876f3c8a>

I’m adding it here again:

import os  
import openai  
#importing dotenv to access the environment variable open ai key #from .env file  
from dotenv import load\_dotenv  
load\_dotenv()  
openai.api\_key = os.getenv("OPENAI\_API\_KEY")  
  
#using the while loop to continously take inputs from user and  
#give responses like in a chat  
while True:  
 user\_input = input("You: ") # Get user input  
  
 # Create a list of messages with the user's input  
 messages = [  
 {"role": "system", "content": "You are a helpful assistant."},  
 {"role": "user", "content": user\_input}  
 ]  
  
 # Make the API call  
 completion = openai.ChatCompletion.create(  
 model="gpt-3.5-turbo",  
 messages=messages  
 )  
  
 # Print the assistant's response  
 print("Assistant:", completion.choices[0].message["content"])  
  
 # Ask whether to continue or stop  
 user\_choice = input("Continue? (y/n): ")  
 if user\_choice.lower() != "y":  
 print("Glad to help bye!")  
 break # Exit the loop if the user doesn't want to continue

Let’s also get the openAI api key from the openAI website : <https://platform.openai.com/>

We can get this key by first opening the api keys option from the top left profile icon drop down, also explaing with pictures in part 1.

Let’s create the .env file now  
and add the openai key there:

OPENAI\_API\_KEY=<api key>

Now let’s modify the code in main.py to receive voice inputs and give voice outputs using Whisper AI and pyttsx3 respectively!

the code block to use whisperAI to get voice input and transcribe it into text:

# Note: you need to be using OpenAI Python v0.27.0 for the code below to work  
import openai  
audio\_file= open("/path/to/file/audio.mp3", "rb")  
transcript = openai.Audio.transcribe("whisper-1", audio\_file)

But this only takes already available voice file mp3 as of now, Let’s use sounddevice library for recording live voice.

import sounddevice as sd  
from scipy.io.wavfile import write  
  
fs = 44100 # Sample rate  
seconds = 3 # Duration of recording  
  
myrecording = sd.rec(int(seconds \* fs), samplerate=fs, channels=2)  
sd.wait() # Wait until recording is finished  
write('output.wav', fs, myrecording) # Save as WAV file

Now that we have both the live recordind (using sound device), speech to text (using whisper AI), AI model (gpt 3.5 Openai) and output text to speech using pyttsx3

Let’s create the final modified code of main.py:

import numpy as np  
import openai  
import random  
from scipy.io.wavfile import write  
import sounddevice as sd  
import pyttsx3  
  
# Set your OpenAI API key here  
from dotenv import load\_dotenv  
load\_dotenv()  
openai.api\_key = os.getenv("OPENAI\_API\_KEY")  
  
# Global variable to store audio data  
audio = []  
  
# Adjectives to generate random names for voices  
adjectives = ["beautiful", "sad", "mystical", "serene", "whispering", "gentle", "melancholic"]  
nouns = ["sea", "love", "dreams", "song", "rain", "sunrise", "silence", "echo"]  
#initializing pytts for text to speech output  
engine = pyttsx3.init()  
engine.setProperty('rate', 130)  
  
   
def generate\_random\_name():  
 # to generate random unique names for the audio voice recordings  
 adjective = random.choice(adjectives)  
 noun = random.choice(nouns)  
 return f"{adjective} {noun}"  
  
def new\_record\_audio():  
 # to record audio as wav file  
 print("Recording... Press 's' to stop.")  
 fs = 44100  
 seconds = 6  
 myrecording = sd.rec(int(seconds \* fs), samplerate=fs, channels=2)  
 sd.wait() # Wait until recording is finished  
 audio\_name = generate\_random\_name()  
 write(f'./{audio\_name}.wav', fs, myrecording) # Save as WAV file   
 print("Recording stopped.")  
 return f'./{audio\_name}.wav'  
   
def transcribe\_audio(audio\_path):  
 print ("entered transcribe", "./"+audio\_path)  
 audio\_file= open(audio\_path, "rb")  
 print(audio\_file)  
 transcript = openai.Audio.transcribe("whisper-1", audio\_file)  
 print(transcript)  
 return transcript['text']  
  
  
  
def speech\_to\_text(response):  
 # to generate the final output voice from text  
 engine.say(response)  
 engine.runAndWait()  
  
  
  
  
def main():  
 while True:  
 print("Press 's' to stop recording and transcribe the audio.")  
 # Start recording live voice input  
 recorded\_audio\_path = new\_record\_audio()  
 print("Recording stopped. Transcribing audio...")  
 # Save the recorded audio as a WAV file  
 print("Recorded audio saved to:", recorded\_audio\_path)  
 print("----end---")  
 # Transcribe the audio  
 transcript = transcribe\_audio(recorded\_audio\_path)  
 # Create a list of messages with the user's input  
 messages = [  
 {"role": "system", "content": "You are a helpful assistant."},  
 {"role": "user", "content": transcript}  
 ]  
 print("Transcript:")  
 print(transcript)  
 # Make the API call for gpt AI  
 completion = openai.ChatCompletion.create(  
 model="gpt-3.5-turbo",  
 messages=messages  
 )  
 response = completion.choices[0].message["content"]  
 # Print the assistant's response  
 print("Assistant:", response)  
   
 # Convert output to voice  
 speech\_to\_text(response)  
  
 # Ask whether to continue or stop  
 user\_choice = input("Continue? (y/n): ")  
 if user\_choice.lower() != "y":  
 print("Glad to help bye!")  
 break # Exit the loop if the user doesn't want to continue  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()  
  
#Replace "YOUR\_OPENAI\_API\_KEY" with your actual OpenAI API key. Run the script, and it will start recording your voice input. Press the 's' key when you're #done speaking to stop the recording and transcribe the audio using OpenAI's Whisper API. The transcribed text will be displayed in the console.

And that’s it!!  
with that we have the updated our text input/output AI assistant from part 1 to a voice AI assistant! My own Jarvis lol.  
Thanks for reading till the end. My name is Sangeeth Joseph, I’m a fullstack freelance developer working on upwork. Follow me for more interesting AI projects and news.