Personal AI Assistant

Python

September24,2022

ABASTRACT: -

An AI personal assistant is a piece of software that understands verbal or written commands and completes task assigned by the client. Virtual assistants are Artificial Intelligent based programs. Such as Siri, Alexa and Google.

This AI assistant is a smart computer program that understands human natural languages through voice commands or text and performs tasks for the user. Unlike the big names mentioned earlier like, Alexa and Siri, our program can perform only certain tasks that are defined already. Still, it can perform most frequent computer operations that we need in our day-to-day life to make our lives lot easier.

ACKNOWLEDGEMENT: -

I would like to thank my mentor – My senior Mukesh for his advice and inputs on this project. Many thanks to my friends and seniors as well, who spent countless hours to listen and provide feedbacks.

TABLE OF CONTENT

INTRODUCTION: -

1.1 CONTEXT

This project has been done as part of my course for the CSE(H) at Lovely Professional University. Supervised by Mr.Mukesh, I had two months to fulfil the requirements in order to succeed the module.

1.2 MOTIVATION

The motivation for AI assistant has been drawn on from three major sources Google's assistant, MCU and Seniors Suggestions. As mentioned above, the major burst of motivation was Google's Home assistant and MCU. I've been always wondering that can I make AI assistant act to my commands in the manner I want it to. Always discussing about the same along with my associates. No longer it took me to figure out that this customization can be performed effortlessly by use "Python" language and some of the libraries. Being a CS student, it was an ambition to make one and today's it's here in the form than it was ever before.

1.3 IDEA

In our day-to-day life we have to perform lots of action to get our tasks done. Out of all the action we do some are most common and are replicated on daily basis. So, if we can make our conscious mind get rid of these fundamental activities so that it can work more freely and creatively on some of the most vital tasks. So, I created my very own personal AI assistant that can follow up all the commands like I want it to be. As a first project, I wanted my project to cover all the grounds by being capable of performing all the tasks that we need frequently in our day-to-day life. So as the goal was to perform daily tasks, I dig up in my own life and tried to find out what are the most commonly used operations that makes my life easier and then in-carved them in my program. Some of the most common tasks such as, google search, YouTube search, taking screenshots, opening websites, playing music, opening any folder etc., can be performed effortlessly by my AI assistant.

Proposed Structure:

Our personal AI assistant is built on base on three major functions: -

a. **Speak() function: -**
   This function takes string as an argument and by using pyttsx3 module.

b. **Takecommand() function: -**
   Takecommand() func. is used to take command from user. It is an argument less function and this function uses speech_recognition module and its other API's for performing its functions.
   This function returns string in lowercase.

c. **Taskexe() function: -**
   Taskexe() func. is used to execute the task asked by user and this function is composed is composed of several if-elif-else statement which are executed on the basis of what string is returned by takecommand() function.

LIBRARIES & Functions:

1. **pyttsx3: -**
   It is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance.
   a. Init function: - Construct a new engine instance or reuses the existing instance for the driver's name (here sapi5).
   b. Getproperty: - This function is used to get the current value of property (engine). This function returns all the voices that are present in our system.
   c. Setproperty: -This function is used to set a specific property out of all the properties that are present with. Some of major properties includes-
      i. Voices- It is an array of the Id's of voices where you have stored all the voices present in our system.
      ii. Rate- Defines the speed of the engine, how fast or slow our engine will speak up things.

2. **Speech_recoginition: -**
   It is a library for performing speech recognition, with support for several engines and APIs, online and offline mode.
   a. Microphones(): - It is a class that is present inside speech recognition library and it creates a new Microphone instance, which represents a physical microphone on computer.
   b. Recognizer(): - It is a class that is present inside speech recognition library. Creates a new recognizer instance, which represents a collection of speech recognition functionality.
   c. Listen(): - records a single phrase from source into "AudioData" instance which is then returned.
   d. Recognize_google(): - Performs speech recognition on audio data instance, using the google Speech Recognition API.

b. Webbrowser: -

The webbrowser module provides a high-level interface to allow displaying web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing. It accepts a URL as the argument.

Used commands: -
search of YouTube using browser, launching websites, to open your location, open app and close app.

c. OS: -

The OS module in Python provides functions for interacting with the operating system. This module provides a portable way of using operating system-dependent functionality.

Used commands: -
openapp(), closeapp(), screenshots().

Used functions: -

A. Startfile(): - This function is used to open any an app present at particular locations.
B. System(): - This function is used to close particular apps that are already opened. To close we have to use “TASKKILL /F /im app.exe” as an argument.

d. pywhatkit: -

pywhatkit is a Python library for sending what’s app messages at certain time, play YouTube video, perform a google search and get information on particular topic.

Used Commands: - WhatsApp() and music().

Used Functions: -

A. Sendwhatmsg_instantly(): -As the name suggests function is used to instantly send the message. Takes number of receiver and message as the arguments.
B. Playonyt(): - This function is used to stream anything on youtube. It searches for the string present in its argument and plays first video rectified after search.

e. **time: -**

Python has a module named time to handle time-related tasks. The time function returns the number of seconds passed since epoch.

Used Command: -
to set the alarm.

f. **Pytz: -**

This module serves the date-time conversion functionalities and helps user serving international client's base. It enables time-zone calculations in our Python applications and also allows us to create time zone aware datetime instances.

Used Command: -
To set the alarm.

g. **Wikipedia: -**

Wikipedia is a multilingual online encyclopaedia created and maintained as an open collaboration project by a community of volunteer editors using a wiki-based editing system. In order to extract data from Wikipedia, we must first install the Python Wikipedia library, which wraps the official Wikipedia API.

Used Command: -
search on Wikipedia.

Used functions: -

a. Summary(): - This function takes the two arguments, first the topic to be searched and number of lines to be printed or uttered.

h. **Pyautogui: -**
Pyautogui lets your Python scripts control the mouse and keyboard to automate interactions with other applications.

Used Commands: -
to take screenshots.

Used Functions: -

A. Screenshot(): - This function takes screenshot of the window being opened.
B. Save(): - Takes the location URL as its argument to save screen shot.

i. **Keyboard: -**
Python provides a library named keyboard which is used to get full control of the keyboard. It's a small Python library which can hook global events, register hotkeys, simulate key presses and much more.

Used Commands: -
Chrome automation, mute, play, restart, pause, Fullscreen etc.

Used Functions: -

A. Press(): - To select any particular key from keyboard.

B. Press_and_release(): - To select multiple keys one after another.

j. **Pyjokes: -**
Python supports creation of random jokes using one of its libraries. Let us explore it a little more, pyjokes is a python library that is used to create one-line jokes for programmers. Informally, it can also be referred as a fun python library which is pretty simple to use.

Used Command: -
tell me a joke.

Used Functions: -
A. Getjokesv: - Method used to get random jokes from pyjokes module.

k. **Datetime: -**
Python has a module named datetime to work with dates and times. One of the classes defined in the datetime module is datetime class. We then used now() method to create a datetime object containing the current local date and time.

Used Command: -
set alarm.

l. Translator(): -
Comes under google trans API. As the name suggests it is used to translate text.

Used Commands: -
open translator.

m. **PyDictionary(): -**

This module is used to get meaning, antonyms and synonyms using meaning(), antonym() and synonym() functions respectively. These take string as its argument and then perform functions as their name suggests on string.

Used Command: -
meaning of word.

n. **Speedtest: -**

Module is used to audit the data connection speed.

Used Command: -
internet speed.

Used Functions: -
A. Download(): - Function used to get down loading speed.
B. Upload():- Function used to get up loading speed.

Screenshots:

```
def takecommand() :
    command  = sr.Recognizer()
    with sr.Microphone() as source:
        print ("Listening......")
        command.pause_threshold = 0.8
        audio = command.listen(source)
        try:
            print("Recognising....")
            query = command.recognize_google(audio, language = 'en-in' )
            print(f"You Said : {query} ")
        except Exception as Error:
            speak("Not able to recognize. Please Say again")
            return "None"
        return query.lower()
```

Figure 1 (takecommand function)

```
def speak(audio):
    print("    ")
    Assistant.say(audio)
    print(f": {audio}")
    print('    ')
    Assistant.runAndWait()
```

Figure 2 (speak function)

```
def TaskExe():

    def Music():...
    def Whatsapp():...
    def openapps():...
    def closeapps():...
    def ChromeAuto():...
    def Dict():...
    def screenshot():...
    def TakeHindi():...
    def Trans():...
    def SpeedTest():...

    while True:
        query = takecommand()
        if 'hello' in query:...
        elif 'what is your name' in query:...
        elif 'how are you' in query:...
        elif 'you need a break' in query:...
        elif 'search on youtube' in query:...
        elif 'website' in query:...
        elif 'launch'in query:...
        elif 'play music' in query:...
        elif 'wikipedia'in query:...
        elif 'send a whatsapp message' in query:...
        elif 'screenshot' in query:...
        elif 'open facebook' in query:...
        elif 'open opera' in query:...
        elif 'open firefox' in query:...
        elif 'open google maps' in query:...
        elif 'open vs code' in query:...
```

Figure 3.1 (TaskExe function)

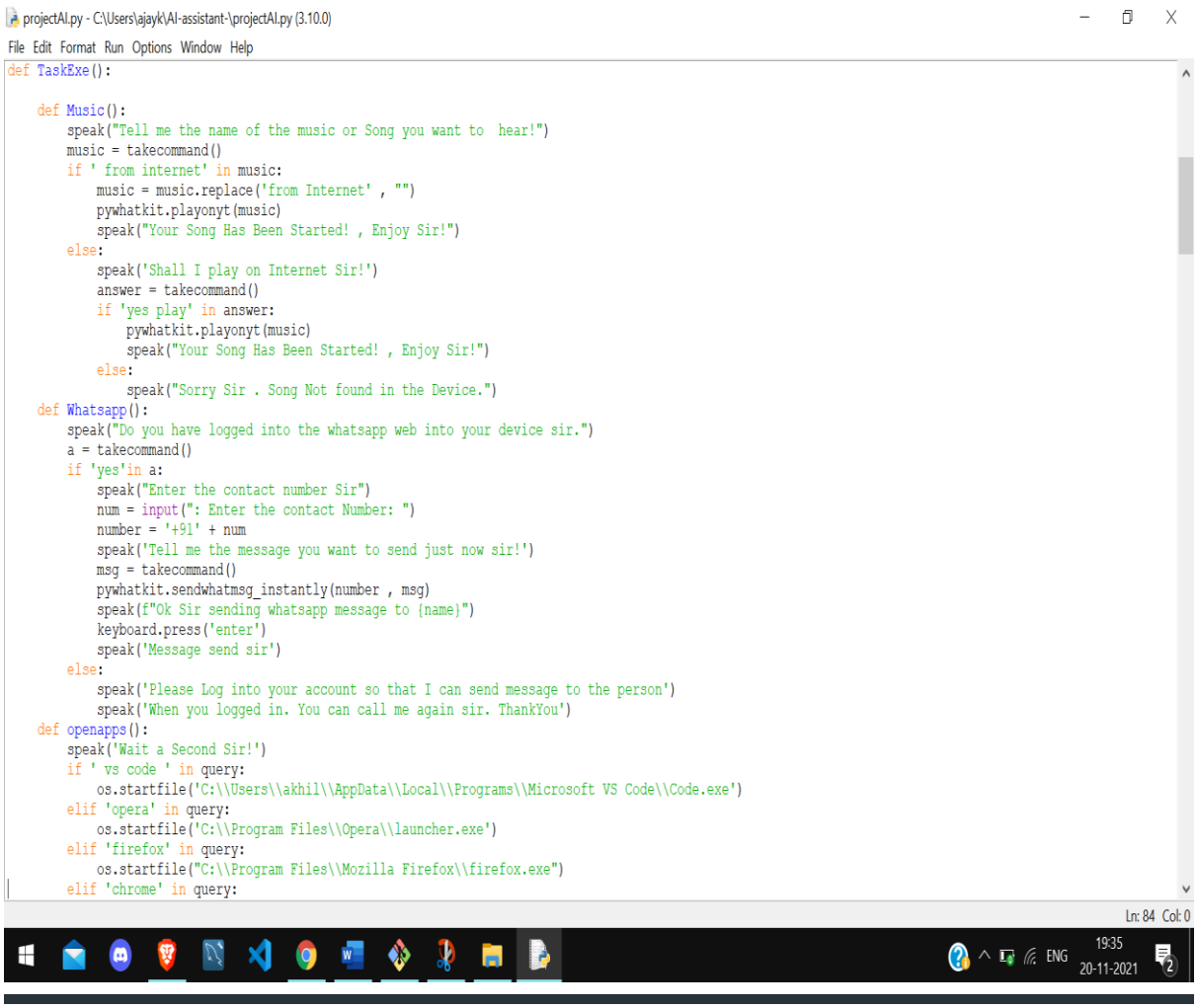


Figure 3.2 (TaskExe function)

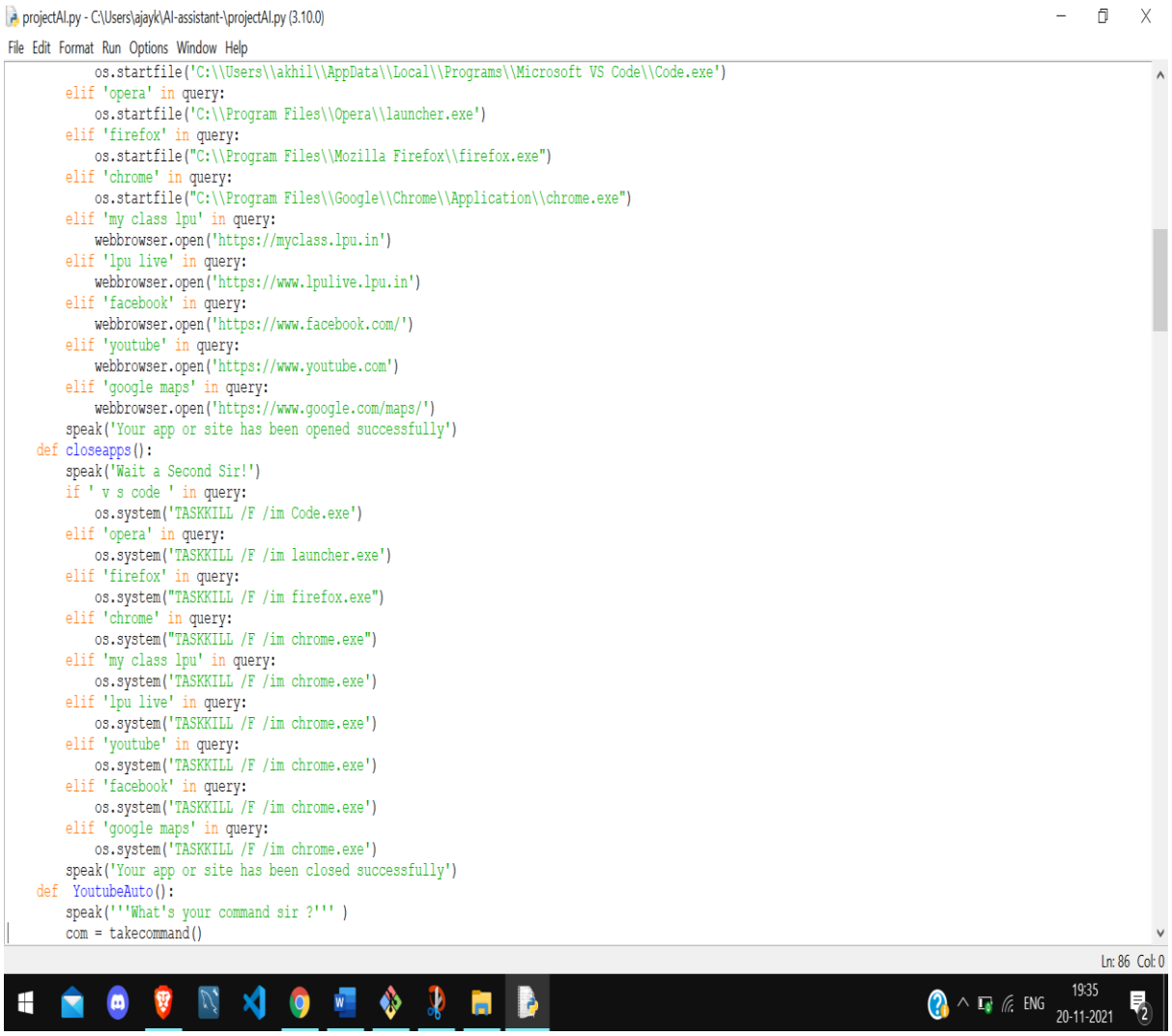


Figure 3.2 (TaskExe function)

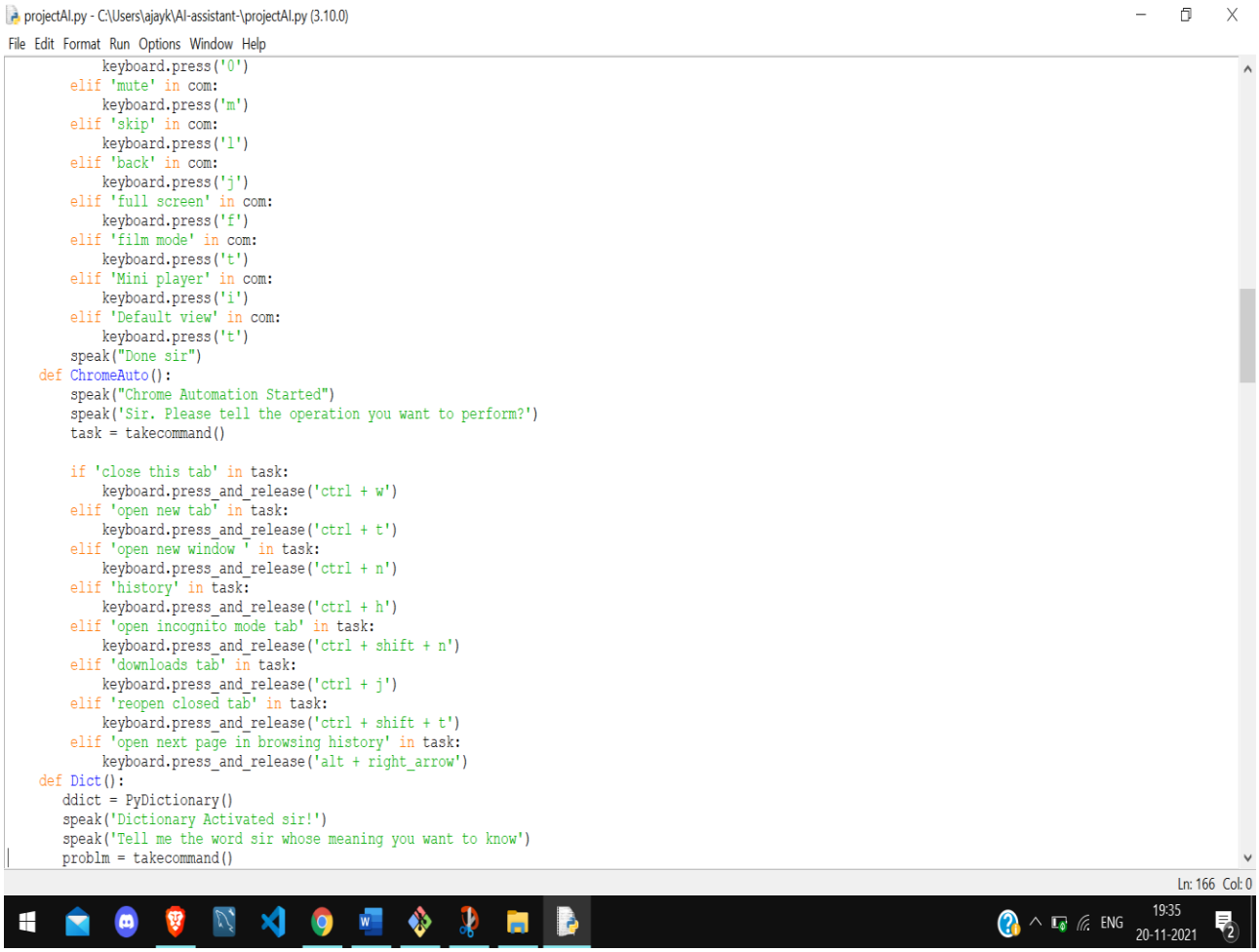


Figure 3.3 (TaskExe function)

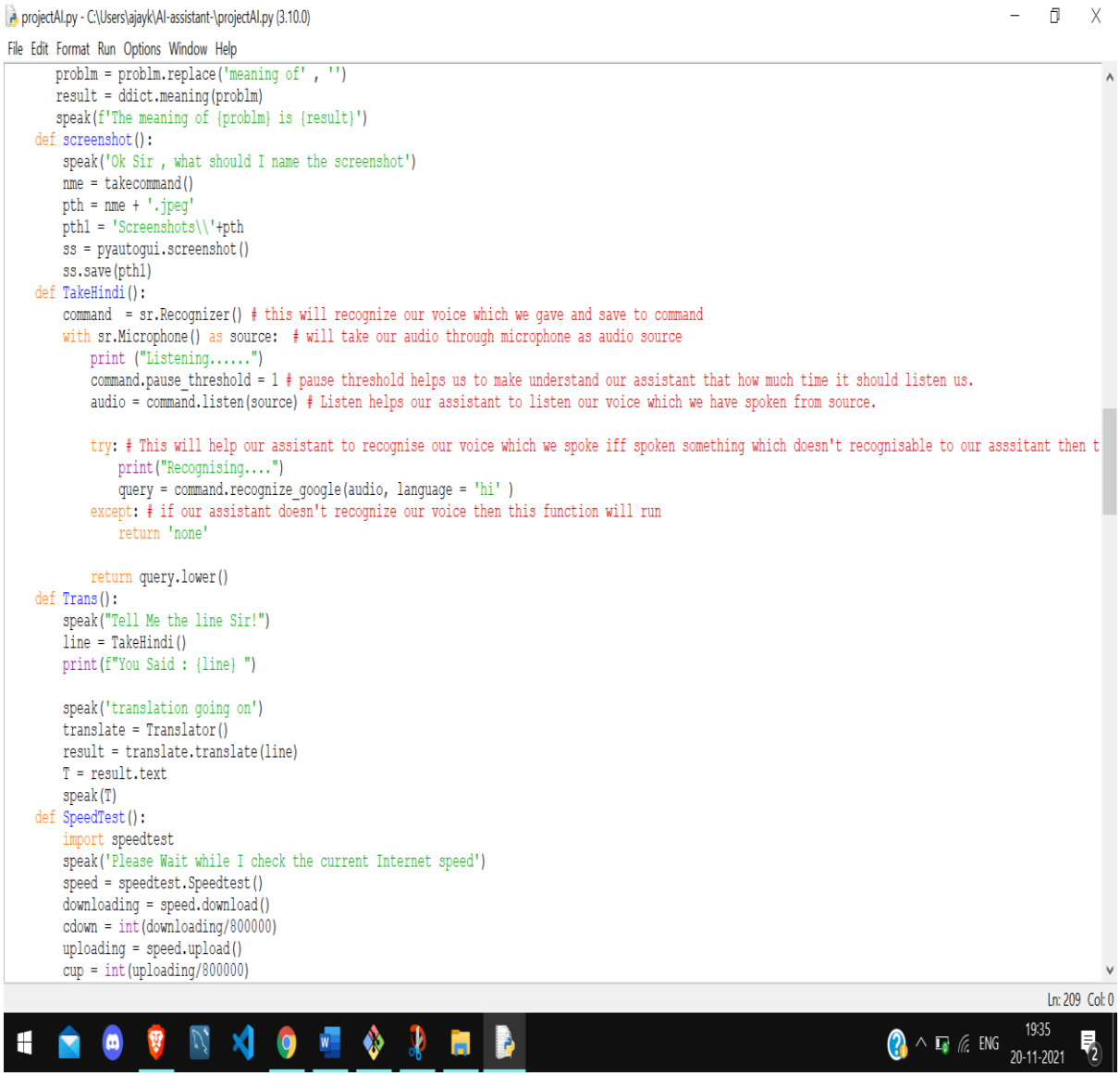


Figure 3.4 (TaskExe function)

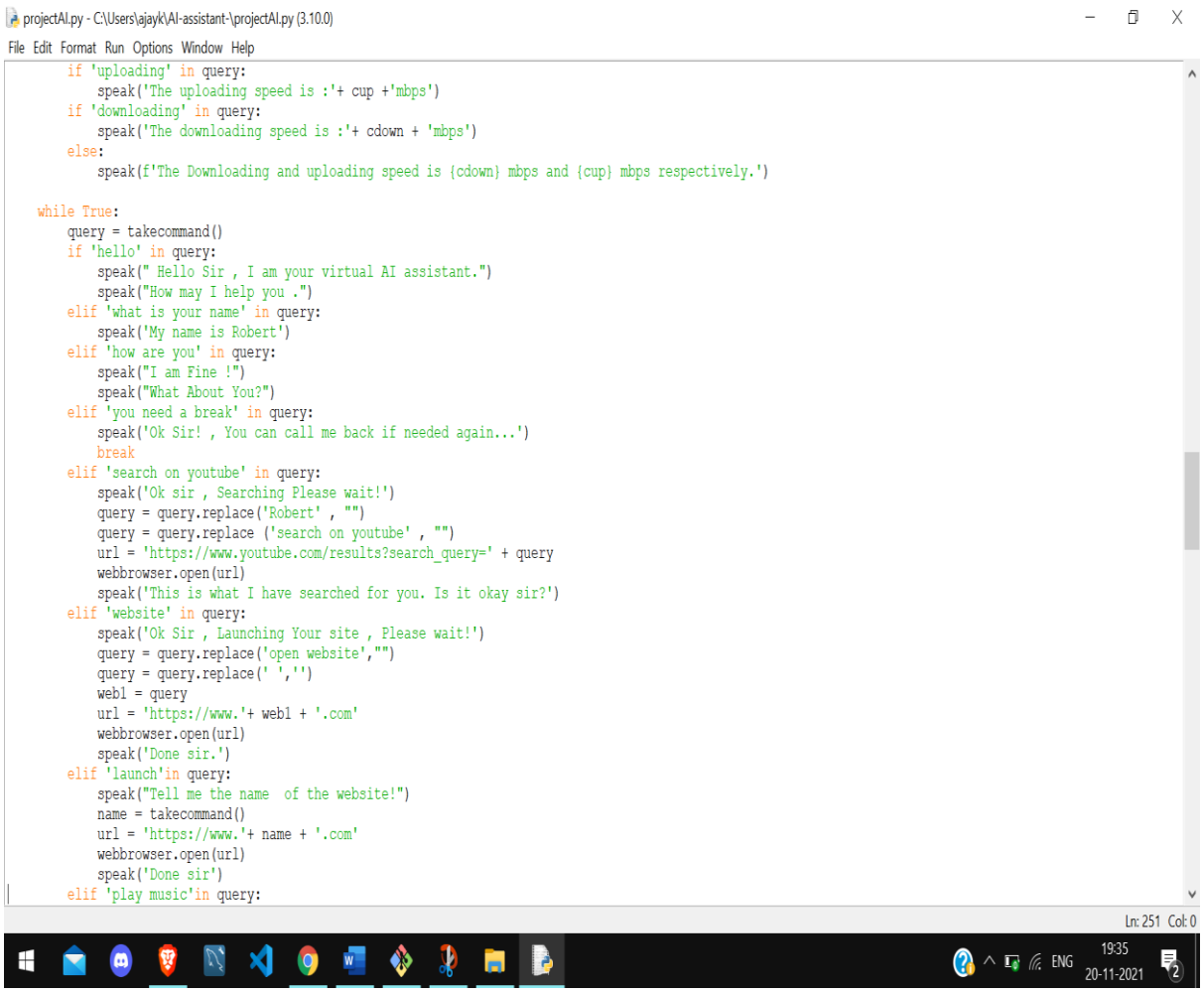


Figure 3.5 (TaskExe function)

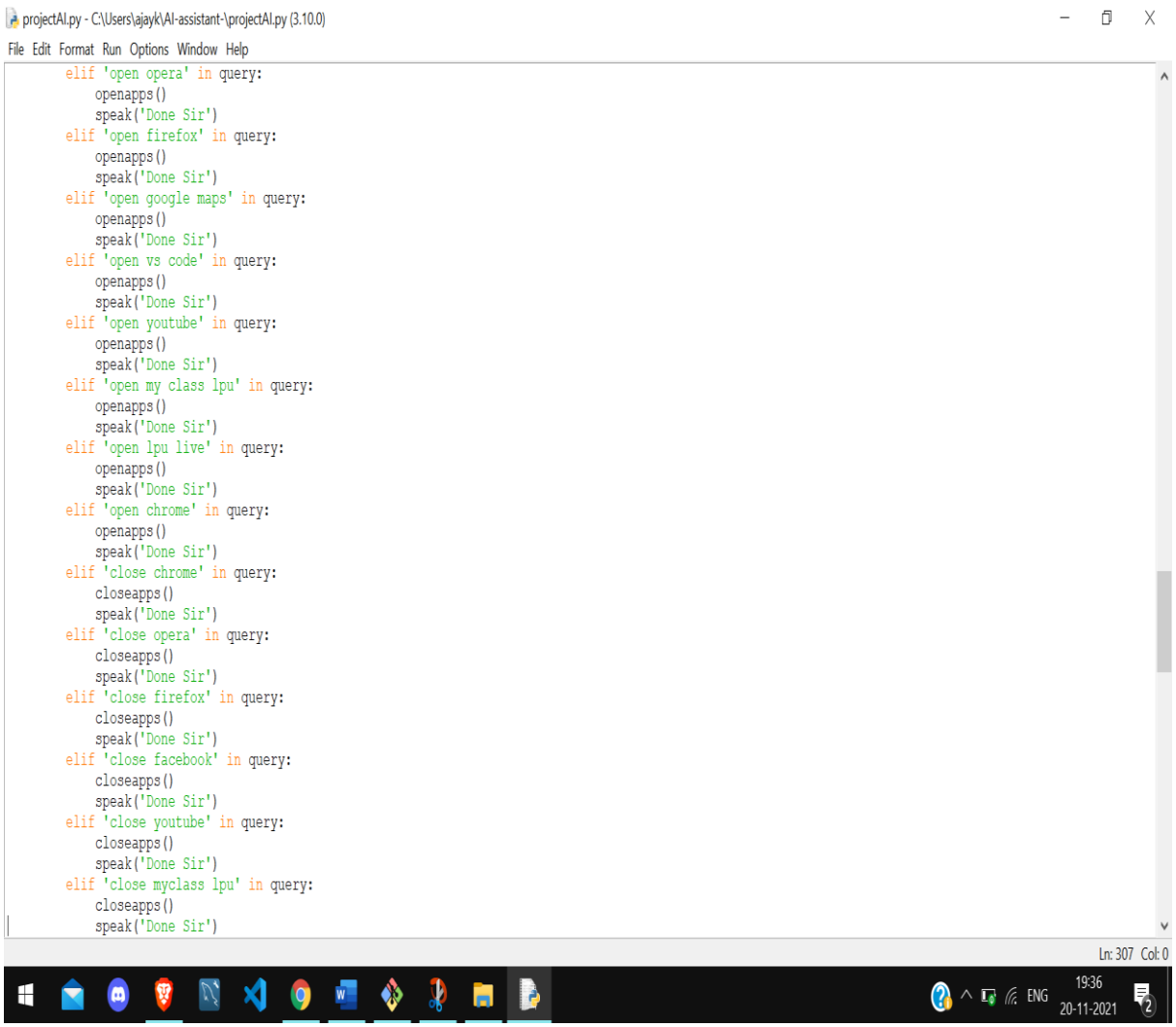


Figure 3.6 (TaskExe function)

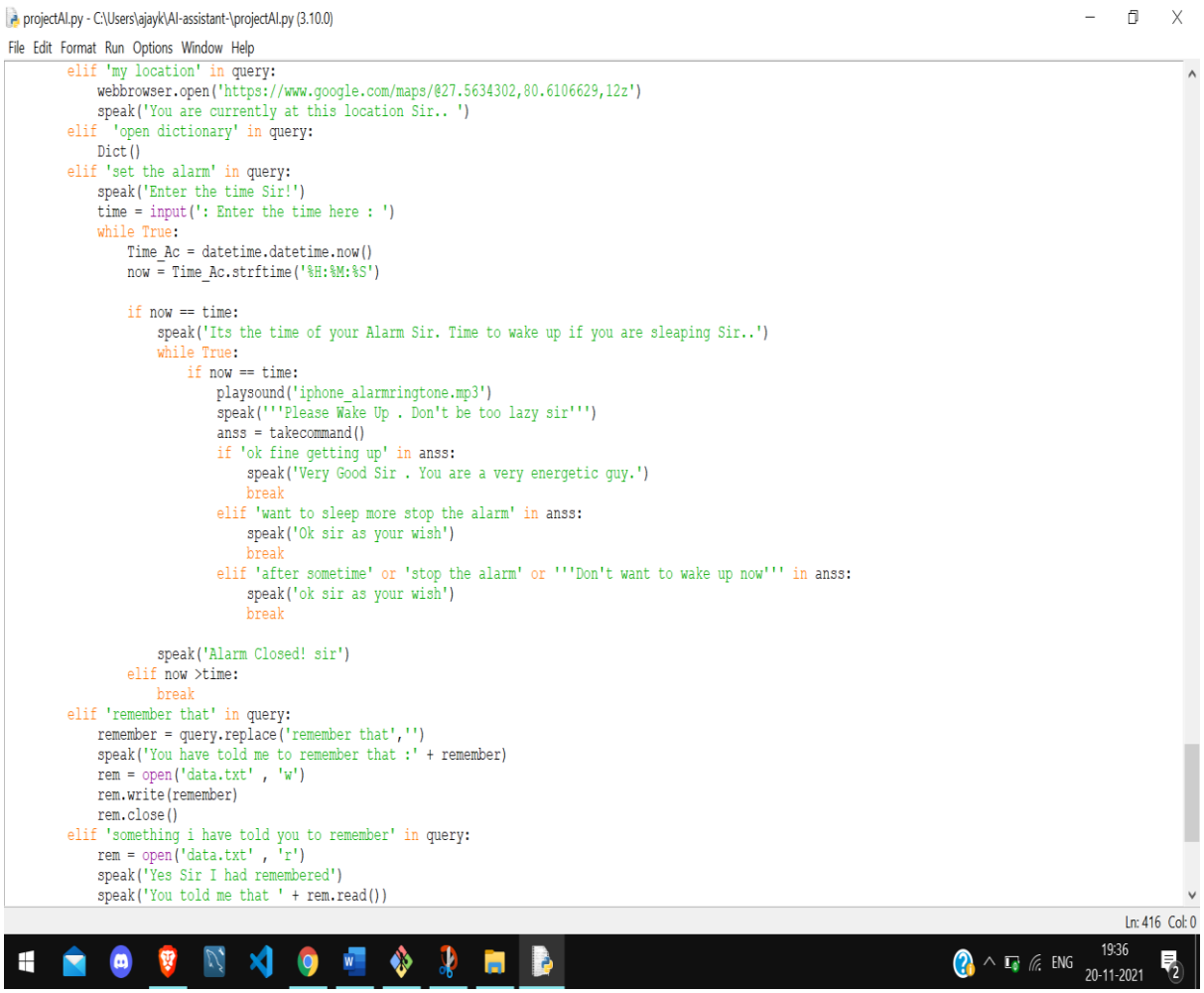
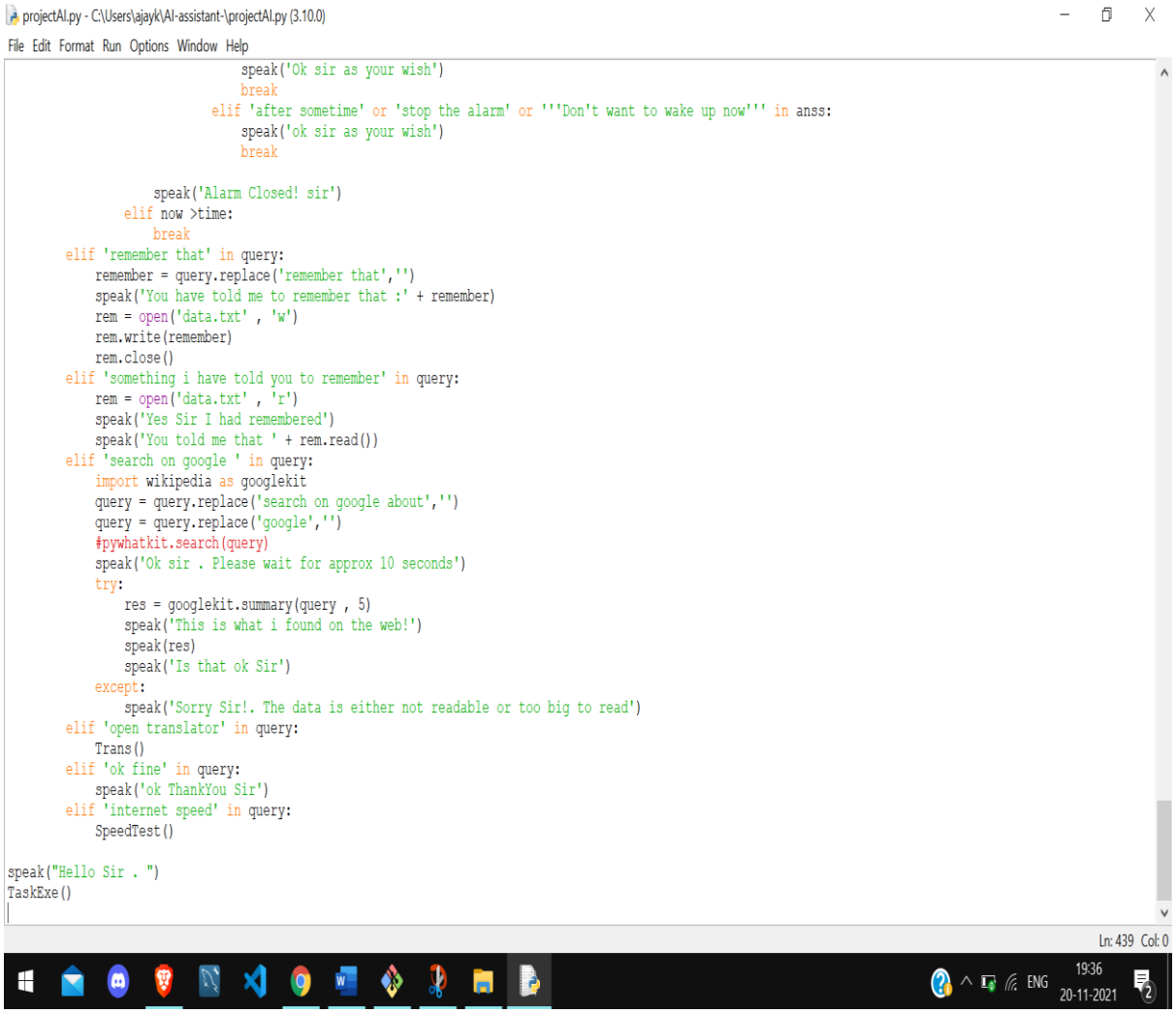
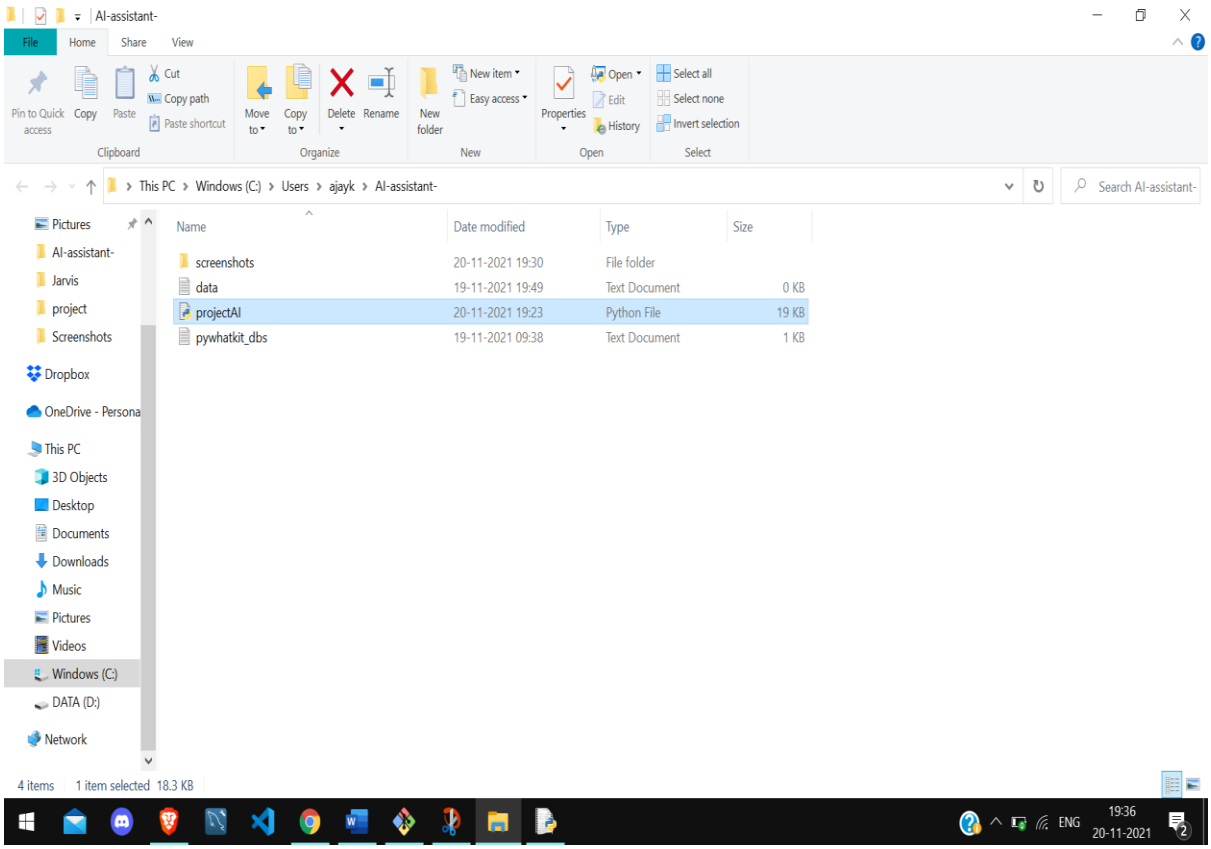


Figure 3.7 (TaskExe function)

Figure 3.8 (TaskExe function)



Repository AI assistant:



Screenshots by AI assistant:

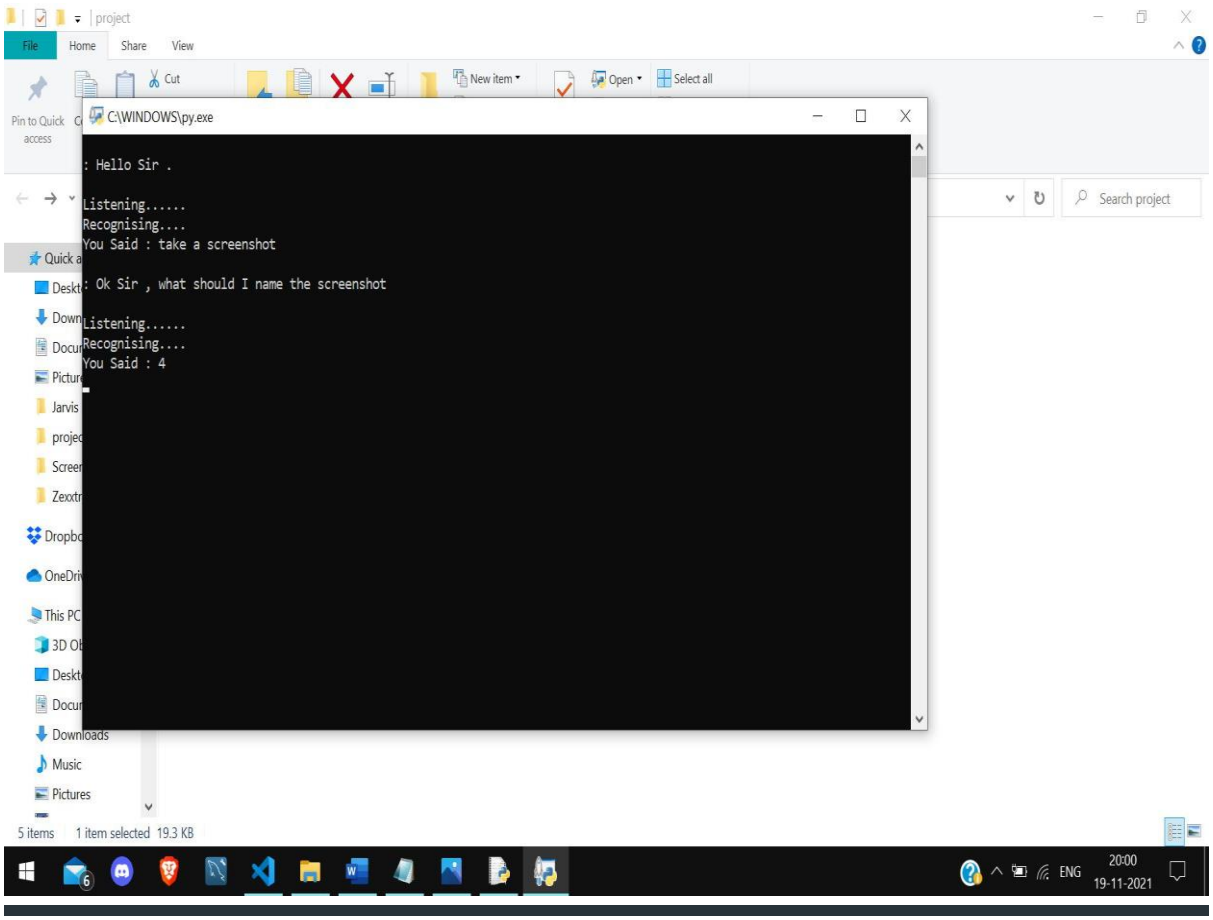


Figure 4 (screen shot command)

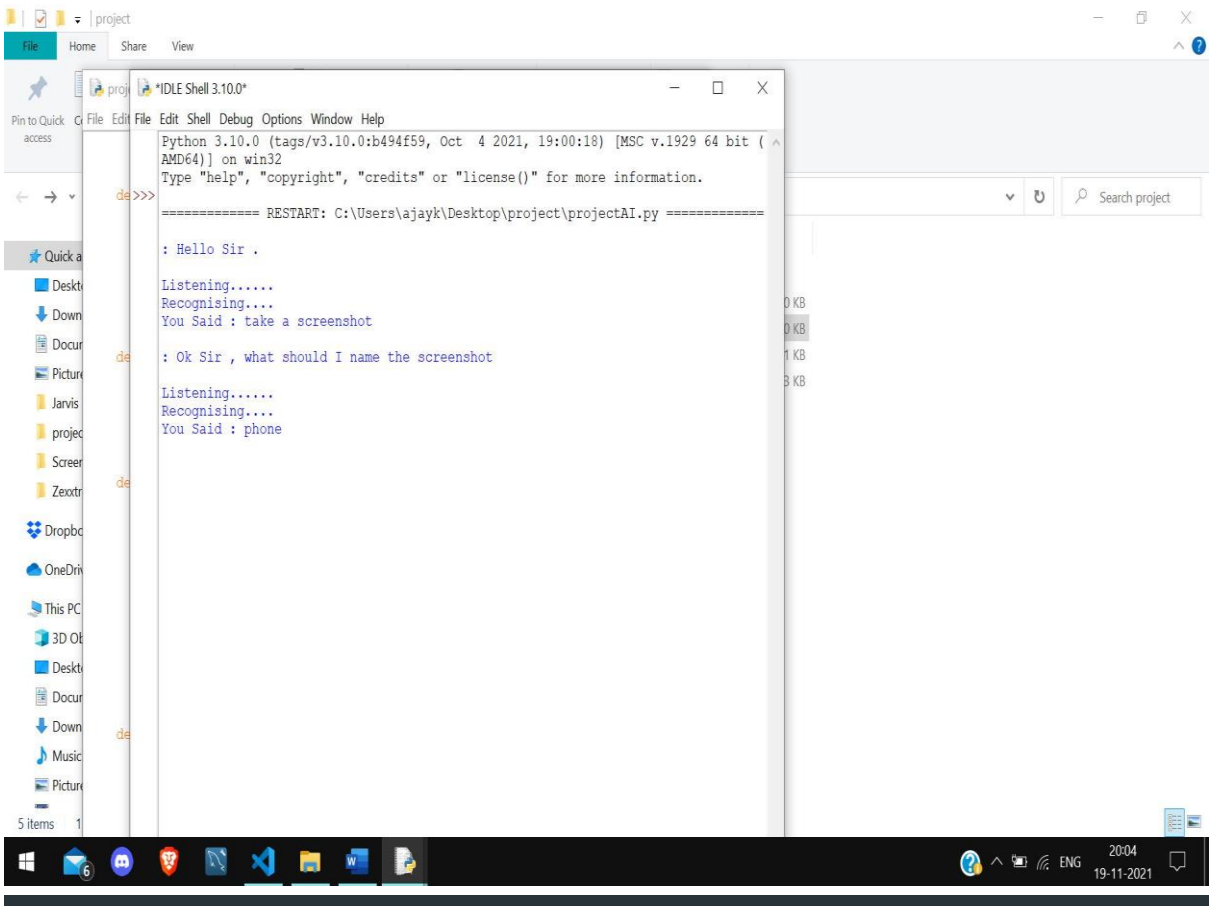


Figure 5 (screenshot command on shell)

References: -

To conduct this project following resources have been used: -

A. VS CODE: -
https://code.visualstudio.com/

B. Linked IN: -
https://www.linkedin.com/

C. Geek for geeks: -
   a. https://www.geeksforgeeks.org/keyboard-module-in-python/
   b. https://www.geeksforgeeks.org/python-text-to-speech-by-using-pyttsx3/
   c. https://www.geeksforgeeks.org/speech-recognition-in-python-using-google-speech-api/
   d. https://www.geeksforgeeks.org/introduction-to-pywhatkit-module/
   e. https://www.geeksforgeeks.org/wikipedia-module-in-python/

D. Miscellaneous docs: -
https://docs.python.org/3/library/os.html

E. Programiz: -
https://www.programiz.com/python-programming/time

F. Python Host: -
https://pythonhosted.org/pytz/

G. YouTube: -
https://www.youtube.com/

H. Stackoverflow: - https://stackoverflow.com/

CONCLUSION: -

It is our team's hope that this document will be of huge help with understanding of our little project as we have used a different approach which has proved beneficial for us and easy for us to understand the vast ocean that python has in case of APIs. We have reached the maximum accuracy which we could in order to make our program as much optimize, so as the user can have lag free experience. In future we will try to add some new stuffs like GUI, more features and try to make our assistant even more flexible and more optimised as there's always a hope of optimization.

Blank Page left intentionally for reader's Notes.