Project Topic: DESKTOP VOICE ASSISTANT USING PYTHON

ABSTRACT:

Ever thought of having a personal assistant that would do everything you would ask it to do? In this modern era of computers, a “Digital” personal assistant is always a much-preferred option, right? In this project a personal voice assistant **CRISTIANO** (name of voice assistant) is built, and it’s an intelligent virtual voice assistant that will allow you to interact with your desktop via voice commands.

INTRODUCTION:

A Voice Assistant user interface is an interface that allows us to interact with a desktop through the medium of voice commands. It can be called or referred as various names such as Intelligent personal assistant, voice assistant, smart assistants etc. The good thing about these voice assistants are that these allow hands-free user experience and usability. An Intelligent personal assistant or voice assistant can help the user with some basic tasks. These can recognize human voice commands and can perform tasks like greeting, searching Wikipedia, playing any media file etc. It uses python’s speech recognition

module to recognize the voice command and translate speech to text and vice versa. One most important part of speech recognition is the speech, this speech acts as an input to the system which further processes the command by converting the received speech to text and return appropriate output in textual form, which is later converted to speech. The evolution of voice assistants in the recent years has been quite indulging and how it made its way to the everyday life of a normal human being. This technology can be observed in smartphones, smart speakers, Infotainment systems, Healthcare, automobile industry etc. Some of the mainstream voice assistants are Siri, Google Assistant, Amazon Alexa and Google Home pad.

REQUIREMENTS:

- PYTHON 3.8
- PYTHON PACKAGES
- CODE EDITOR PLATFORM: VISUAL STUDIO CODE
- MICROPHONE

PYTHON PACKAGES & MODULES DESCRIPTION:

- pyttsx3
- speech recognition
- Datetime
- Wikipedia
- Webbrowser
- OS
- smtplib

PYTTSX3:

“pyttsx3” is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech. The pyttsx3 module supports two voices first is female and the second is male which is provided by “sapi5” for windows. In this project male voice is used using sapi5 which works on windows.

INSTALLATION:

Pip install pyttsx3

```
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)

#speak func will pronounce the string which is passed to it
def speak(text):
    engine.say(text)
    engine.runAndWait()
```

SPEECH RECOGNITION:

Speech Input Using a Microphone and Translation of Speech to Text. Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs. Configure Microphone (For external microphones): It is advisable to specify the microphone during the program to avoid any glitches.

Installation: pip install speech_recognition

```
35  #this fun will take commamnd from microphone
36  def  takeCommand():
37      r = sr.Recognizer()
38      with sr.Microphone() as source:
39          print("Listening..")
40          audio = r.listen(source)
41
42      try:
43          print("Recognizing..")
44          query = r.recognize_google(audio, language = 'en-in')
45          print(f"user said: {query}\n")
46
47      except Exception as e:
48          print("Say that again please")
49          speak("Say that again please")
50          query = None
51
```

DateTime Module:

It is a module in python used to work with date and time. In this project it is imported and used to get current time from

voice assistant Cristiano. Also, in wishMe function datetime module is used to greet the master while initializing. If it is morning it will greet Good Morning Master Name and so.

```
22  #this func will greet you based on current time
23  def wishMe():
24      hour = int(datetime.datetime.now().hour)
25
26      if hour>=0 and hour <12:
27          speak("Good Morning" + MASTER)
28      elif hour>=12 and hour<18:
29          speak("Good Afternoon" + MASTER)
30      else:
31          speak("Good Evening" + MASTER)
32
33      speak("I am Cristiano. How may I help You?")
```

Wikipedia:

Wikipedia is a Python library that makes it easy to access and parse data from Wikipedia. Search Wikipedia, get article summaries from a page. In this project Wikipedia module is used to get information from Wikipedia and read the results to master as per the user query.

```
69      #executing tasks as per the query
70      if 'wikipedia' in query.lower():
71          speak('Searching wikipedia..')
72          query = query.replace("wikipedia", "")
73          results = wikipedia.summary(query, sentences = 2)
74          print(results)
75          speak(results)
```

WEBBROWSER:

Web browser is a module used to open browser by simply calling the open() function from this **module** will open URL using the default **browser** . In this project it opens in chrome browser since chrome path is set, It opens google.com, youtube.com, gmail.com

```
#to open google
elif 'open google' in query.lower():
    url = "google.com"
    chrome_path = 'C:/Program Files (x86)/Google/Chrome/Application/chrome.exe %s'
    webbrowser.get(chrome_path).open(url)
#to open youtube
elif 'open youtube' in query.lower():
    #webbrowser.open("youtube.com") # opens in internet explorer
    url = "youtube.com"
    chrome_path = 'C:/Program Files (x86)/Google/Chrome/Application/chrome.exe %s'
    webbrowser.get(chrome_path).open(url)
#to open gmail in chrome
elif 'open gmail' in query.lower():
    url = "gmail.com"
    chrome_path = 'C:/Program Files (x86)/Google/Chrome/Application/chrome.exe %s'
    webbrowser.get(chrome_path).open(url)
```

OS:

The **OS module in python** provides functions for interacting with the operating system.

In this project it is used to play music by accessing the music folder in system.

```
#play music func
elif 'play music' in query.lower():
    songs_dir = "D:\\music"
    songs = os.listdir(songs_dir)
    print(songs)
    os.startfile(os.path.join(songs_dir, songs[1]))
```

LITERATURE SURVEY:

PAPER TOPIC: RASPBERRY PI BASED PERSONAL VOICE ASSISTANT USING PYTHON (2020)

- **AUTHOR**:P Srinivas, T Sai Teja, CH Bhavana, R Likhith, K Sathish Kumar (Asst. Professor)

- This paper is to demonstrate the implementation of a voice user interface (VUI) as a personal voice assistant that can perform several assignments or tasks for the user such as setting up the alarm, reminders for the day, knowing the weather forecast, reading news feed, to play a song from the playlist, ask about the details of a movie,

finding out the meaning of an unknown word, to read a article from Wikipedia, controlling electronic gadgets (such as Television, air-condition, blinders etc.) The personal voice assistant is very reliable and offers hands-free user experience to the user and is quite indulging how it has transformed in recent years. Our system resolves many drawbacks observed in current systems making it more reliable and scalable for future. Further many additional modules or packages can be introduced to expand the network and improving the user-end experience

PAPER TOPIC: ACOLYTE – A PERSONAL ASSISTANT (2020)

AUTHOR: Sri Harish,Aravind Sriram

▶ This project is to create a simple stand-alone application that helps less tech savvy people in the world to use the computer without feeling ignorant or computer illiterate. Computers have became a very important devices and as well as less expensive over time.The application works same like Siri/ Google Assistant etc. But the application deals with the computer itself mainly.Currently it takes

text as input as most of the people are not very good at speaking.

PAPER TOPIC: PERSONAL ASSISTANT WITH VOICE RECOGNITION INTELLIGENCE (2017)

- **AUTHOR**: Dr. Kshama V. Kulhalli , Dr.Kotrappa Sirbi , Mr. Abhijit J. Patankar .

- Personal Assistant with Voice Recognition Intelligence, which takes the user input in form of voice or text and process it and returns the output in various forms like action to be performed or the search result is dictated to the end user. In addition, this proposed system can change the way of interactions between end user and the PC. The system is being designed in such a way that all the services provided by the PC are accessible by the end user on the user's voice commands.

PAPER TOPIC: VOICE-BASED ASSISTANT USING PYTHON (2019)

- **AUTHOR:** Nagesh Singh Chauhan

- Voice assistants come in somewhat small packages and can perform a variety of actions after hearing your command. They can turn on lights, answer questions,

play music, place online orders and do all kinds of AI-based stuff. Voice assistants are not to be confused with virtual assistants, which are people who work remotely and can, therefore, handle all kinds of tasks. Rather, voice assistants are technology based. As voice assistants become more robust, their utility in both the personal and business realms will grow as well. The more a person interacts with voice-activated devices, the more trends, and patterns the system identifies based on the information it receives. Then, this data can be utilized to determine user preferences and tastes, which is a long-term selling point for making a home smarter. Google and Amazon are looking to integrate voice-enabled artificial intelligence capable of analyzing and responding to human emotion.

PAPER TOPIC: VOICE CONTROLLED PERSONAL ASSISTANT USING RASPBERRY (2019)

Authors: Ass. Prof. Emad S. Othman, Senior Member IEEE - Region 8, High Institute for Computers and Information Systems, AL-Shorouk Academy, Cairo – Egypt The project

involves the design and application of speech-to-text conversion and face recognition methods using Raspberry Pi.

This project can act as a prototype for many advanced applications. Usually projects belong to a single industry in terms of application, but my project has various aspects like entertainment, computation, face recognition and security. It can help the visually impaired to connect with the world by giving them access to Wikipedia, Calculator, Email and Music all through their voice. The project can also keep people secure as it can be used as a surveillance system which captures the face of the person standing at the door.

CODE:

```
import pyttsx3
import datetime
import speech_recognition as sr
import wikipedia import
webbrowser
import os
```

```

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
# print(voices[1].id)
```

```
engine.setProperty('voice', voices[1].id)

def speak(audio):
    engine.say(audio)
    engine.runAndWait()

def wishMe():
    hour = int(datetime.datetime.now().hour)
    if hour>=0 and hour<12:
        speak("Good morning Vaish")
    elif hour>=12 and hour<4:
        speak("Good afternoon Vaish")
    else:
        speak("Good Evening Vaish")

    speak("I am your Assistant Cristiano")

def takeCommand():
    # takes my command from microphone and gives text
    r =sr.Recognizer()
    with sr.Microphone() as
        source: print("listening...")
```

```
        r.pause_threshold = 1
        audio = r.listen(source)
    try:
        print("recognizing...")
        query = r.recognize_google(audio, language ='en-in')
        print("user said : ", query)
    except Exception as e:
        print(e)
        speak("Sorry Vaish, can you repeat that again?")
        return "None"
    return query

if __name__ == "__main__":
    wishMe()
    while True:
        speak("How can i help you?")
        query = takeCommand().lower()
        if 'wikipedia' in query:
                speak("searching in wikipedia")
                query = query.replace("wikipedia", "")
                results = wikipedia.summary(query, sentences = 2)
                speak("According to wikipedia")
```

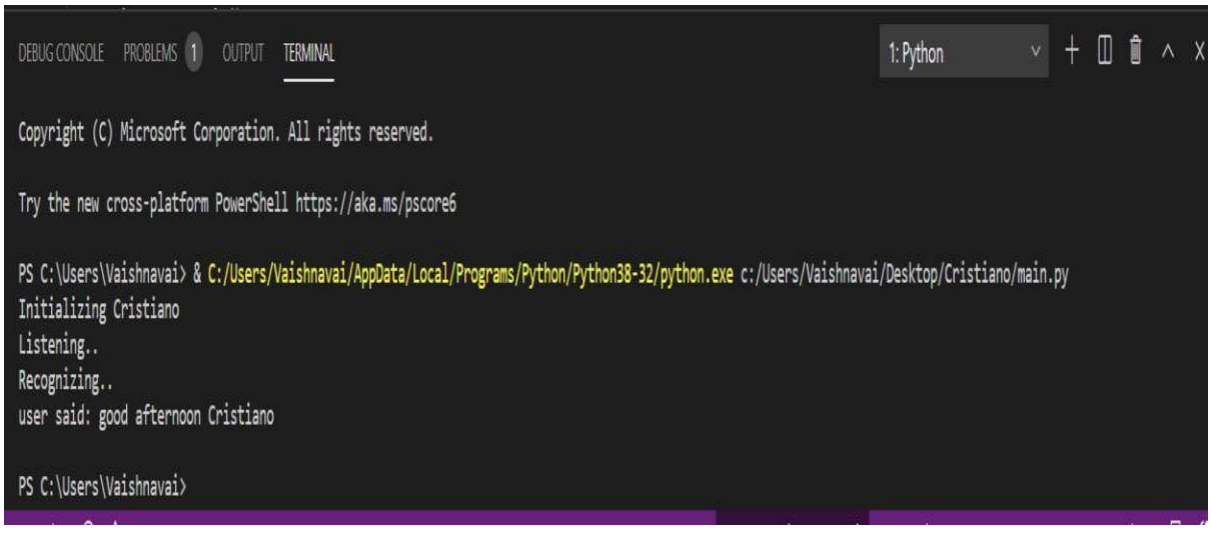
```
            print(results)
            speak(results)

        elif 'open youtube' in query:
            webbrowser.open("youtube.com")
            speak("youtube is opened")
        elif 'open google' in query:
            webbrowser.open("google.com")
            speak("google is opened")
        elif 'open gmail' in query:
            webbrowser.open("gmail.com")
            speak("gmail is opened")
        elif 'play music' in query:
            music_dir = 'D:\\music'
            songs = os.listdir(music_dir)
            print(songs)
            os.startfile(os.path.join(music_dir, songs[0]))
            speak("music is being played")
        elif 'time' in query:
            strTime = datetime.datetime.now().strftime("%H:%M:
%S")
            speak(f"the time is {strTime}")
```

```
        elif 'open code' in query:
            codepath = "C:\\Users\\Vaish\\AppData\\Local\\Programs\\Microsoft VS Code\\Code.exe"
            os.startfile(codepath)
        elif 'stop' in query:
            speak("see you soon vaish")
            exit()
        else :
            webbrowser.open(query)
```

IMPLEMENTATION:

- GREETING:

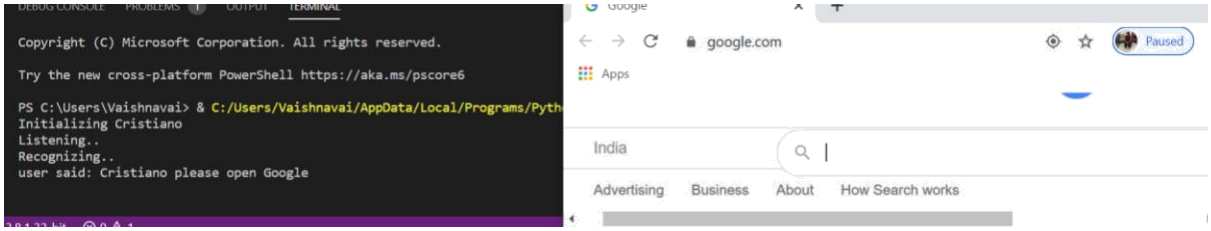


ASK CURRENT TIME:

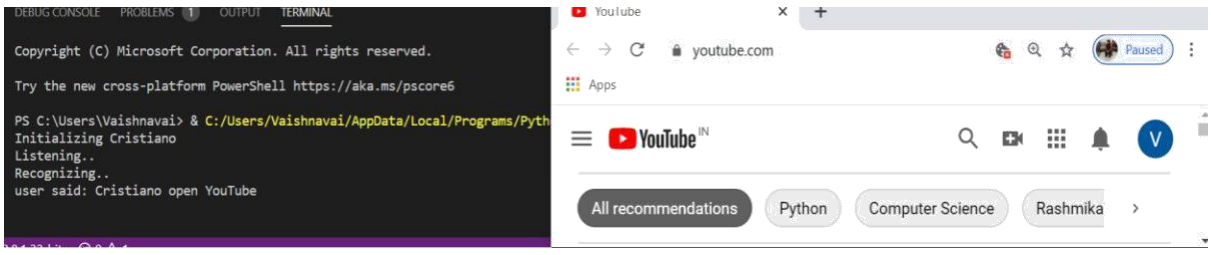
```
PS C:\Users\Vaishnavai> & C:/Users/Vaishnavai/AppData/Local/Programs/Python/Python38-32/python.exe c:/Users/Vaishnavai/Desktop/Cristiano/main.py
Initializing Cristiano
Listening..
Recognizing..
user said: Cristiano what is the time now


Vaishu the time is 14:38:40
PS C:\Users\Vaishnavai>
```

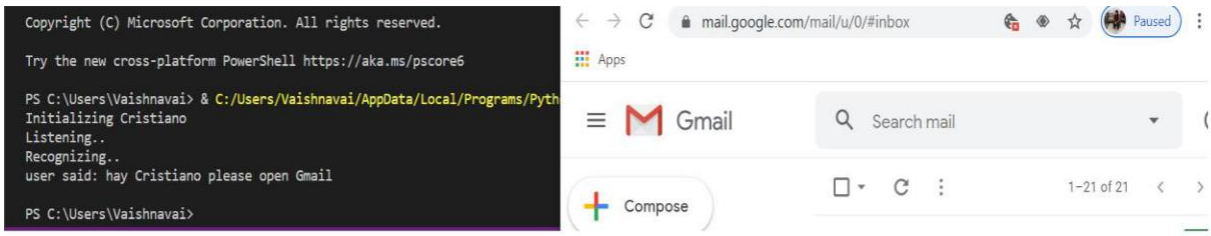
To Open Google



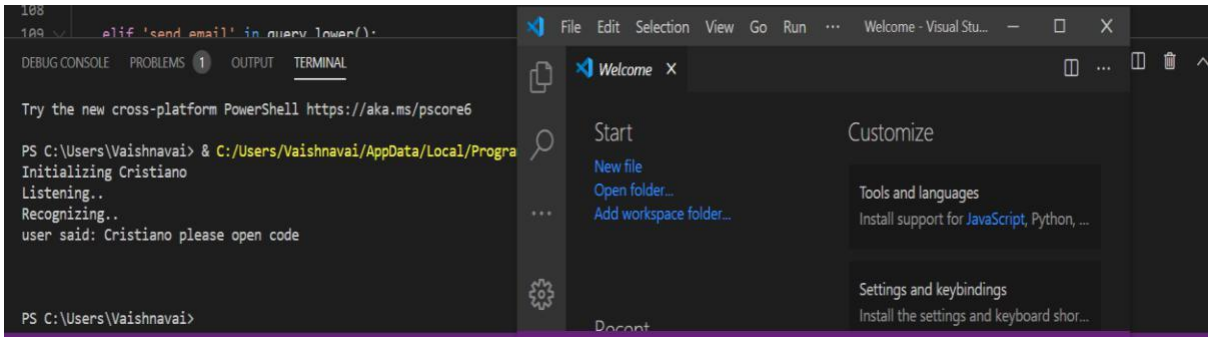
To open youtube



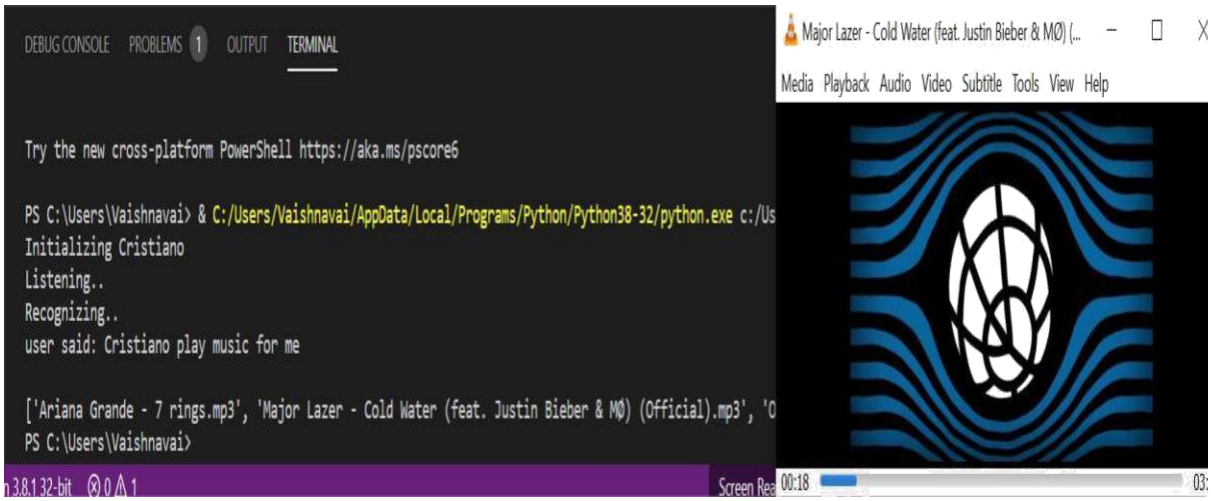
To open gmail



To open code editor



To play Music:



CONCLUSION:

Using pyttsx instead of gTTS helps to work in offline mode. This voice assistant can be customized based on

the user needs because we are using python it is more flexible and make easy to use. Further we can extend this project by using python libraries like smtplib to send email, send WhatsApp messages like other artificial voice assistants like Siri, ok google. Voice assistant can be trained accordingly to make best use of it.