INTRODUCTION

A voice-based virtual assistant is an AI-powered application that can interact with users through spoken commands and responses. Virtual assistant is used to run machine like laptop or PC’s on my own command. Virtual assistant is an application program that understands natural language and voice commands to complete tasks for the users. Virtual assistant is used to perform a typical task like showing datetime, managing emails, open apps, etc. on your command. Now day’s virtual assistant is very useful to human. It makes human life easier like operate PC’s or laptop on only voice command. Virtual assistant is a less time consuming. By using virtual assistant saves our time and contribute in other works. Virtual assistants are typically cloud-based program that requires internet connected devices. Virtual assistant is the flexibility to contract for just the services they need. For creating virtual assistant for your computer go from basics python. Virtual assistants are task-oriented. Virtual assistant’s ability to understand and perform requests. Virtual assistants is a software that understands verbal and written commands and completes task assigned by clients. Virtual assistants are able to interpret human speech and respond via synthesized voices. There are several voice assistants in market like Siri from Apple, Google Assistant from Google, Alexa as a smart speaker from Amazon which is developed by using Raspberry Pi, Microsoft Cortona. . In this project report, Explore the design, development, and implementation of a voice-based virtual assistant using Python and discuss the various tools and techniques used in building the virtual assistant, including speech recognition, natural language processing, and text-to-speech synthesis. They can respond to voice or text commands (such as those from online Chabot’s). An inciting phrase or wake word is required by voice-based intelligent assistants before the instruction to activate the listener. This system is made to operate effectively on desktop computers. By taking care of the user's repetitive activities and supplying information from an online source, personal assistant software increases user productivity.

BACKGROUND

Virtual assistants have come a long way since their inception in the 1960s. Initially, they were basic computer programs that were designed to respond to a limited set of voice commands. However, with the advent of artificial intelligence (AI) and natural language processing (NLP) technologies, virtual assistants have evolved to become much more sophisticated and capable of handling complex tasks. The proliferation of smartphones and smart home devices has led to a surge in the popularity of virtual assistants, with companies such as Amazon, Apple, Google, and Microsoft all offering their own voice-based virtual assistants. These virtual assistants have become an integral part of our daily lives, allowing us to set reminders, play music, control smart home devices, and even order groceries with a simple voice command. In recent years, there has been a growing interest in developing custom voice-based virtual assistants that can be personalized to specific use cases and industries. This has led to an increase in research and development in the field of AI-powered voice assistants, with many companies and organizations investing heavily in this technology. The development of virtual assistants has not been without challenges, however. One of the biggest challenges has been ensuring that virtual assistants can accurately understand and interpret voice commands. This has required the development of advanced speech recognition and natural language processing technologies. Despite these challenges, the potential benefits of virtual assistants are vast, with applications in healthcare, education, finance, and many other industries. The development of custom voice-based virtual assistants is an exciting area of research and development, and has the potential to revolutionize the way we interact with technology in the future.

Figure 1: Virtual Assistants

PURPOSE, SCOPE AND APPLICABILITY

PURPOSE

Purpose of virtual assistant is to being capable of voice interaction, music playback, making to do lists, Setting alarms, Streaming podcasts, Playing audiobooks, and providing Weather, Traffic, Sports, and other Real time information, such as news. Virtual assistant sense able users to speak natural language voice commands in order to operate the device and it's apps. There is an increased overall awareness and a higher level of comfort demonstrated specifically by millennial consumers. In this ever evolving digital world where speed, efficiency, and convenience constantly being optimized, it's clear that we are moving towards less screen interaction.

SCOPE

Voice assistants will continue to offer more individualized experiences as they get better at differentiating between voices. However, it's not just developers that need to address the complexity of developing for voice as brands also need to understand the capabilities of each device and integration and if it makes sense for them specific brand. They will also need to focus on maintaining a user experience that is consistent within the coming years as complexity becomes more of a concern. This is because the visual interface with voice assistants is missing. Users simply cannot see or touch a voice interface.

The scope of a voice-based virtual assistant for Windows can be quite extensive. Here are a few potential areas where a virtual assistant could be helpful:

System and app control: A virtual assistant could help users navigate their computer and control apps using voice commands, such as opening or closing apps, launching websites, or adjusting system settings.

Information retrieval: Users could ask the virtual assistant for information, such as the weather, news headlines, or stock prices, and receive a spoken response.

Productivity: The virtual assistant could help users manage their to-do lists, set reminders, and create calendar events, all through voice commands.

Entertainment: Users could use the virtual assistant to play music, movies, or TV shows, or to control their smart home devices.

Customer service: Companies could use virtual assistants to help customers navigate their websites or products, or to answer common questions and provide support.

Overall, the scope of a voice-based virtual assistant for Windows is quite broad and could be customized to meet the needs of individual users or businesses.

APPLICABILITY

The applicability of a voice-based virtual assistant for Windows is extensive and can be useful in various areas. Here are a few potential areas where a virtual assistant could be applied:

Accessibility: A virtual assistant can assist users with disabilities or those who have difficulty using a mouse and keyboard to operate their computers.

Efficiency: By using voice commands, users can save time and effort when completing tasks on their computers, such as navigating to different files and folders, opening and closing applications, and even performing complex operations.

Productivity: Users can utilize virtual assistants to manage their daily tasks such as setting up reminders, creating to-do lists, and scheduling meetings.

Automation: Virtual assistants can be used to automate repetitive tasks such as sending emails, replying to messages, or creating reports.

Home automation: Users can integrate their virtual assistant with their smart home devices to control lighting, thermostats, and other appliances through voice commands.

Customer service: Companies can use virtual assistants to handle customer queries, provide support, and even assist customers in purchasing products.

Overall, the applicability of a voice-based virtual assistant for Windows is vast, and it can be customized to meet the needs of individual users or businesses. It can improve efficiency, productivity, and accessibility while also making computing more convenient and enjoyable for users.

OBJECTIVES

The primary goal of creating virtual assistant software is to employ user-generated material, semantic data sources available on the web, and knowledge databases to provide expertise. An intelligent virtual assistant's primary function is to provide users with information. This can be done in a professional setting, for instance, using a chat interface on the company website. Virtual helpers can help companies save a tonne of time. Usually spend hours conducting online research before creating the report using our own language. The speed of voice searches is one of their key benefits. In fact, it's said that voice searches are four times faster than written ones: whereas humans can write roughly 40 words per minute. For example, on the business website, with a chat interface. On the mobile platform, the intelligent virtual assistant is available as a call-button operated service where a voice asks the user "What can I do for you?" and then responds to verbal input. Currently, the project aims to provide the Windows Users with a Virtual Assistant that would not only aid in their daily routine tasks like searching the web, playing music and many others but also help in automation of various activities. In the long run , aim to develop a complete server assistant, by automating the entire server management process deployment, backups, auto-scaling, logging, monitoring and make it smart enough to act as a replacement for a general server administrator.

Virtual assistants are designed to provide speech control of computers**.** The virtual assistant may open YouTube and webpage, run Google, play, open a file, and numerous other commands.

The advantages of a virtual assistant include time savings.

• The ability to use your computer hands-free.

• It is easy to operate.

LITERATURE REVIEW

In the existing system of virtual assistant there are several virtual assistants in market by using Artificial Intelligence technology. Many companies have used the dialogue systems technology to establish various kinds of Virtual Personal Assistants (VPAs) based on their applications and areas, such as Microsoft's Cortona for Windows and Espeak for Linux, Siri for Apple, Google Assistants For Android.[1] The first digital virtual assistant installed on a smartphone of apple was Siri, It was introduced as a feature of the iPhone in 2011. Aim of that virtual assistant was to add in tasks such as sending a text message, making phone calls, checking the weather or setting up an alarm. Over time, it has developed to provide restaurant locations, search the internet, and provide driving directions. In 2014 Cortona virtual assistants was developed by Microsoft. Cortona uses Bing search engine for performing tasks like answering questions for the users, setting remainder, etc. In 2016 Google Assistant was developed by google. It is primarily available for mobiles and smart home devices. Google Assistants via chat on google messaging app and via voice on google smart home speaker.[3] Cortona was developed by Microsoft as personal virtual assistant for windows, iOS, android, etc. In windows operating system Cortona works only for windows 10. It was released for windows 10 in 2015. In windows 10 Cortona is in Icon form on taskbar next to the search bar for use the Cortona application we try to setup for activate the Cortona in our laptop or PC's. It is easy to search but it takes more time to setup. It is very time consuming. It works in windows only for windows 10. It is not helpful for other windows version or explorer like windows 7, 8, etc. Therefore for other versions of windows we try to make the personal virtual assistant which is able to access on any windows explorer such as windows 7,8,10. In this project we use Python as a programming language and Visual Studio Code as a platform on which we execute our code for virtual assistant.

SYSTEM CONFIGURATION

System configuration mainly refers to the specification of a given computer system, from its hardware components to the software and various processes that are run within that system. It refers to what types and models of devices are installed and what specific software is being used to run the various parts of the computer system.

- Windows: 10 or newer
- MAC: OS X v10.7 or higher

SOFTWARE SPECIFICATION

Operating system	Windows 10
Coding language	Python language
Software tool	Visual Studio, MS office

HARDWARE SPECIFICATION

We strongly recommend a computer fewer than 5 years old.

- Processor: Minimum 1 GHz; Recommended 2GHz or more
- Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)
- Hard Drive: Minimum 32 GB; Recommended 64 GB or more
- Memory (RAM): Minimum 1 GB; Recommended 4 GB or above
- Sound card w/speakers
- Some classes require a camera and microphone

Processor	Intel(R) Celeron(R) CPU J3060 @ 1.60GHz 1.60GHz
Hard disk	1TB
RAM	4GB

METHODOLOGY

The Speech Recognition library has many built-in features that will enable the assistant to understand the command given by the user and the response will be sent back to the user in voice, using Text to Speech functions, in the proposed concept for an efficient way to implement a personal voice assistant. The underlying algorithms will translate the voice into text when the assistant records a user's voice instruction. And the assistant will carry out the appropriate action in accordance with the keywords present in the text (based on the user's demand). The functions included in various libraries enable this. Additionally, the assistant used some APIs to complete all of the capabilities. These APIs were utilised by us for a variety of tasks, including computations and the extraction of news from online sources. It will make a request, and the API will then provide the appropriate response. APIs like WOLFRAMALPHA are highly useful for carrying out computations and conducting quick online searches. Also, not every API will be able to translate the raw JSON data into text when obtaining data from the web. In order to parse the JSON Data arriving from websites into string format, designers used a library named JSON. This allows us to gather news from online sources and pass it as input to another function for later use. Furthermore, there are a variety of libraries available, each corresponding to a distinct technology, including libraries like Random with the help of the library OS, designers implemented functionalities related to operating systems, such as a system's shutdown, PDF reader, etc.

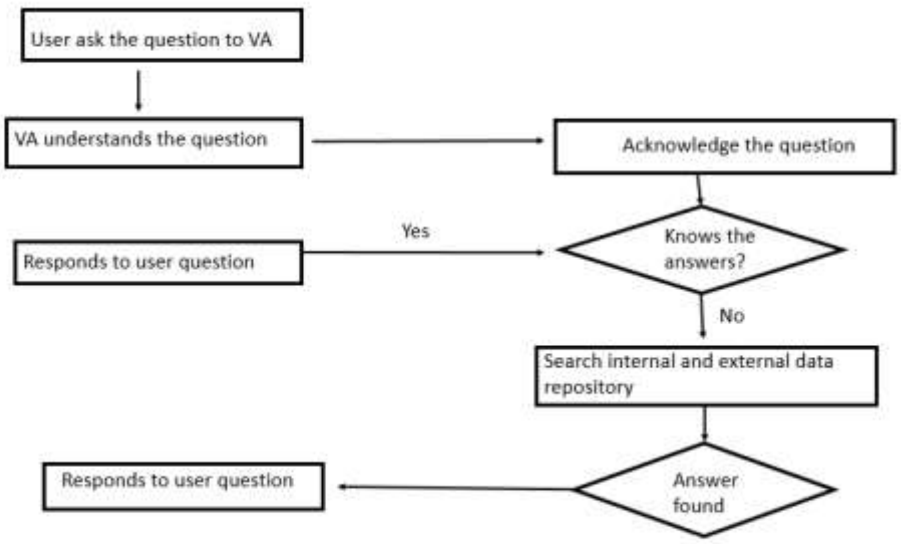

User ask the question to VA
VA understands the question
Acknowledge the question
Yes
Responds to user question
Knows the answers?
No
Search internal and external data repository
Responds to user question
Answer found

Figure2: Working Methodology

TECHNOLOGIES USED

Python was decided as the programming language for this project because of its adaptability and accessibility to a large number of libraries. Designers used Python programming language-supporting Microsoft Visual Studio Code (IDE) to create the Virtual Assistant. Python has a speech recognition package that includes certain built-in functions. Designers will first define a function that will turn the text into speech. Designers utilize the pyttsx3 library for that. The library instance will be set to a variable. Designers utilize the say() method and supply the text as an argument; the result is a vocal response. Another function has been defined to identify the user's voice command. Designers provide the microscope source in that function, use the appropriate functions within its bounds, and save the output in a variable. Designers are able to utilize a variety of services for the entire process, including the Microsoft Bing and Google Speech Recognition engines as well as goods made by other major corporations like IBM, Houndify, etc. Designers select Google's Voice Recognition Engine for this project, which will translate the relevant analogue voice command into a digital text format. The Assistant will look for the keyword after receiving that text as input. even respond to inquiries from people, speaking in various voices. There is no requirement that one must issue a command that is precisely described in order to initiate a specific operation. User is free to issue commands in natural language for user. This voice-activated personal assistant for the PC was created using the Python 3.8.3 programming language. Microsoft Visual Code served as the IDE (Integrated Development Environment).

SYSTEM DESIGN

There are three modules in this Assistant. The first is that the assistant will accept user voice input. Second, analysing the user's input and matching it to the appropriate intent and function. The third option is the assistance speaking the user's outcome throughout. The assistant will initially begin to collect user input. The assistant will translate the analogue voice input into digital text after receiving it. The assistant will start asking the user for input again if it was unable to translate the voice into text. If converted, it will begin to analyse the input and map it to a certain function. Thereafter, the user will receive the output by voice command.

Imported Modules

Speech recognition

To translate vocal input into text, the system makes use of Google's online speech recognition system. With the use of this, users can talk into a microphone to input text in exchange for voice input from a special corpus that is gathered on a computer network server at the information centre. The text is then delivered to Google Cloud for speech recognition after being temporarily kept on-site. After that, the voice assistant programme receives the same text and sends it.

Datetime

The Datetime package is utilised to display the Date and Time. Python has a built-in datetime module.

Wikipedia

Wikipedia is a fantastic and extensive source of knowledge, as almost all know. In order to search Wikipedia or to obtain further information, designers have used the Wikipedia module in the project. Use pip install Wikipedia to install this Wikipedia module.

Web browser

To perform Web Search. This module comes built-in with Python.

OS

The OS module in Python offers tools for interacting with the operating system. The standard utility modules for Python include OS. This module offers a method for using operating system-dependent features.

Pyjokes

Pyjokes is a tool for collecting jokes online. Pyjokes is included in our project since it includes jokes. It's also quite entertaining. Pyjokes is the one-line joke that adds interest to our project.

Requests

Python's Requests module enables you to send HTTP requests. It is used to send GET and POST requests. It hides the challenges of submitting requests behind a tidy, accessible API.

Sub process

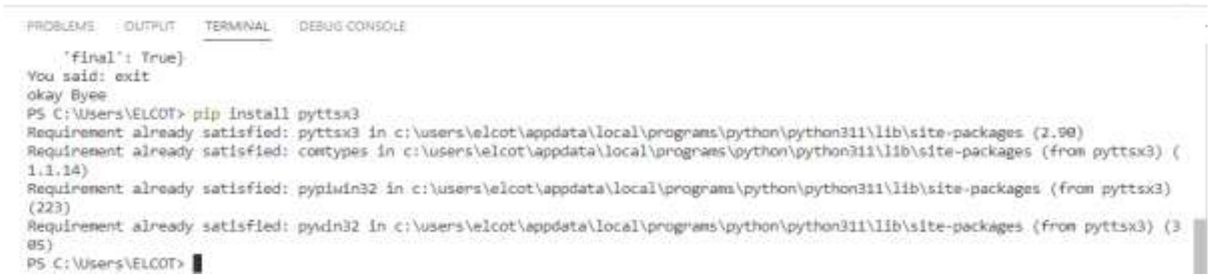
This module is used to retrieve information about system sub processes that are involved in commands like shutdown and sleep, etc. This module is pre-installed with Python.

Pyttsx3

Python's *pyttsx3* module converts text to speech. It works offline and is suitable with Python 2 and Python 3 unlike comparable libraries.

JSON

JavaScript Object Notation, or JSON, is a coding language. Data storage and delivery can be done using the lightweight JSON format. Data transmitted from a server to a web page is frequently sent in JSON format. JSON is "self-descriptive" and simple to comprehend. Recognizing speech



```
    'final': True}
You said: exit
okay Byee
PS C:\Users\ELCOT> pip install pyttsx3
Requirement already satisfied: pyttsx3 in c:\users\elcot\appdata\local\programs\python\python311\lib\site-packages (2.90)
Requirement already satisfied: comtypes in c:\users\elcot\appdata\local\programs\python\python311\lib\site-packages (from pyttsx3) (
1.1.14)
Requirement already satisfied: pypiwin32 in c:\users\elcot\appdata\local\programs\python\python311\lib\site-packages (from pyttsx3)
(223)
Requirement already satisfied: pywin32 in c:\users\elcot\appdata\local\programs\python\python311\lib\site-packages (from pyttsx3) (3
05)
PS C:\Users\ELCOT> 
```

Figure3: Install Python packages.

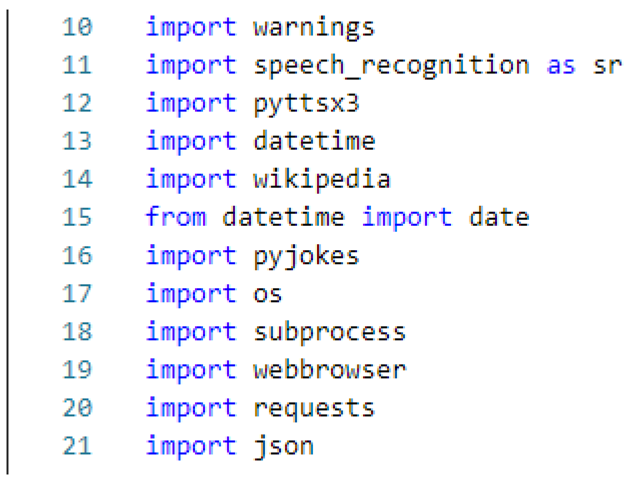
Features of Workmate

List of tasks performed by my personal assistant “WORKMATE” on my laptop are as follows:

1) Give voice as well as text response to the user greetings
2) It will answer to the some predefined questions like
    a. Who created you?
    b. What is your name?
    c. Search Wikipedia
    d. Workmate can tell current time
    e. Workmate can tell current date and day
    f. Workmate also tell jokes.
    g. Workmate can open the file explorer in my laptop
    h. Workmate can open YouTube
    i. It open Gmail account.
    j. It will able to show date and time in dd-mm-yyyy hh:mm format
    k. It can open google chrome by just voice command.
    l. It can able to Shut down my personal computer

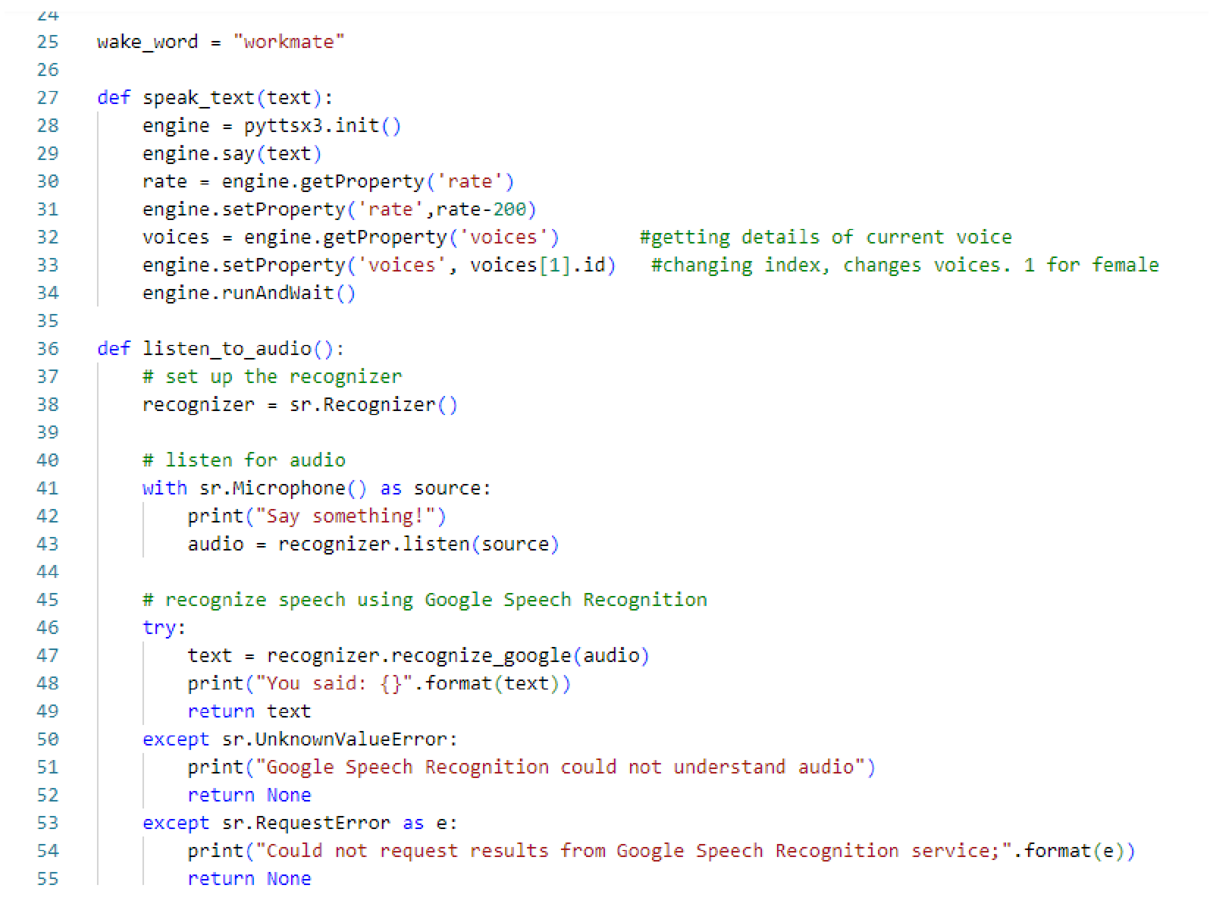
RESULTS AND IMPLEMENTATION

Imported packages



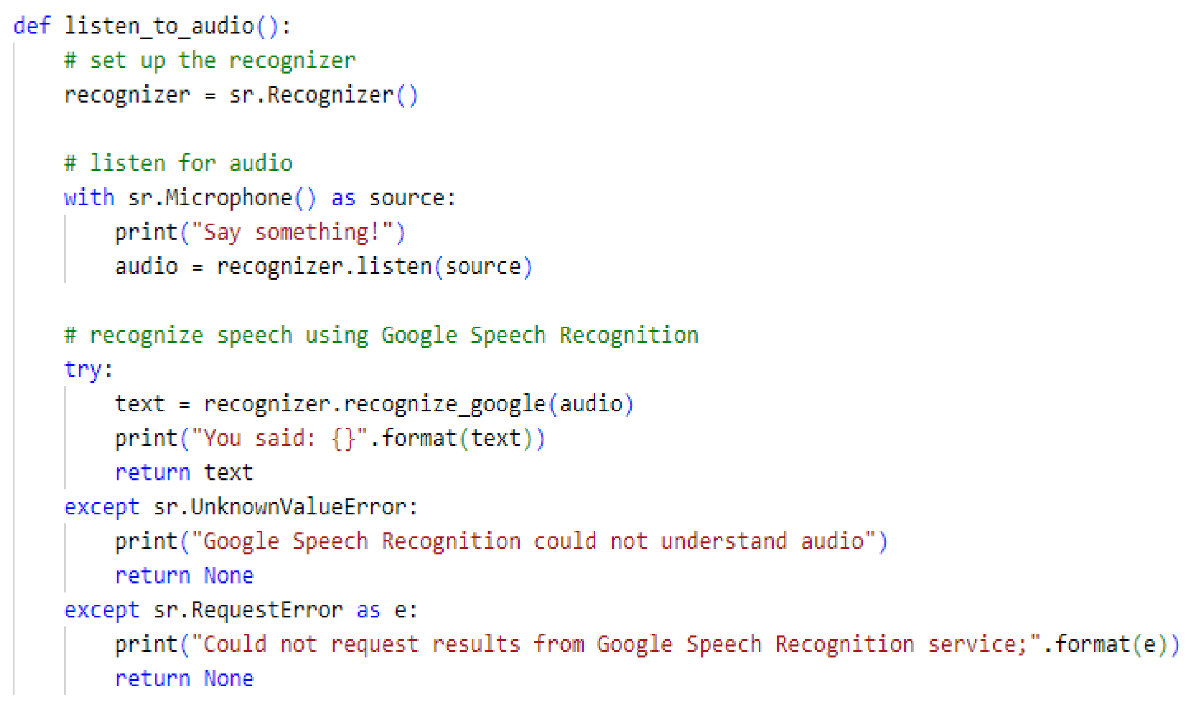
```python
import warnings
import speech_recognition as sr
import pyttsx3
import datetime
import wikipedia
from datetime import date
import pyjokes
import os
import subprocess
import webbrowser
import requests
import json
```

Set up recognizer voice

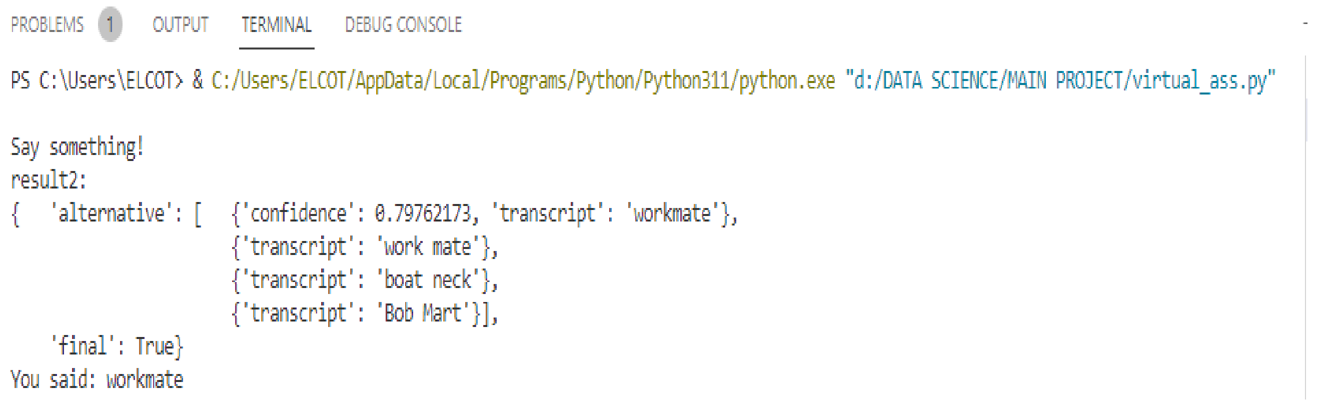


```python

wake_word = "workmate"

def speak_text(text):
    engine = pyttsx3.init()
    engine.say(text)
    rate = engine.getProperty('rate')
    engine.setProperty('rate',rate-200)
    voices = engine.getProperty('voices')       #getting details of current voice
    engine.setProperty('voices', voices[1].id)   #changing index, changes voices. 1 for female
    engine.runAndWait()

def listen_to_audio():
    # set up the recognizer
    recognizer = sr.Recognizer()

    # listen for audio
    with sr.Microphone() as source:
        print("Say something!")
        audio = recognizer.listen(source)

    # recognize speech using Google Speech Recognition
    try:
        text = recognizer.recognize_google(audio)
        print("You said: {}".format(text))
        return text
    except sr.UnknownValueError:
        print("Google Speech Recognition could not understand audio")
        return None
    except sr.RequestError as e:
        print("Could not request results from Google Speech Recognition service;".format(e))
        return None
```

User Input: The assistant will wait for the users to give voice command for further processing.



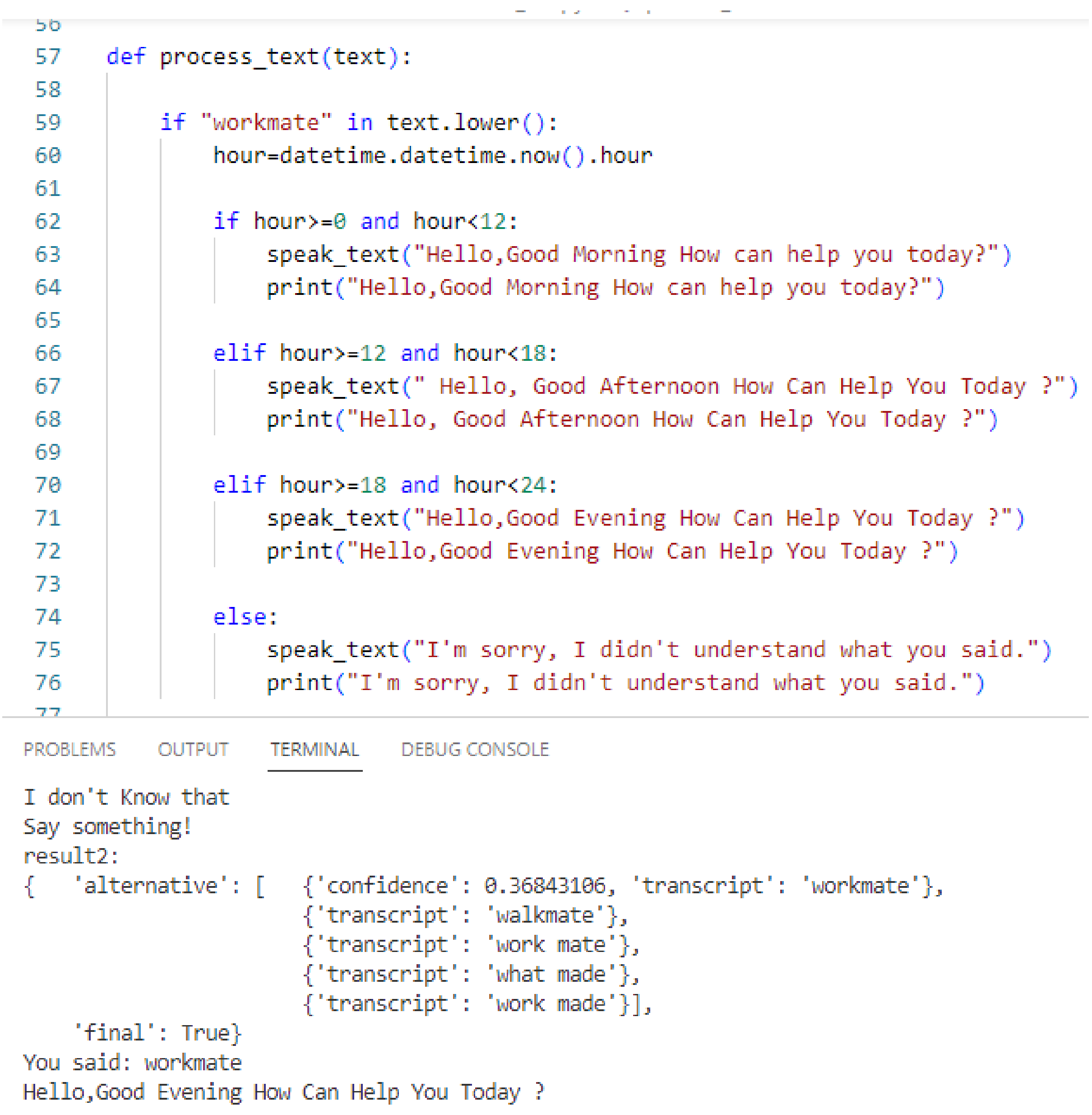
```python
def listen_to_audio():
    # set up the recognizer
    recognizer = sr.Recognizer()

    # listen for audio
    with sr.Microphone() as source:
        print("Say something!")
        audio = recognizer.listen(source)

    # recognize speech using Google Speech Recognition
    try:
        text = recognizer.recognize_google(audio)
        print("You said: {}".format(text))
        return text
    except sr.UnknownValueError:
        print("Google Speech Recognition could not understand audio")
        return None
    except sr.RequestError as e:
        print("Could not request results from Google Speech Recognition service;".format(e))
        return None
```



PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE

```
PS C:\Users\ELCOT> & C:/Users/ELCOT/AppData/Local/Programs/Python/Python311/python.exe "d:/DATA SCIENCE/MAIN PROJECT/virtual_ass.py"

Say something!
result2:
{   'alternative': [   {'confidence': 0.79762173, 'transcript': 'workmate'},
                       {'transcript': 'work mate'},
                       {'transcript': 'boat neck'},
                       {'transcript': 'Bob Mart'}],
    'final': True}
You said: workmate
```

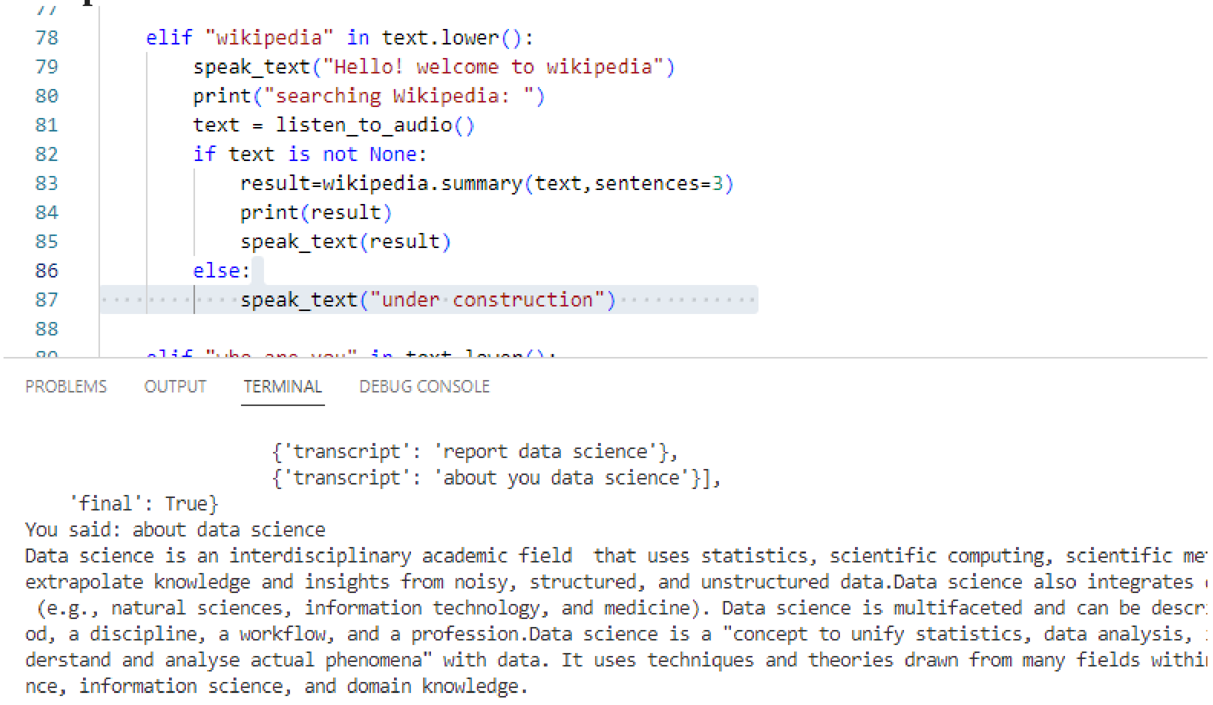
Greetings



```python
def process_text(text):

    if "workmate" in text.lower():
        hour=datetime.datetime.now().hour

        if hour>=0 and hour<12:
            speak_text("Hello,Good Morning How can help you today?")
            print("Hello,Good Morning How can help you today?")

        elif hour>=12 and hour<18:
            speak_text(" Hello, Good Afternoon How Can Help You Today ?")
            print("Hello, Good Afternoon How Can Help You Today ?")

        elif hour>=18 and hour<24:
            speak_text("Hello,Good Evening How Can Help You Today ?")
            print("Hello,Good Evening How Can Help You Today ?")

        else:
            speak_text("I'm sorry, I didn't understand what you said.")
            print("I'm sorry, I didn't understand what you said.")
```

```
I don't Know that
Say something!
result2:
{   'alternative': [   {'confidence': 0.36843106, 'transcript': 'workmate'},
                       {'transcript': 'walkmate'},
                       {'transcript': 'work mate'},
                       {'transcript': 'what made'},
                       {'transcript': 'work made'}],
    'final': True}
You said: workmate
Hello,Good Evening How Can Help You Today ?
```

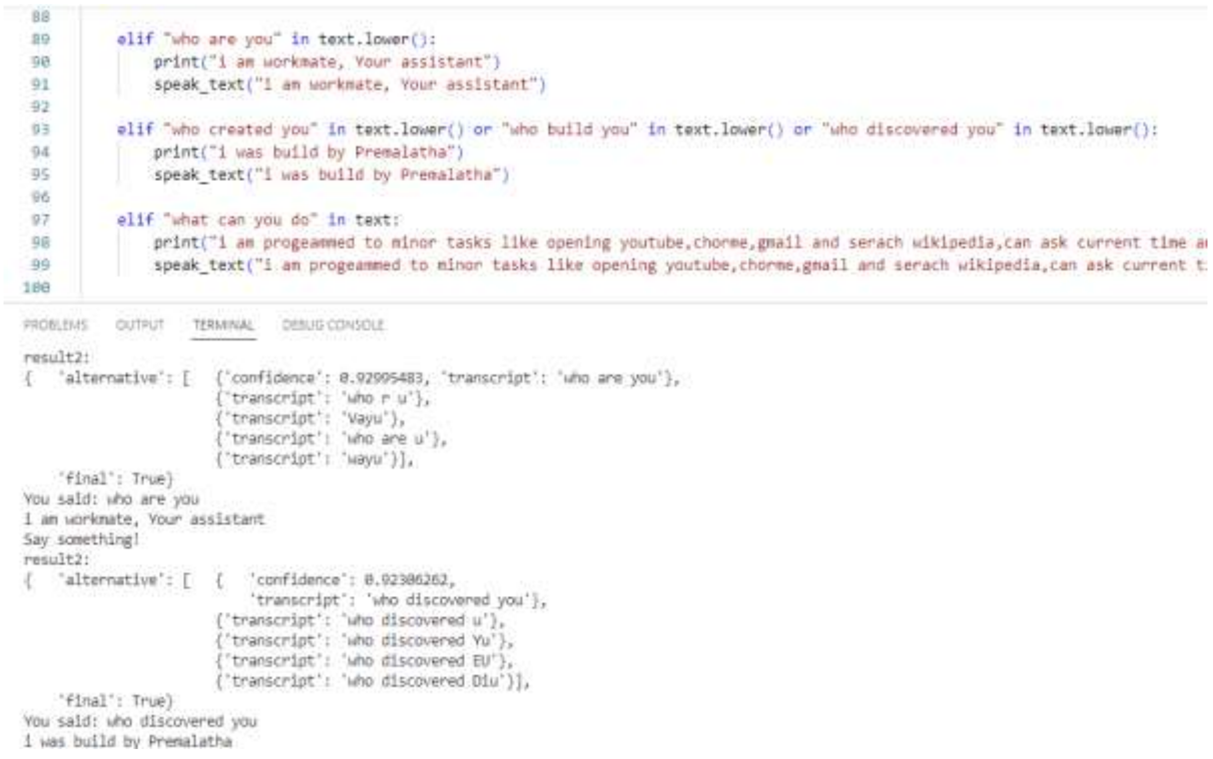
Wikipedia search



```python
    elif "wikipedia" in text.lower():
        speak_text("Hello! welcome to wikipedia")
        print("searching Wikipedia: ")
        text = listen_to_audio()
        if text is not None:
            result=wikipedia.summary(text,sentences=3)
            print(result)
            speak_text(result)
        else:
            speak_text("under construction")

```

```
                        {'transcript': 'report data science'},
                        {'transcript': 'about you data science'}],
    'final': True}
You said: about data science
Data science is an interdisciplinary academic field  that uses statistics, scientific computing, scientific me
extrapolate knowledge and insights from noisy, structured, and unstructured data.Data science also integrates
 (e.g., natural sciences, information technology, and medicine). Data science is multifaceted and can be descr
od, a discipline, a workflow, and a profession.Data science is a "concept to unify statistics, data analysis,
derstand and analyse actual phenomena" with data. It uses techniques and theories drawn from many fields withi
nce, information science, and domain knowledge.
```

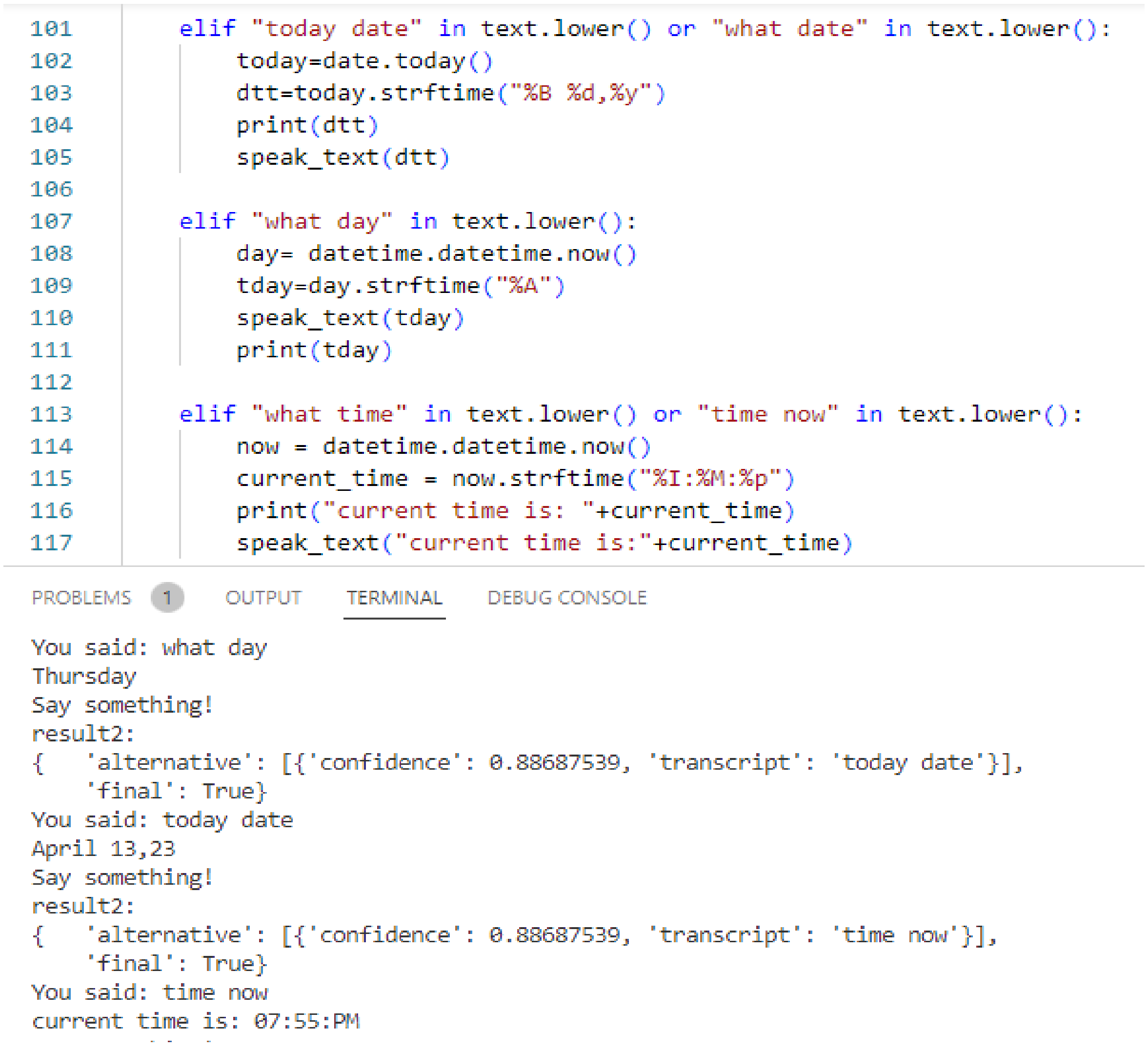
Some predefined question



```python

    elif "who are you" in text.lower():
        print("i am workmate, Your assistant")
        speak_text("i am workmate, Your assistant")

    elif "who created you" in text.lower() or "who build you" in text.lower() or "who discovered you" in text.lower():
        print("i was build by Premalatha")
        speak_text("i was build by Premalatha")

    elif "what can you do" in text:
        print("i am progeammed to minor tasks like opening youtube,chorme,gmail and serach wikipedia,can ask current time a
        speak_text("i am progeammed to minor tasks like opening youtube,chorme,gmail and serach wikipedia,can ask current t

```

```
result2:
{   'alternative': [   {'confidence': 0.92995483, 'transcript': 'who are you'},
                       {'transcript': 'who r u'},
                       {'transcript': 'Vayu'},
                       {'transcript': 'who are u'},
                       {'transcript': 'wayu'}],
    'final': True}
You said: who are you
i am workmate, Your assistant
Say something!
result2:
{   'alternative': [   {   'confidence': 0.92386262,
                           'transcript': 'who discovered you'},
                       {'transcript': 'who discovered u'},
                       {'transcript': 'who discovered Yu'},
                       {'transcript': 'who discovered EU'},
                       {'transcript': 'who discovered Diu'}],
    'final': True}
You said: who discovered you
i was build by Premalatha
```

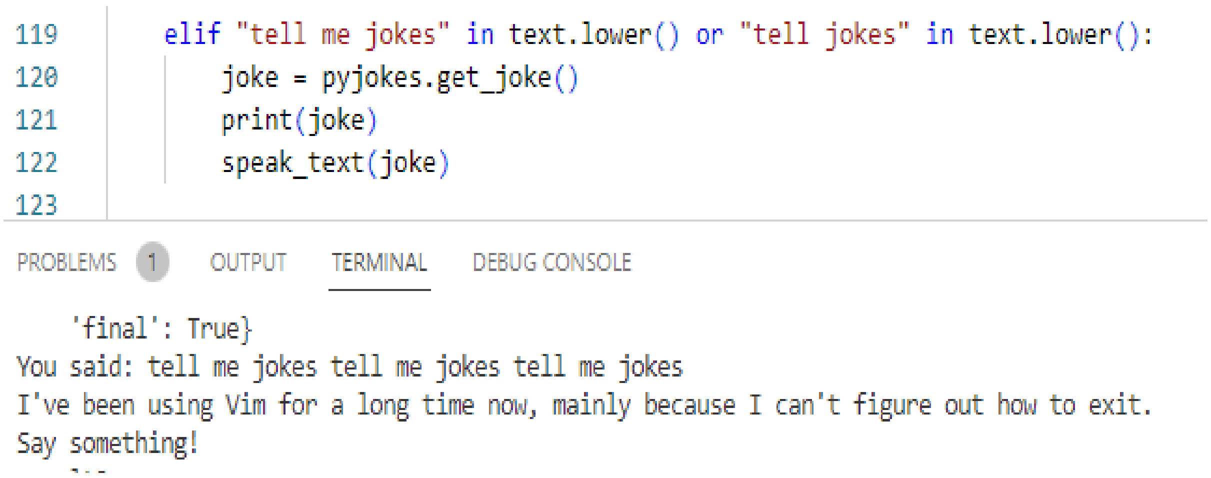
Date and time



```
    elif "today date" in text.lower() or "what date" in text.lower():
        today=date.today()
        dtt=today.strftime("%B %d,%y")
        print(dtt)
        speak_text(dtt)

    elif "what day" in text.lower():
        day= datetime.datetime.now()
        tday=day.strftime("%A")
        speak_text(tday)
        print(tday)

    elif "what time" in text.lower() or "time now" in text.lower():
        now = datetime.datetime.now()
        current_time = now.strftime("%I:%M:%p")
        print("current time is: "+current_time)
        speak_text("current time is:"+current_time)
```

```
You said: what day
Thursday
Say something!
result2:
{   'alternative': [{'confidence': 0.88687539, 'transcript': 'today date'}],
    'final': True}
You said: today date
April 13,23
Say something!
result2:
{   'alternative': [{'confidence': 0.88687539, 'transcript': 'time now'}],
    'final': True}
You said: time now
current time is: 07:55:PM
```

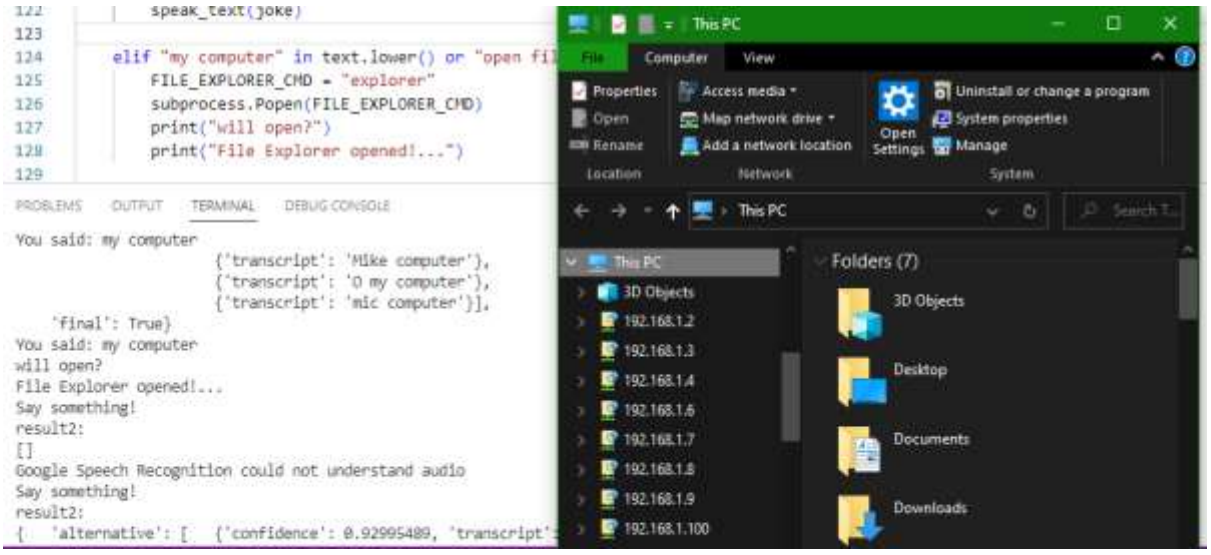
Pyjokes



```
    elif "tell me jokes" in text.lower() or "tell jokes" in text.lower():
        joke = pyjokes.get_joke()
        print(joke)
        speak_text(joke)

```

```
    'final': True}
You said: tell me jokes tell me jokes tell me jokes
I've been using Vim for a long time now, mainly because I can't figure out how to exit.
Say something!
```

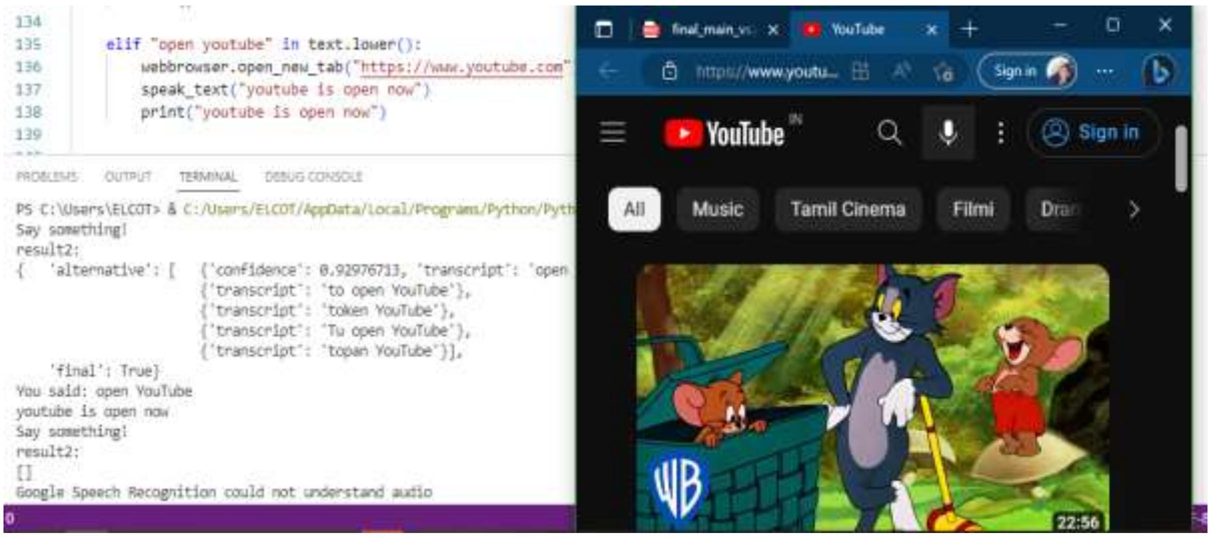
Open File explorer



```
        speak_text(joke)

    elif "my computer" in text.lower() or "open fil
        FILE_EXPLORER_CMD = "explorer"
        subprocess.Popen(FILE_EXPLORER_CMD)
        print("will open?")
        print("File Explorer opened!...")

```

```
You said: my computer
                            {'transcript': 'Mike computer'},
                            {'transcript': 'O my computer'},
                            {'transcript': 'mic computer'}],
    'final': True}
You said: my computer
will open?
File Explorer opened!...
Say something!
result2:
[]
Google Speech Recognition could not understand audio
Say something!
result2:
{   'alternative': [   {'confidence': 0.92995489, 'transcript'
```

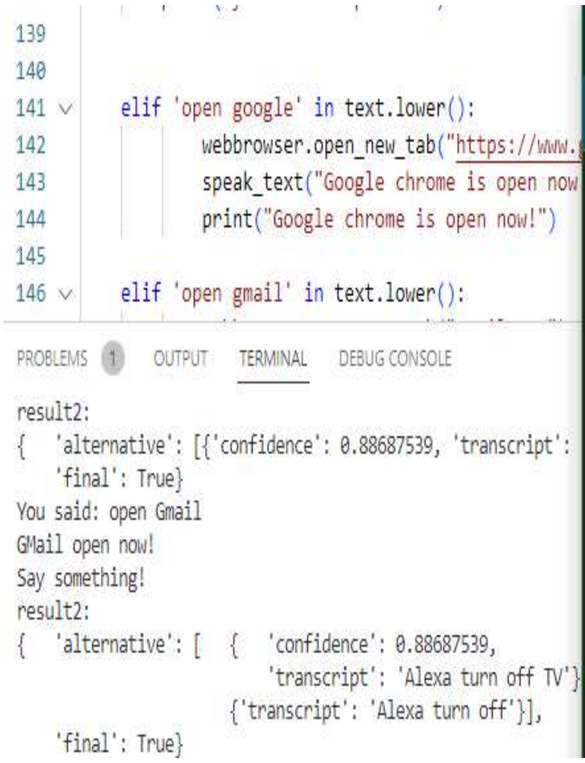
Open YouTube



```

    elif "open youtube" in text.lower():
        webbrowser.open_new_tab("https://www.youtube.com"
        speak_text("youtube is open now")
        print("youtube is open now")

```

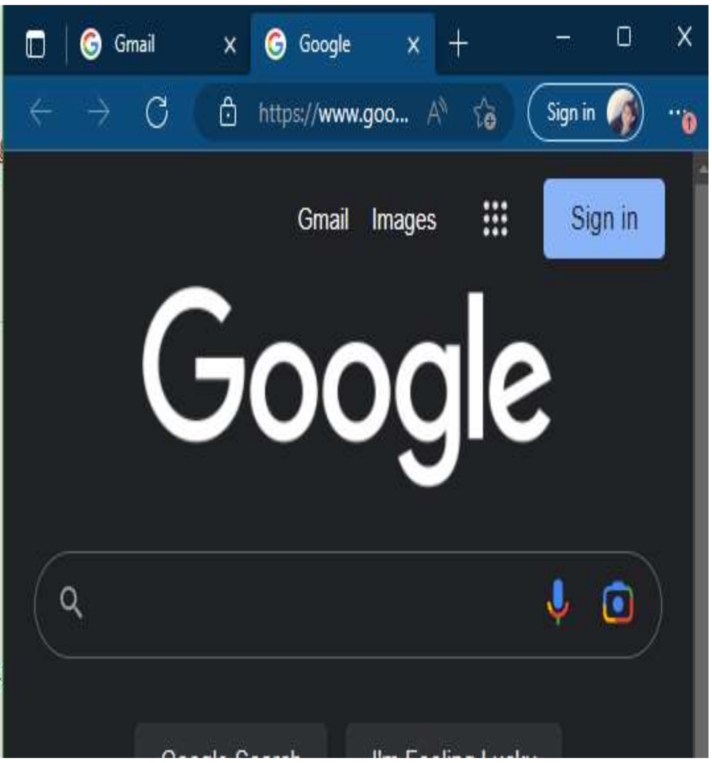
```
PS C:\Users\ELCOT> & C:/Users/ELCOT/AppData/Local/Programs/Python/Pyth
Say something!
result2:
{   'alternative': [   {'confidence': 0.92976713, 'transcript': 'open
                       {'transcript': 'to open YouTube'},
                       {'transcript': 'token YouTube'},
                       {'transcript': 'Tu open YouTube'},
                       {'transcript': 'topan YouTube'}],
    'final': True}
You said: open YouTube
youtube is open now
Say something!
result2:
[]
Google Speech Recognition could not understand audio
```

Open Google

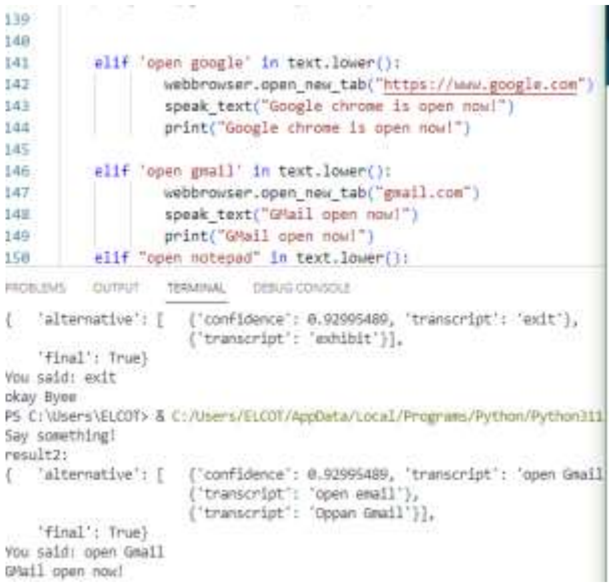


```
    elif 'open google' in text.lower():
            webbrowser.open_new_tab("https://www.
            speak_text("Google chrome is open now
            print("Google chrome is open now!")

    elif 'open gmail' in text.lower():
```

```
result2:
{   'alternative': [{'confidence': 0.88687539, 'transcript':
    'final': True}
You said: open Gmail
GMail open now!
Say something!
result2:
{   'alternative': [   {   'confidence': 0.88687539,
                           'transcript': 'Alexa turn off TV'},
                       {'transcript': 'Alexa turn off'}],
    'final': True}
```

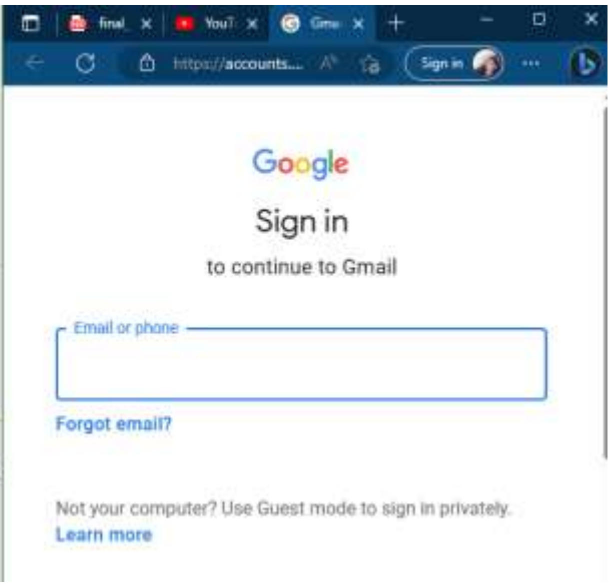


Open Gmail

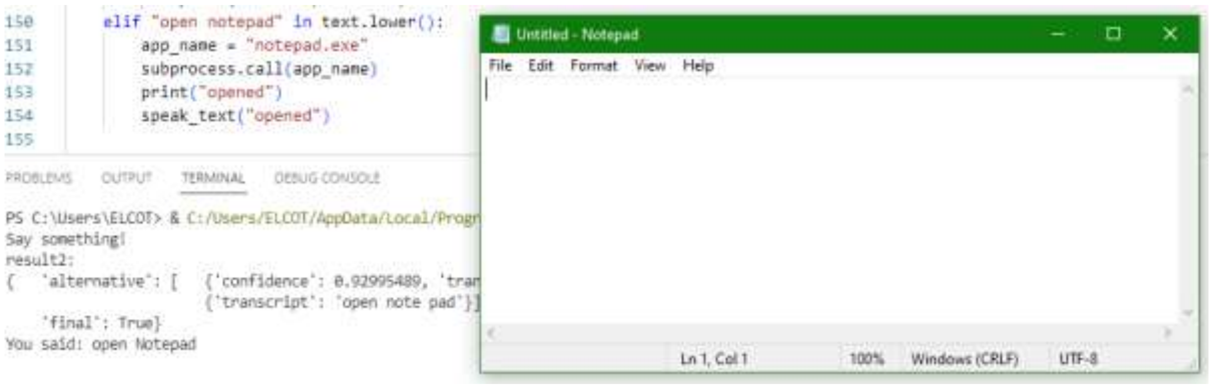


```
    elif 'open google' in text.lower():
            webbrowser.open_new_tab("https://www.google.com")
            speak_text("Google chrome is open now!")
            print("Google chrome is open now!")

    elif 'open gmail' in text.lower():
            webbrowser.open_new_tab("gmail.com")
            speak_text("GMail open now!")
            print("GMail open now!")
    elif "open notepad" in text.lower():
```

```
{   'alternative': [   {'confidence': 0.92995489, 'transcript': 'exit'},
                       {'transcript': 'exhibit'}],
    'final': True}
You said: exit
okay Byee
PS C:\Users\ELCOT> & C:/Users/ELCOT/AppData/Local/Programs/Python/Python311
Say something!
result2:
{   'alternative': [   {'confidence': 0.92995489, 'transcript': 'open Gmail
                       {'transcript': 'open email'},
                       {'transcript': 'Oppan Gmail'}],
    'final': True}
You said: open Gmail
GMail open now!
```



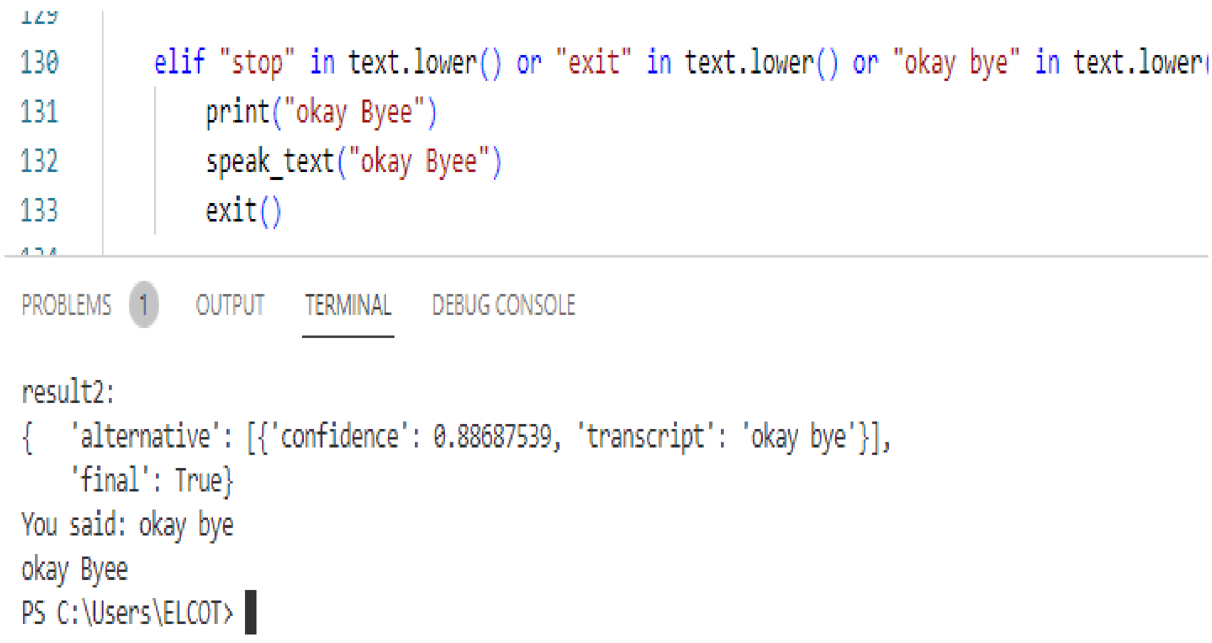
Open Notepad



```
    elif "open notepad" in text.lower():
        app_name = "notepad.exe"
        subprocess.call(app_name)
        print("opened")
        speak_text("opened")
```

```
PS C:\Users\ELCOT> & C:/Users/ELCOT/AppData/Local/Progr
Say something!
result2:
{   'alternative': [   {'confidence': 0.92995489, 'tran
                       {'transcript': 'open note pad'}]
    'final': True}
You said: open Notepad
```

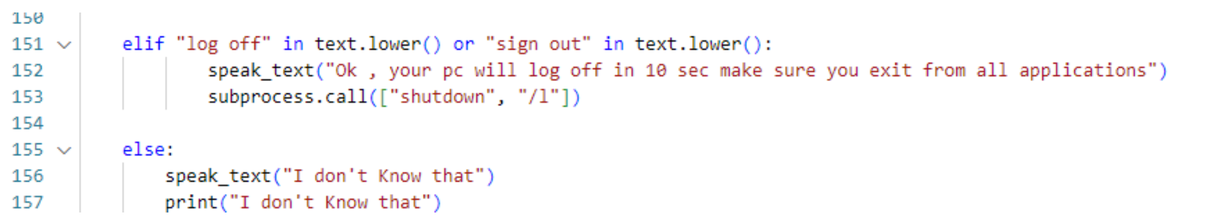
Exit the Program



```
    elif "stop" in text.lower() or "exit" in text.lower() or "okay bye" in text.lower(
        print("okay Byee")
        speak_text("okay Byee")
        exit()
```

```
result2:
{   'alternative': [{'confidence': 0.88687539, 'transcript': 'okay bye'}],
    'final': True}
You said: okay bye
okay Byee
PS C:\Users\ELCOT> █
```

Shut down



```
    elif "log off" in text.lower() or "sign out" in text.lower():
            speak_text("Ok , your pc will log off in 10 sec make sure you exit from all applications")
            subprocess.call(["shutdown", "/l"])

    else:
        speak_text("I don't Know that")
        print("I don't Know that")
```

CONCLUSION

In this Project, discussed about Personal Virtual Assistant For Windows Using Python. Virtual assistant makes life easier to humans. Virtual assistant is the flexibility to contract for just the services they need. As like Alexa, Cortona, Siri, Google assistant and also make virtual assistant using python for all windows versions. Use Artificial Intelligence technology for this project. Virtual Personal Assistants are effective way to manage or organize your schedule. Virtual Personal assistants are also reliable than Human Personal Assistant because, virtual personal Assistants are more portable, loyal and available to use anytime. Our virtual assistant will be intimate you with suggestions and taking instructions, and will know more about you. Can expect this device to be permanent.

FUTURE SCOPE

The future scope of voice-based virtual assistants is vast and promising. With advancements in technology and AI, virtual assistants are expected to become even more intelligent, intuitive, and personalized. Some of the potential future developments in voice-based virtual assistants are:

Multilingual support: As virtual assistants become more popular worldwide, there will be a need for multilingual support. Future virtual assistants are expected to be able to understand and respond in multiple languages, making them more accessible to users globally.

Integration with smart homes: With the rise of smart home devices, virtual assistants are expected to be integrated with them to control home appliances such as lights, thermostats, and security systems.

Emotional intelligence: Future virtual assistants are expected to be able to detect and respond to emotions. They will be able to understand human emotions and respond appropriately, making the interaction with them more human-like.

Personalization: Virtual assistants are expected to become more personalized to the user's needs and preferences. They will be able to learn from user interactions and adapt to their needs, providing more relevant and customized responses.

Natural language processing: Natural language processing (NLP) is an AI technique that allows virtual assistants to understand and respond to natural language inputs. Future virtual assistants are expected to be equipped with advanced NLP algorithms that can interpret more complex inputs and provide more accurate and relevant responses.

Augmented reality: With the advent of augmented reality (AR) technology, virtual assistants are expected to be integrated with AR devices, allowing users to interact with them in a more immersive way.

Better security and privacy: As virtual assistants become more ubiquitous, there will be a need for better security and privacy measures to protect user data. Future virtual assistants are expected to be equipped with better security and privacy features to protect user information.

The future scope of voice-based virtual assistants is vast and exciting. With advancements in technology and AI, virtual assistants are expected to become even more intelligent, intuitive, and personalized, providing users with a more efficient and convenient way of completing various tasks. Additionally, researchers evaluated the dialogue system using a variety of cloud servers, including Amazon Web Services and Google Cloud. In the second stage, researchers used fresh machine learning models, such as Convolutional Neural Networks and Deep Neural Networks, to test the Graph Model. With the testing and training data sets and gathered from various people and environments such as various backgrounds and positions in rooms, various user distances from the camera, and various user characters next evaluated and trained the model. In the third step, researchers are attempting to track and analyse nonverbal movements as well as the user's facial expression and body language using the Kinect sensors and camera. After testing each system step and taking into account the model outcomes, the final stage.

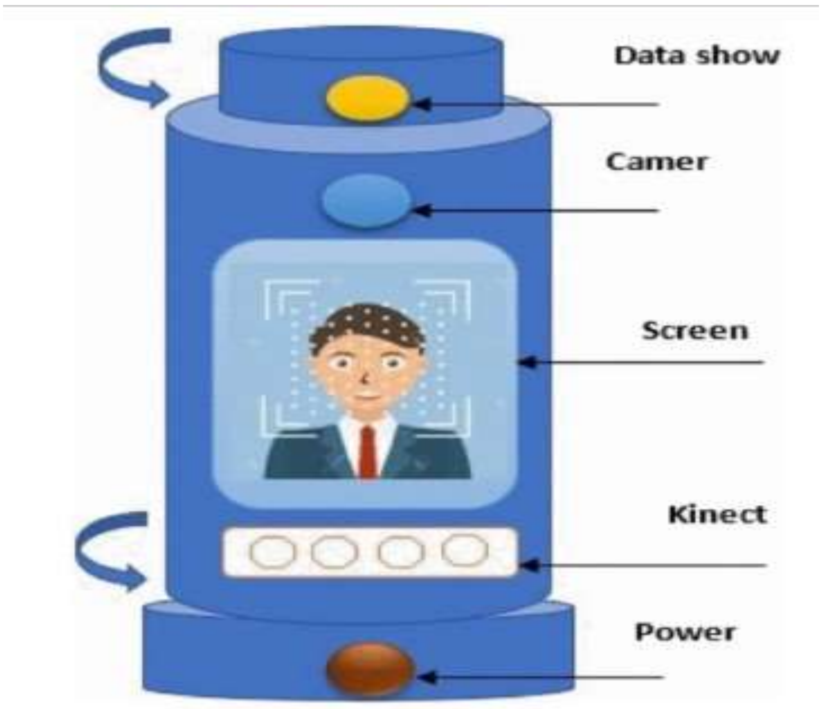


Figure 4: Future of Virtual Assistant

REFERENCES

1. Elizabeth Sucupira Furtado, Virgilio Almedia And Vasco Furtado, “Personal Digital Assistants: The Need Of Governance”

2. Zecheng Zhan, Virgilio Almedia, And Meina Song, “Table-To-Dialog: Building Dialog Assistants To Chat With People On Behalf Of You”

3. Yusuf Ugurlu, Murat Karabulut, Islam Mayda “A Smart Virtual Assistant Answering Questions About COVID-19” Mathangi Sri “NLP In Virtual Assistants”

4. Anxo Pérez, Paula Lopez-Otero, Javier Parapar. “Designing An open-Source Virtual Assistant”

Websites

1. https://www.researchgate.net/publication/351290964_The_Voice_Enabled_Personal_Assistant_for_Pc_using_Python
2. https://www.irjmets.com/uploadedfiles/paper/volume3/issue_7_july_2021/14275/1628083554.pdf
3. https://www.academia.edu/40229169/PROJECT_REPORT_ON_VIRTUAL_ASSISTANT_SUBMITTED_BY_NAAZNEEN_JATU