MINI PROJECT REPORT

Voice Assistant Using Python

ACKNOWLEDGEMENT

The merciful guidance bestowed to us by the almighty made us stick to this project to a successful end. We humbly pray with sincere heart for his guidance to continue forever.

We pay thanks to our project guide **Dr. Pankaj Kumar** who has given guidance and light to us during this project. His versatile knowledge has helped us in the critical times during the span of this project.

We pay special thanks to our Head of Department **Dr. Sansar Singh Chauhan** who has been always present as a support and help us in all possible ways during this project.

We also take this opportunity to express our gratitude to all those who have been directly and indirectly with us during the completion of the project.

We want to thank our friends who have always encouraged us during this project.

At last thanks to all the faculty of CSE department who provided valuable suggestions during the period of project.

ABSTRACT

Voice assistants using Python can be developed with a variety of libraries and APIs, such as speech recognition, \pyttsx3, and PyAudio. The basic idea is to allow users to interact with a computer system through voice commands, and receive responses in natural language.

To create a voice assistant using Python , the following steps are generally involved.

1. Installing the required libraries: First, the necessary Python libraries such as speech recognition, pyttsx3, and PyAudio need to be installed.

2. Setting up the microphone: A microphone is needed to capture the user's voice input. PyAudio can be used to record audio from the microphone.

3. Speech recognition: The speech recognition library can be used to convert the user's spoken words into text that the computer can understand.

4. Text-to-speech conversion: The pyttsx3 library can be used to convert the computer's response into speech.

5. Developing the functionality: Depending on the application, various functionalities can be developed, such as answering questions, providing information, or performing tasks.

Overall, creating a voice assistant using Python can be a fun and educational project, and can also have practical applications in various domains such as home automation, customer service, and healthcare.

INDEX

INTRODUCTION

Voice assistants using Python are computer programs that use speech recognition and natural language processing to interpret spoken commands and respond with synthesized speech or text. Python is a popular programming language for building voice assistants because of its simplicity, flexibility, and rich set of libraries and APIs.

A voice assistant built with Python can have a wide range of applications, including personal assistants, smart home devices, customer service bots, and voice-controlled applications for people with disabilities. The goal of a voice assistant is to provide a more natural and convenient interface for users to interact with computers, especially in situations where typing or using a touchscreen is not practical or desirable.

To build a voice assistant using Python, various techniques and tools can be used, such as speech recognition, natural language processing, text-to-speech conversion, and machine learning. The development process involves designing the user interface, defining the functionalities, integrating with other systems, and testing and refining the performance.

Overall, a voice assistant built with Python can be a useful and engaging tool that enhances the user experience and opens up new possibilities for human-computer interaction.

A Python-based voice assistant can be a powerful tool for automating tasks and providing personalized services to users. For example, a voice assistant can be developed to control smart home devices, answer questions, play music, or provide news updates. Voice assistants can also be used in healthcare to monitor patients and provide support for medical conditions.

Python-based voice assistants can be developed with the help of libraries such as speech recognition, pyttsx3, and PyAudio. These libraries provide the necessary tools for capturing voice input, recognizing speech, and generating speech output. With these tools, developers can create voice assistants that can interact with users in a natural and intuitive way.

Voice assistants have become an increasingly popular way for users to interact with computers and smart devices. Using natural language processing and speech recognition technologies, voice assistants allow users to perform tasks, get information, and control devices through simple voice commands. Python, a popular programming language, can be used to create voice assistants with the help of various libraries and APIs.

MODULES

There are several modules that can be used to create a voice assistant in Python. Some of the most popular modules are:

1. SpeechRecognition: This module provides speech recognition functionality and can recognize speech from different sources, such as a microphone, an audio file, or a pre-recorded speech.

2. Pyttsx3: This module provides text-to-speech conversion functionality and can be used to generate speech output in response to user commands.

3. PyAudio: This module provides the necessary tools for recording audio from a microphone, which can be useful for capturing user input.

4. Wikipedia-API: This module provides an easy-to-use interface for accessing the Wikipedia API, which can be used to extract information from Wikipedia in response to user queries.

5. Requests: This module allows developers to send HTTP/1.1 requests to APIs, which can be used to access web services and perform various tasks.

6. OpenCV: This module provides computer vision functionality and can be used to detect faces and facial expressions, which can be useful for building more advanced voice assistants that can interact with users in a more natural and intuitive way.

There are many other modules and APIs available that can be used to create a voice assistant in Python, depending on the specific needs and requirements of the project.

SYSTEM ANALYSIS

A system analysis for a voice assistant using Python involves the process of identifying and analyzing the requirements and features of the system to be developed. This includes understanding the needs and goals of the users, as well as the technical constraints and resources available for the development of the system.

Here are some key steps that can be involved in a system analysis for a voice assistant using Python:

1. Identify the key features and functionalities of the system: This involves identifying the various tasks that the voice assistant will perform, such as answering questions, providing information, controlling devices, or performing tasks. The system analysis should also identify the various user interactions with the voice assistant, such as voice commands, touch screen input, or other input methods.

2. Understand the user needs and goals: This involves understanding the needs and goals of the users who will be interacting with the voice assistant. The system analysis should take into account the specific user profiles, including demographics, preferences, and use cases.

3. Identify the technical requirements and constraints: This involves identifying the technical requirements and constraints for the development of the voice assistant. This includes the hardware and software platforms that will be used, the programming languages and libraries that will be used, as well as any other technical constraints such as processing power, memory, or network connectivity.

4. Develop a system architecture: This involves developing a high-level architecture of the system, including the various components, modules, and interfaces that will be used to implement the system.

5. Evaluate the feasibility of the system: This involves evaluating the feasibility of the system, taking into account the technical requirements, the user needs and goals, and the resources available for the development of the system.

Overall, a system analysis for a voice assistant using Python is an important step in the development process, as it helps to ensure that the system is designed to meet the needs of the users, while also being feasible to implement given the available technical and resource constraints.

SOFTWARE REQUIREMENTS

Here are some of the software requirements for developing a voice assistant using Python:

1. Python: A programming language that provides a wide range of libraries and modules that are essential for developing a voice assistant. The latest version of Python can be downloaded from the official website.

2. SpeechRecognition: A Python library that provides speech recognition functionality, allowing the voice assistant to recognize speech from different sources, such as a microphone or an audio file.

3. Pyttsx3: A Python library that provides text-to-speech functionality, allowing the voice assistant to generate speech output in response to user commands.

4. PyAudio: A Python library that provides the necessary tools for recording audio from a microphone, which is essential for capturing user input.

5. Wikipedia-API: A Python library that provides an easy-to-use interface for accessing the Wikipedia API, which can be used to extract information from Wikipedia in response to user queries.

6. Requests: A Python library that allows developers to send HTTP/1.1 requests to APIs, which can be used to access web services and perform various tasks.

7. OpenCV: A Python library that provides computer vision functionality and can be used to detect faces and facial expressions, which can be useful for building more advanced voice assistants that can interact with users in a more natural and intuitive way.

Other libraries and modules can also be used depending on the specific requirements of the project. Additionally, an integrated development environment (IDE) such as PyCharm or Visual Studio Code can be used to write and debug the code. Finally, the voice assistant can be deployed on a device or a cloud platform such as Amazon Web Services or Microsoft Azure.

HARDWARE REQUIREMENTS

The hardware requirements for a voice assistant using Python will depend on the specific needs and requirements of the project. Here are some of the common hardware components that may be required:

1. Microphone: A microphone is essential for capturing voice input from the user. A high-quality microphone can help improve the accuracy and reliability of the voice recognition system.

2. Speaker: A speaker is necessary for providing speech output to the user. A good quality speaker can improve the overall user experience.

3. Processor: The processor is responsible for executing the voice assistant software and handling the various tasks, such as speech recognition, natural language processing, and text-to-speech conversion. A fast and powerful processor can improve the speed and responsiveness of the system.

4. Memory: Sufficient memory is necessary to store the various data and models used by the voice assistant. This includes language models, voice models, and other resources used for speech recognition and natural language processing.

5. Storage: Adequate storage is required to store the voice assistant software and any associated data, such as audio recordings and other files.

6. Network connectivity: The voice assistant may require network connectivity to access online resources, such as web services or databases. A stable and fast internet connection can help improve the reliability and speed of the system.

7. Display: A display may be required if the voice assistant has a graphical user interface or needs to display information to the user.

The specific hardware requirements will depend on the use case and the requirements of the voice assistant. For example, a voice assistant designed to control smart home devices may require additional hardware components, such as sensors and actuators, to interact with the devices.

TECHNOLOGY

Python

Python is a high-level, interpreted programming language that is widely used for general-purpose programming. It was first released in 1991 by Guido van Rossum and has since become one of the most popular programming languages in the world.

One of the reasons for Python's popularity is its ease of use and readability. Its syntax is designed to be simple and easy to understand, which makes it a great language for beginners. Additionally, Python has a large standard library, which means that many common tasks, such as file I/O and network programming, can be done with built-in modules.

Python is also known for its versatility. It can be used for a wide variety of tasks, including web development, data analysis, scientific computing, artificial intelligence, and more. It is also supported on many different platforms, including Windows, macOS, Linux, and various mobile platforms.

Some key features of Python include:

I. Dynamically typed: Unlike statically-typed languages such as Java or C++, Python does not require variable types to be explicitly defined. Instead, the interpreter infers the type of a variable at runtime.

II. Interpreted: Python code is interpreted at runtime, which means that there is no need to compile the code before running it. This makes it easy to write and test code.

III. Object-oriented: Python is an object-oriented language, which means that it allows you to define classes and objects, and to use inheritance and other object-oriented programming concepts.

IV. Garbage collection: Python has automatic memory management, which means that it automatically cleans up memory that is no longer being used.

V. Large standard library: Python comes with a large standard library that provides many useful modules for performing common tasks, such as file I/O, network programming, and more.

Overall, Python is a powerful and versatile language that is well-suited to a wide variety of programming tasks, making it a great choice for both beginners and experienced programmers.

Pyttsx3

Pyttsx is a Python package that provides a cross-platform text-to-speech (TTS) API. It allows developers to write Python scripts that can convert written text into spoken words, using different TTS engines.

Pyttsx is easy to use, and it supports different TTS engines, such as SAPI5 on Windows, NSSpeechSynthesizer on Mac OS X, and eSpeak on Linux. It also provides different options for controlling the speed, volume, and pitch of the spoken text.

Here's an example of how to use pyttsx to convert text to speech:

```
import pyttsx

engine = pyttsx.init()

engine.say("Hello, how are you?")

engine.runAndWait()
```

This will initialize the TTS engine, say "Hello, how are you?", and wait for the speech to finish before continuing with the script.

PyAudio

PyAudio is a Python library for working with audio. It provides bindings for the PortAudio library, which is a cross-platform audio I/O library that allows you to record and play back audio in real time.

PyAudio can be used to record audio from a microphone or other input device, or to play back audio through a speaker or other output device. It supports a variety of audio formats, including WAV, AIFF, and MP3.

Here's an example of how to use PyAudio to record audio from a microphone:

```
import pyaudio

import wave

# set up the audio stream

FORMAT = pyaudio.paInt16

CHANNELS = 1

RATE = 44100

CHUNK = 1024

RECORD_SECONDS = 5

WAVE_OUTPUT_FILENAME = "output.wav"

p = pyaudio.PyAudio()

stream = p.open(format=FORMAT, channels=CHANNELS,

            rate=RATE, input=True,

            frames_per_buffer=CHUNK)


# record audio

frames = []


for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):

   data = stream.read(CHUNK)

   frames.append(data)


# stop the stream and close the audio device

stream.stop_stream()

stream.close()

p.terminate()
```

```
# save the recorded audio to a file
wf = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
wf.setnchannels(CHANNELS)
wf.setsampwidth(p.get_sample_size(FORMAT))
wf.setframerate(RATE)
wf.writeframes(b''.join(frames))
wf.close()
```

This code sets up the PyAudio stream, records audio for five seconds, and saves the recorded audio to a WAV file. Note that you may need to install the PyAudio library and its dependencies before you can use it in your Python environment.

Here are some of the key technologies that may be used in building a voice assistant using Python:

1. Speech recognition: This technology is used to convert spoken words into text that the voice assistant can interpret. There are several Python libraries that can be used for speech recognition, such as SpeechRecognition and Google Cloud Speech-to-Text API.

2. Natural language processing (NLP): This technology is used to analyze and understand the meaning of natural language input. NLP can be used to extract relevant information from spoken commands or questions, and to generate appropriate responses. Some popular Python libraries for NLP include Natural Language Toolkit (NLTK), spaCy, and TextBlob.

3. Text-to-speech (TTS): This technology is used to convert text generated by the voice assistant into spoken words. Python libraries such as pyttsx and Google Text-to-Speech API can be used for TTS.

4. Dialog management: This technology is used to manage the conversation flow between the user and the voice assistant. Dialog management can be used to keep track of the user's context and history, and to generate appropriate responses based on that information. Libraries such as Rasa and ChatterBot can be used for building conversational agents.

5. Machine learning: This technology can be used to improve the accuracy of speech recognition and NLP. Machine learning algorithms can be trained on large amounts of data to improve the accuracy of the voice assistant. Libraries such as scikit-learn and TensorFlow can be used for machine learning in Python.

Pywhatkit

Pywhatkit is a Python library that provides various useful functions such as sending WhatsApp messages, playing videos on YouTube, searching on Google, converting text to handwriting, and more. However, it does not have built-in voice assistant functionality.

To create a voice assistant in Python, you can use libraries like SpeechRecognition and pyttsx3. SpeechRecognition allows you to recognize speech input from a microphone, while pyttsx3 can be used to generate speech output.

Smtplib

Smtplib is a Python library used for sending email messages using the Simple Mail Transfer Protocol (SMTP). The SMTP protocol is used for sending email messages between servers, and smtplib provides an easy-to-use interface for sending email messages from a Python script.

CV

CV in Python typically refers to computer vision, which is a field of study that focuses on how computers can be made to interpret and analyze visual data from the world. In Python, there are several popular libraries and frameworks for computer vision, such as OpenCV, scikit-image, and TensorFlow.

OpenCV (Open Source Computer Vision) is a popular computer vision library that provides a wide range of functions for image and video processing, such as image filtering, edge detection, object recognition, and more. Here's an example of how to use OpenCV to read and display an image:

CODING

```
import pyttsx3 #pip install pyttsx3
import speech_recognition as sr #pip install speechRecognition
import datetime
import wikipedia #pip install wikipedia
import webbrowser
import os
import smtplib #pip install secure-smtplib
import cv2 #pip install opencv-python
import random
import requests
from requests import get
import pywhatkit as kit  #pip install pywhatkit
import sys
import pyjokes #pip install pyjokes

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
print(voices[3].id)
engine.setProperty('voice', voices[3].id)

#text to speech
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!")

elif hour>=12 and hour<18:
```

```
speak("Good Afternoon!")

    else:
        speak("Good Evening!")

    speak("i am  ,sir! please tell me how may i help you")

#to convert voice into text
def takeCommand():
    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='hindi-ind')
        print(f"User said: {query}\n")

    except Exception as e:
        # print(e)
        print("Say that again please...")
        speak("Say that again please")
        return "None"
    return query

def sendEmail(to, content):
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.ehlo()
    server.starttls()
    server.login('ritik691singh@gmail.com', 'Ritik895@')
    server.sendmail('ritik691singh@gmail.com', to, content)
    server.close()
```

```
#for news update
def news():
    main_url = 'http://newsapi.org/v2/top-
headlines?sources=techcrunch&apiKey=9548982f2d484871bc150661464674fb'

    main_page = requests.get(main_url).json()
    print(main_page)
    articles = main_page["articles"]
    print(articles)
    head = []
    day = ["first", "second", "third", "fourth", " fifth", "sixth", "seventh", "eighth", "ninth", "tenth"]
    for ar in articles:
        head.append(ar["title"])
    for i in range (len(day)):
        print(f"today's {day[i]} news is: ", head[i])
        speak(f"today's {day[i]} news is: {head[i]}")

if _name_ == "_main_":
    wishMe()
    while True:
    # if 1:
        query = takeCommand().lower()

        # Logic for executing tasks based on query

        if 'open google' in query:
            speak("sir, what should i search on google")
            cm = takeCommand().lower()
            webbrowser.open(f"{cm}")

        elif 'open youtube' in query:
            speak("sir, what should i search on youtube")
            ct = takeCommand().lower()
```

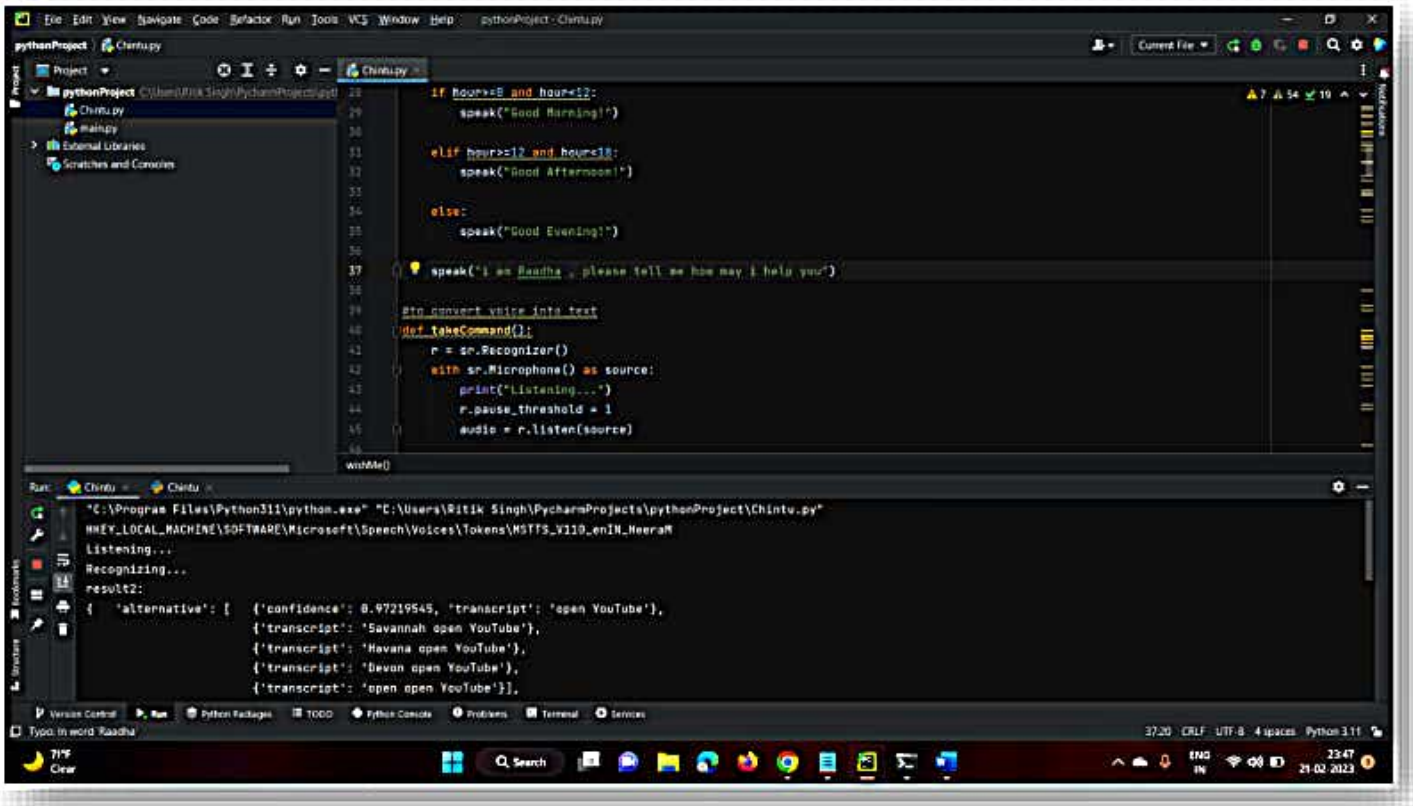
```
    kit.playonyt(f"{ct}")

elif 'open stackoverflow' in query:
    webbrowser.open("stackoverflow.com")

elif 'open facebook' in query:
    webbrowser.open("facebook.com")

elif 'open command prompt' in query:
    os.system("start cmd")

elif 'who made you' in query or 'who created you' in query:
    speak("I have been created by Saurabh.")

elif 'tell me joke' in query:
    joke = pyjokes.get_joke()
    speak(joke)

elif ' power point presentation' in query or 'open ppt' in query:
    speak("opening power point presentation")
    power = r"Presentation of voice assistant.pptx"
    os.startfile(power)

elif 'what is your name' in query or 'who are you' in query:
    speak("my friends call me")
    speak("Raadha")
    print("my friends call me", "Raadha")

elif 'how are you' in query:
    speak("i am fine, thank you")
    speak("how are you, sir")

elif 'fine' in query or 'good' in query:
    speak("it's good to know that you are fine")
```

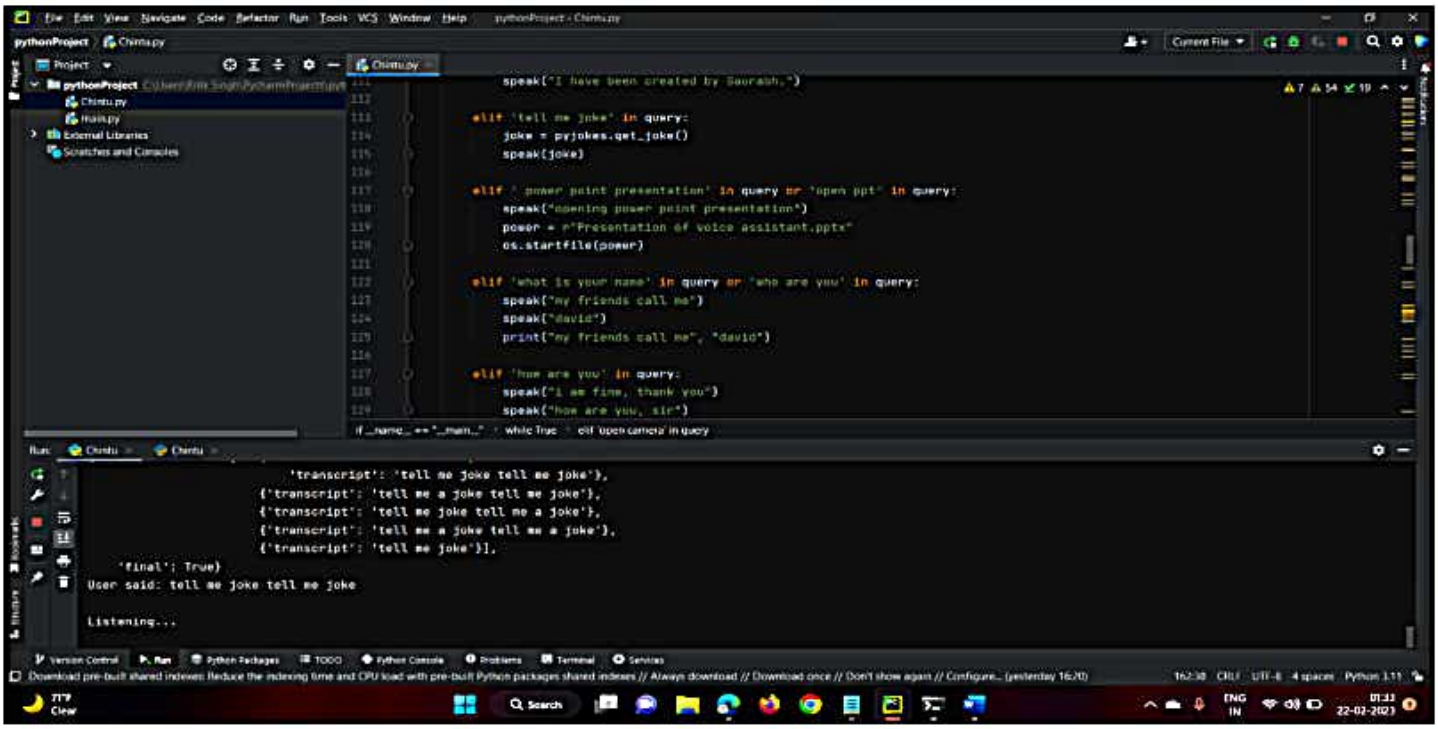
```
#to find location
elif 'where i am' in query or "where we are" in query:
    speak("wait sir, let me check")
    try:
        ipadd = requests.get('https://api.ipify.org').text
        print(ipadd)
        url = 'https://www.google.com/maps/@25.8548335,82.8748292,15z'+ipadd+'.json'
        geo_requests = requests.get(url)
        geo_data = geo_requests.json()
        print(geo_data)
        city = geo_data['city']
        #state = geo_data['state']
        country = geo_data['country']
        speak(f"sir i am not sure, but i think we are in {city} city of {country} country")
    except Exception as e:
        speak("sorry sir due to network issue i am not able to find where we are.")
        pass

elif "shut down the system" in query:
    os.system("shutdown /r /t s")

elif "restart the system" in query:
    os.system("shutdown /r /t s")

elif "sleep the system" in query:
    os.system("rundll32.exe powrprof.dll,SetSuspendState 0,1,0")

elif 'open camera' in query:
    cap = cv2.VideoCapture(0)
    while True:
        ret, img = cap.read()
        cv2.imshow('webcam',img)
        k = cv2.waitKey(50)
```

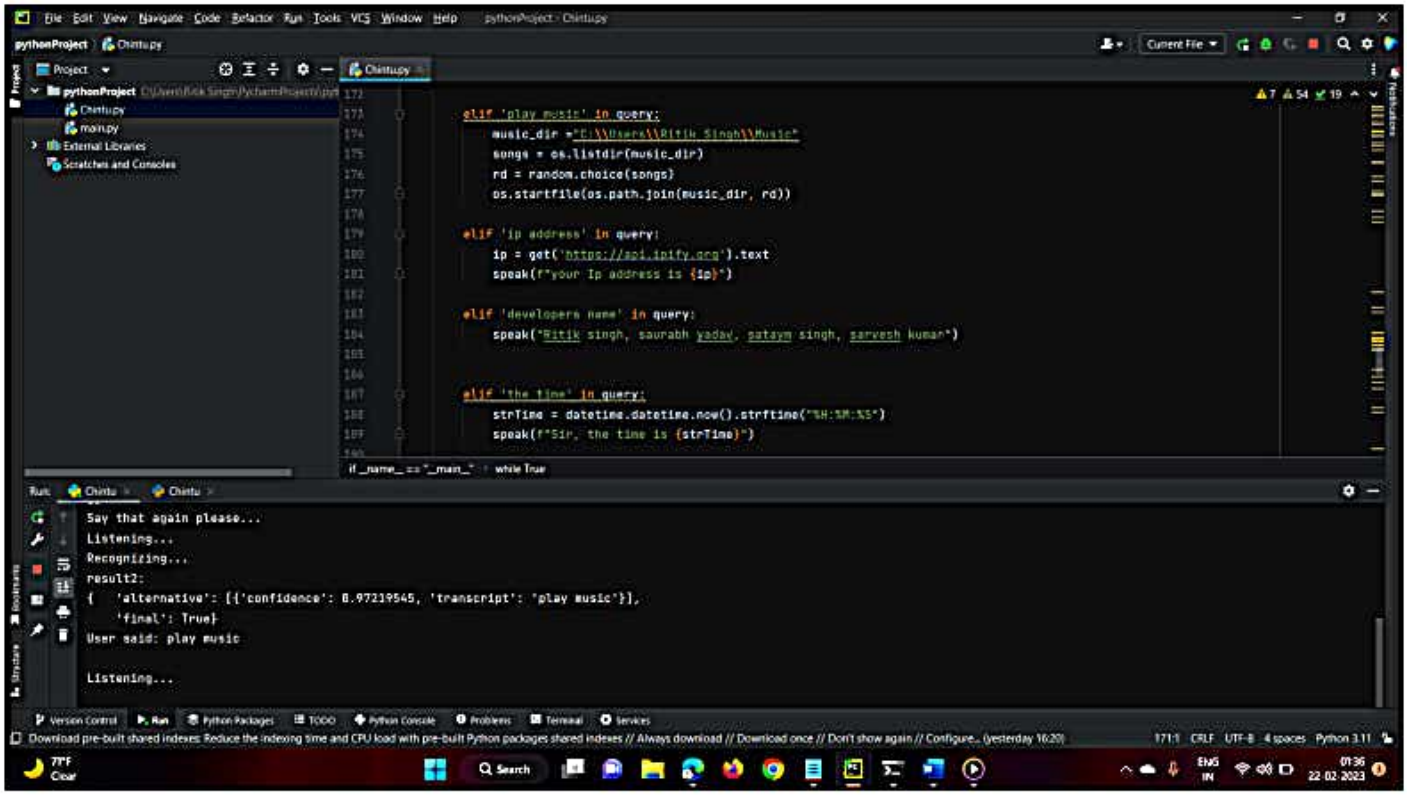
```
            if k==27:
                break;
        cap.release()
        cv2.destroyAllWindows()


    elif 'play music' in query:
        music_dir ="C:\\Users\\Ritik Singh\\Music"
        songs = os.listdir(music_dir)
        rd = random.choice(songs)
        os.startfile(os.path.join(music_dir, rd))

    elif 'ip address' in query:
        ip = get('https://api.ipify.org').text
        speak(f"your Ip address is {ip}")

    elif 'developers name' in query:
        speak("Ritik singh, saurabh yadav, sataym singh, sarvesh kumar")


    elif 'the time' in query:
        strTime = datetime.datetime.now().strftime("%H:%M:%S")
        speak(f"Sir, the time is {strTime}")

    elif 'open notepad' in query:
        npath = "C:\\Program
Files\\WindowsApps\\Microsoft.WindowsNotepad_11.2210.5.0_x64__8wekyb3d8bbwe\\Notepad\\Notepad.exe"
        os.startfile(npath)

    elif 'open code' in query:
        codePath ="C:\\Users\\Ritik Singh\\PycharmProjects\\pythonProject\\Chintu.py "
        os.startfile(codePath)
```

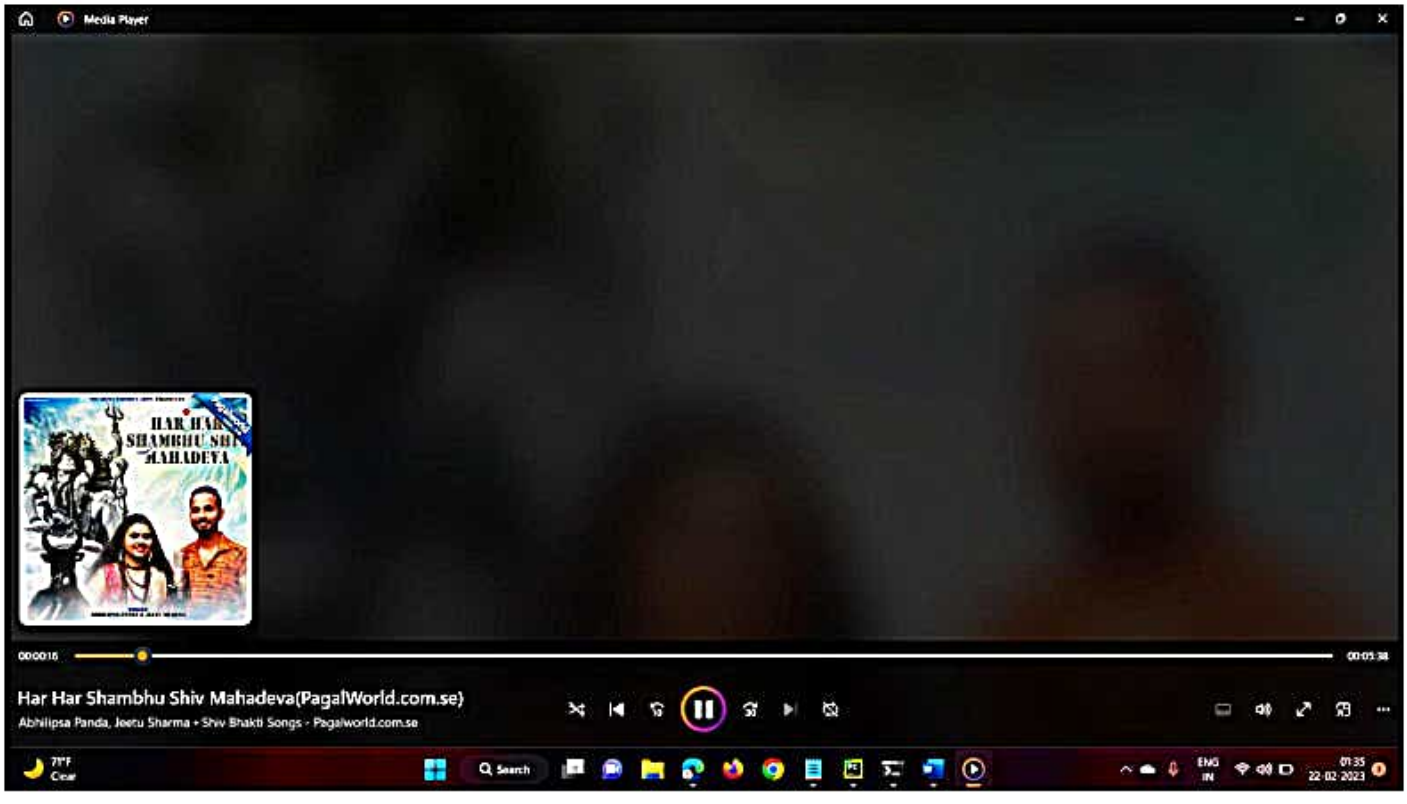
```
elif 'tell me news' in query:
    speak("please wait sir fetching the letest news")
    news()

elif 'wikipedia' in query:
    speak('Searching Wikipedia...')
    query = query.replace("wikipedia", "")
    results = wikipedia.summary(query, sentences=2)
    speak("According to Wikipedia")
    print(results)
    speak(results)


elif 'email to ritik' in query:
    try:
        speak("What should I say?")
        content = takeCommand().lower( )
        to = "cse21160@gbitm.ac.in"
        sendEmail(to, content)
        speak("Email has been sent!")
    except Exception as e:
        print(e)
        speak("Sorry my friend. I am not able to send this email")

elif'send a email' in query:
    try:
        speak("what should i say?")
        content = takeCommand()
        speak("whome should i send")
        to = input()
        sendEmail(to, content)

        speak("email has been sent!")
    except Exception as e:
```

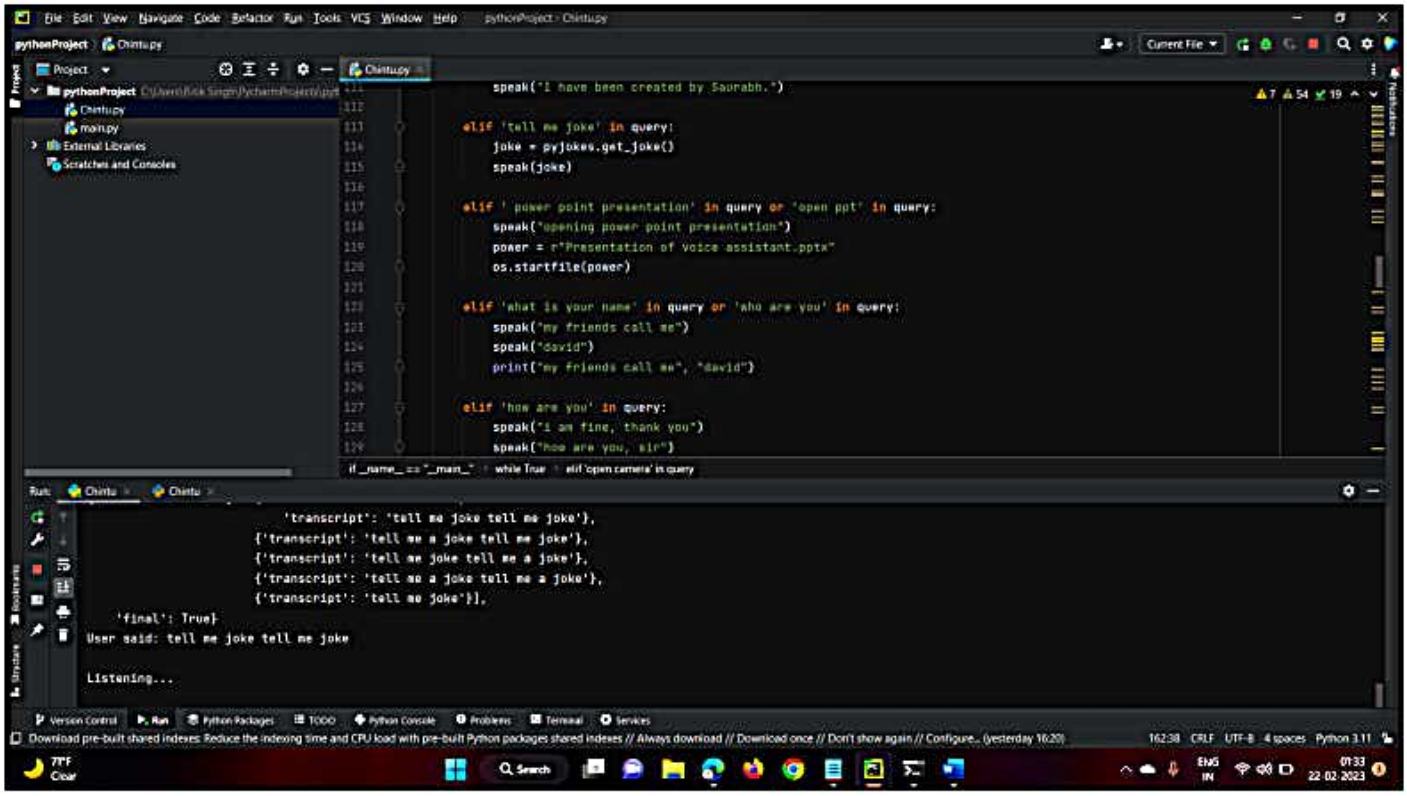
```
        print(e)
        speak("I am not able to send this email")


elif'no thanks' in query or 'exit' in query:
    speak("thanks for using me sir, have a good day.")
    sys.exit()


speak("sir do you have any other work")
```

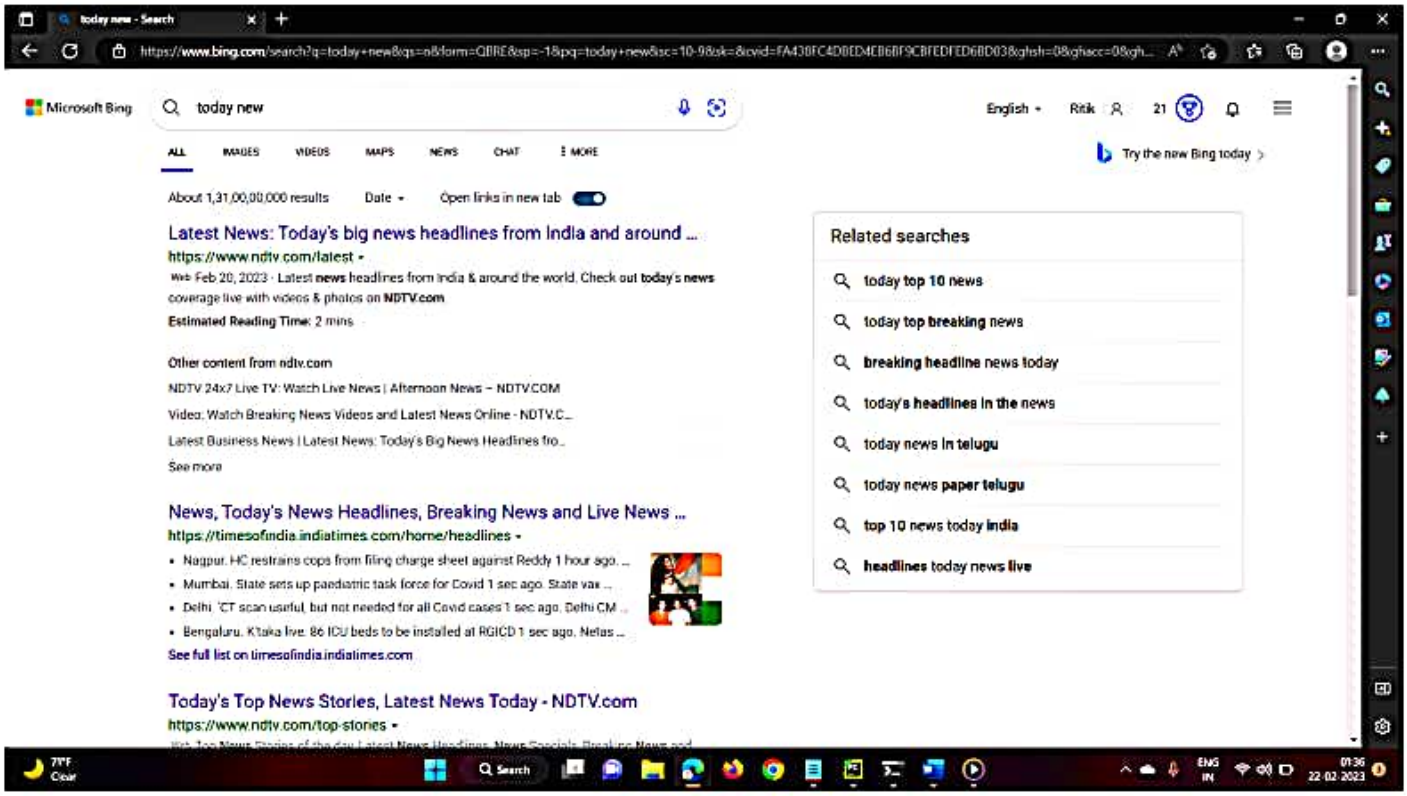

if hour>=0 and hour<12:
speak("Good Morning!")
elif hour>=12 and hour<18:
speak("Good Afternoon!")
else:
speak("Good Evening!")
speak("i am Raadha , please tell me how may i help you")
#to convert voice into text
def takeCommand():
r = sr.Recognizer()
with sr.Microphone() as source:
print("Listening...")
r.pause_threshold = 1
audio = r.listen(source)
wishMe()
"C:\Program Files\Python311\python.exe" "C:\Users\Ritik Singh\PycharmProjects\pythonProject\Chintu.py"
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\MSTTS_V110_enIN_NeerajM
Listening...
Recognizing...
result2:
{ 'alternative': [{'confidence': 0.97219545, 'transcript': 'open YouTube'},
{'transcript': 'Savannah open YouTube'},
{'transcript': 'Hevana open YouTube'},
{'transcript': 'Devon open YouTube'},
{'transcript': 'open open YouTube'}],

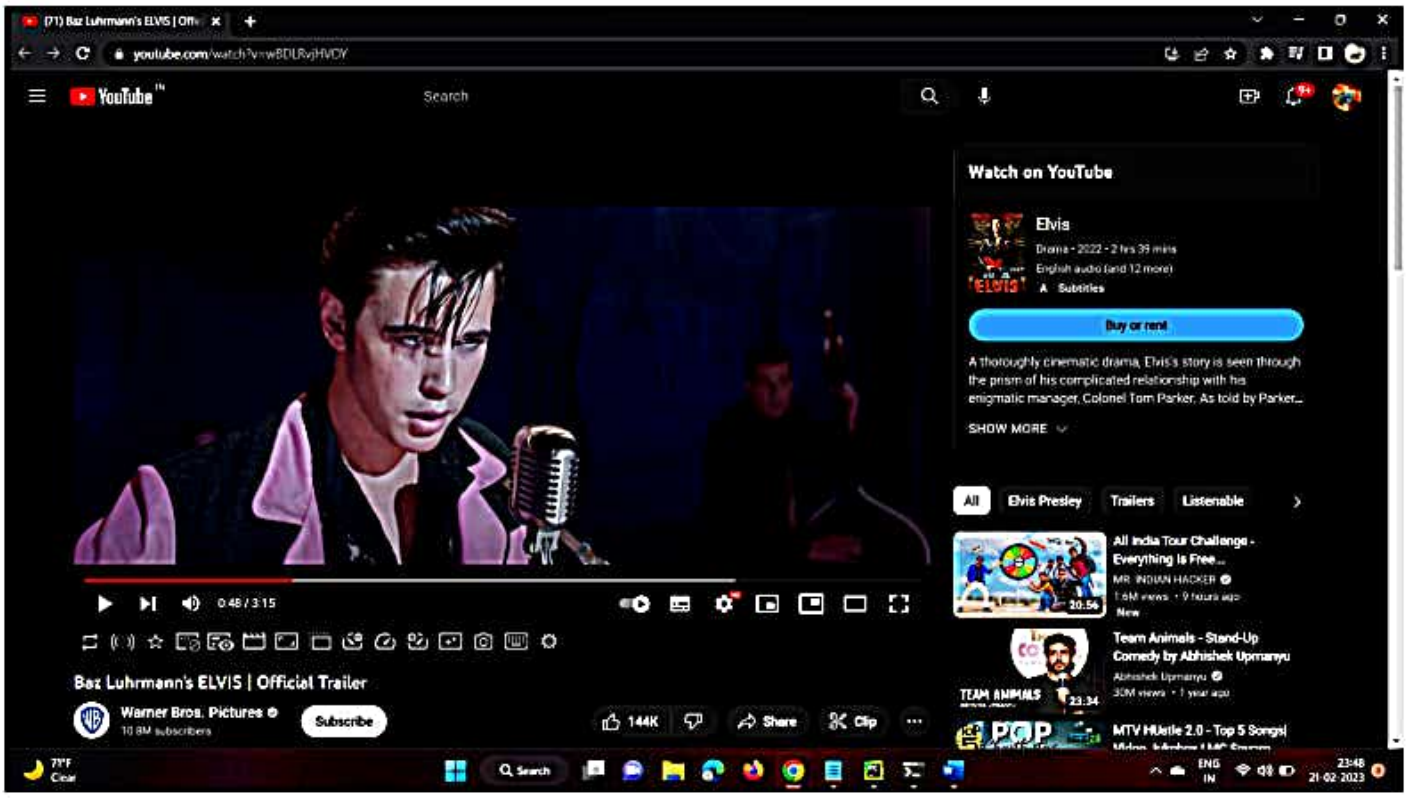

speak("I have been created by Saurabh.")
elif 'tell me joke' in query:
joke = pyjokes.get_joke()
speak(joke)
elif ' power point presentation' in query or 'open ppt' in query:
speak("opening power point presentation")
power = r"Presentation of voice assistant.pptx"
os.startfile(power)
elif 'what is your name' in query or 'who are you' in query:
speak("my friends call me")
speak("david")
print("my friends call me", "david")
elif 'how are you' in query:
speak("i am fine, thank you")
speak("how are you, sir")
'transcript': 'tell me joke tell me joke'},
{'transcript': 'tell me a joke tell me joke'},
{'transcript': 'tell me joke tell me a joke'},
{'transcript': 'tell me a joke tell me a joke'},
{'transcript': 'tell me joke'}],
'final': True}
User said: tell me joke tell me joke
Listening...

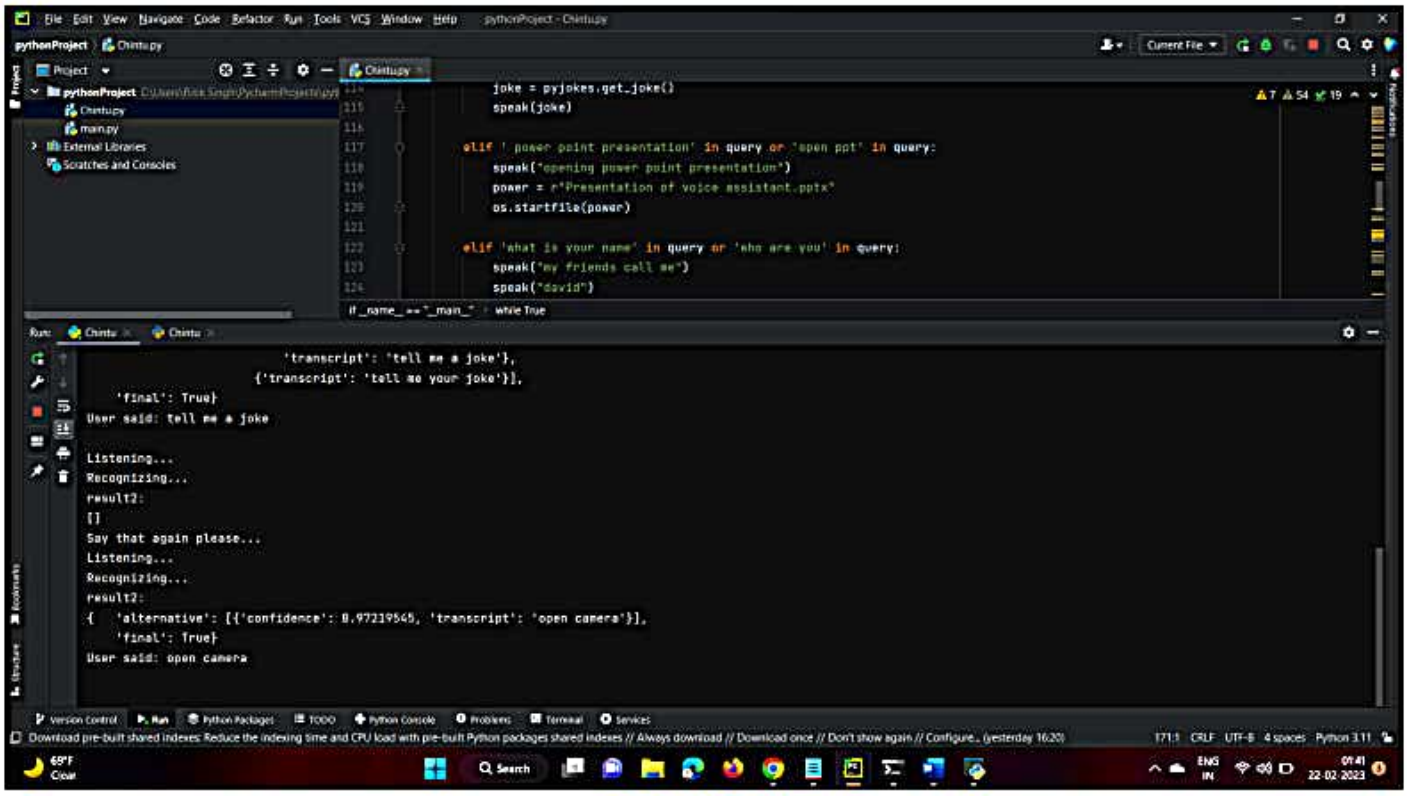

pythonProject - Chintupy
Chintupy
main.py
External Libraries
Scratches and Consoles
elif 'play music' in query:
music_dir ="C:\\Users\\Ritik Singh\\Music"
songs = os.listdir(music_dir)
rd = random.choice(songs)
os.startfile(os.path.join(music_dir, rd))
elif 'ip address' in query:
ip = get('https://api.ipify.org').text
speak(f"your Ip address is {ip}")
elif 'developers name' in query:
speak("Ritik singh, saurabh yadav, satyam singh, sarvesh kumar")
elif 'the time' in query:
strTime = datetime.datetime.now().strftime("%H:%M:%S")
speak(f"Sir, the time is {strTime}")
if __name__ == "__main__" : while True
Say that again please...
Listening...
Recognizing...
result2:
{ 'alternative': [{'confidence': 0.97219545, 'transcript': 'play music'}],
'final': True}
User said: play music
Listening...


Media Player
HAR HAR SHAMBHU SHIV MAHADEVA
Har Har Shambhu Shiv Mahadeva(PagalWorld.com.se)
Abhilipsa Panda, Jeetu Sharma - Shiv Bhakti Songs - Pagalworld.com.se


speak("I have been created by Saurabh.")
elif 'tell me joke' in query:
joke = pyjokes.get_joke()
speak(joke)
elif ' power point presentation' in query or 'open ppt' in query:
speak("opening power point presentation")
power = r"Presentation of voice assistant.pptx"
os.startfile(power)
elif 'what is your name' in query or 'who are you' in query:
speak("my friends call me")
speak("david")
print("my friends call me", "david")
elif 'how are you' in query:
speak("i am fine, thank you")
speak("how are you, sir")
'transcript': 'tell me joke tell me joke'},
{'transcript': 'tell me a joke tell me joke'},
{'transcript': 'tell me joke tell me a joke'},
{'transcript': 'tell me a joke tell me a joke'},
{'transcript': 'tell me joke'}],
'final': True}
User said: tell me joke tell me joke
Listening...


today new - Search
https://www.bing.com/search?q=today+new&qs=n&form=QBRE&sp=-1&pq=today+new&sc=10-9&sk=&cvid=FA430FC4D0ED4EB6BF9CBFEDFED6BD03&ghsh=0&ghacc=0&gh...
Microsoft Bing
today new
English
Ritik
21
ALL
IMAGES
VIDEOS
MAPS
NEWS
CHAT
MORE
Try the new Bing today
About 1,31,00,00,000 results
Date
Open links in new tab
Latest News: Today's big news headlines from India and around ...
https://www.ndtv.com/latest
Web Feb 20, 2023 · Latest news headlines from India & around the world. Check out today's news coverage live with videos & photos on NDTV.com
Estimated Reading Time: 2 mins
Other content from ndtv.com
NDTV 24x7 Live TV: Watch Live News | Afternoon News – NDTV.COM
Video: Watch Breaking News Videos and Latest News Online - NDTV.C...
Latest Business News | Latest News: Today's Big News Headlines fro...
See more
News, Today's News Headlines, Breaking News and Live News ...
https://timesofindia.indiatimes.com/home/headlines
Nagpur: HC restrains cops from filing charge sheet against Reddy 1 hour ago. ...
Mumbai: State sets up paediatric task force for Covid 1 sec ago. State vax ...
Delhi: 'CT scan useful, but not needed for all Covid cases 1 sec ago. Delhi CM ...
Bengaluru: K'taka live: 86 ICU beds to be installed at RGICD 1 sec ago. Netas ...
See full list on timesofindia.indiatimes.com
Today's Top News Stories, Latest News Today - NDTV.com
https://www.ndtv.com/top-stories
Related searches
today top 10 news
today top breaking news
breaking headline news today
today's headlines in the news
today news in telugu
today news paper telugu
top 10 news today india
headlines today news live


youtube.com/watch?v=wBDLRvjHVDY
YouTube
Search
Watch on YouTube
Elvis
Drama · 2022 · 2 hrs 39 mins
English audio (and 12 more)
Subtitles
Buy or rent
A thoroughly cinematic drama, Elvis's story is seen through the prism of his complicated relationship with his enigmatic manager, Colonel Tom Parker. As told by Parker...
SHOW MORE
All
Elvis Presley
Trailers
Listenable
All India Tour Challenge - Everything is Free...
MR INDIAN HACKER
New
Team Animals - Stand-Up Comedy by Abhishek Upmanyu
Abhishek Upmanyu
30M views · 1 year ago
MTV HUstle 2.0 - Top 5 Songs|
0:48 / 3:15
Baz Luhrmann's ELVIS | Official Trailer
Warner Bros. Pictures
10.8M subscribers
Subscribe
144K
Share
Clip
79°F
Clear
Search
ENG
IN
23:48
21-02-2023


File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject - Chintupy
pythonProject
Chintupy
Project
pythonProject
Chintupy
main.py
External Libraries
Scratches and Consoles
joke = pyjokes.get_joke()
speak(joke)
elif ' power point presentation' in query or 'open ppt' in query:
speak("opening power point presentation")
power = r"Presentation of voice assistant.pptx"
os.startfile(power)
elif 'what is your name' in query or 'who are you' in query:
speak("my friends call me")
speak("david")
if __name__ == "__main__"
while True
'transcript': 'tell me a joke'},
{'transcript': 'tell me your joke'}],
'final': True}
User said: tell me a joke
Listening...
Recognizing...
result2:
[]
Say that again please...
Listening...
Recognizing...
result2:
{ 'alternative': [{'confidence': 0.97219565, 'transcript': 'open camera'}],
'final': True}
User said: open camera
Version Control
Run
Python Packages
TODO
Python Console
Problems
Terminal
Services
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (yesterday 16:20)
171:1
CRLF
UTF-8
4 spaces
Python 3.11
69°F
Clear
Search
ENG
IN
09:41
22-02-2023

CONCLUSION

In conclusion, building a voice assistant using Python is a great way to leverage the power of machine learning and natural language processing to create a conversational user interface. With libraries like Speech Recognition and pyttsx3, you can easily recognize speech, convert it to text, and generate a voice output, making it possible to create a voice assistant that can interact with users in a natural and intuitive way.

However, building a fully functional voice assistant requires a lot of work and expertise, as you need to design the user interface, process the user's spoken commands, and integrate with various APIs to retrieve information and perform tasks. There are many advanced voice assistants available today like Siri, Google Assistant, and Alexa, and these use sophisticated machine learning models and have access to a wide range of APIs and services to provide users with a seamless experience. Nonetheless, with Python and a few libraries, you can create your own basic voice assistant and customize it to suit your needs.

BIBLOGRAPHY

Here are some sources that you may find useful for learning more about building voice assistants using Python:

Speech Recognition documentation: https://pypi.org/project/SpeechRecognition/

pyttsx3 documentation: https://pypi.org/project/pyttsx3/

How to Build You Own AI Assistant Using Python: https://www.digitalocean.com/community/tutorials/how-to-build-your-own-ai-assistant-using-python

Creating a Personal Voice Assistant in Python: https://www.geeksforgeeks.org/personal-voice-assistant-in-python/

Building a Voice Assistant with Python: https://www.twilio.com/blog/building-a-voice-assistant-with-pytho

These resources cover a variety of topics, from basic voice recognition and text-to-speech conversion to mo advanced techniques like natural language processing and API integration. They can help you get started w building your own voice assistant and provide guidance on best practices for developing and refining y application.