TALKING FRIDAY

BY
PRIYANSH

Important modules: -

```
import speech_recognition as sr
import webbrowser
import pyttsx3
import musicLibrary
import requests
```

→ To Take Voice Input

→ to open broowser with python

→ To convert text → Voice

→ Self made module to store favourite music inside a dictionary

Important module for fetching data from internet in our case from news API

```
engine = pyttsx3.init()
newsapi = "Your_apihere_"
voices = engine.getProperty('voices')
engine.setProperty('voice',voices[1].id)
```

→ to convert text to speech an object to the class pyttsx3 to use its functions

→ to use news_api

→ [voice1, voice2, voice3]

→ to fetch available voices and assigning voice to our Assistant

```
def speak(text):
    engine.say(text)
    engine.runAndWait()
```

→ A string

→ function created for our convinience to make agent talk whatever we pass to it...

A string

```python
def processCommand(command):
    print(command)
    if("open google"in command.lower()):
        speak(f"opening {command.split(" ")[1]}")
        webbrowser.open("https://google.com")
    elif("open facebook" in command.lower()):
        speak(f"opening {command.split(" ")[1]}")
        webbrowser.open("https://facebook.com")
    elif("open chat gpt" in command.lower()):
        speak(f"opening {command.split(" ")[1]}")
        webbrowser.open("https://chatgpt.com/")
    elif("open github"  in command.lower()):
        speak(f"opening {command.split(" ")[1]}")
        webbrowser.open("https://github.com")
    elif("open linkedin"in command.lower()):
        speak(f"opening {command.split(" ")[1]}")
        webbrowser.open("https://linkedin.com")
    elif("open youtube"in command.lower()):
        speak(f"opening {command.split(" ")[1]}")
        webbrowser.open("https://youtube.com")
    elif("open portfolio"in command.lower()):
        speak(f"opening your {command.split(" ")[1]}")
        webbrowser.open_new_tab("https://helpful-elf-967c0f.netlify.app/")
    elif command.lower().startswith("play"):
        try:
            song = command.lower().split(" ")[1]
            link = musicLibrary.music[song]
            webbrowser.open(link)
        except KeyError:
            speak("Sorry sir,I Dont have this song in my library")
    elif "news" in command.lower():
        r =  requests.get(f"https://newsapi.org/v2/top-headlines?country=us&apiKey={newsapi}")
        if r.status_code == 200:
            data = r.json()
            articles = data.get('articles',[])
            for article in articles:
                speak(article['title'])
```

Command "Open Linked_in" → (" ") → ["Open", "Linkedin"]

to retrive element - At index = 1 of the list

Split by spaces

Used as best practice to check if working within same program or working with some other program

```python
if __name__ == "__main__":
    speak("INITIALIZING FRIDAY")
    #LISTENING FOR THE WAKE WORD JARVIS
    while True:
      # Microphone se input lena
        r = sr.Recognizer()
        print("listening...")
        try:
            with sr.Microphone() as source:
                # Google Speech Recognition API se text me convert karo
                print("processing...")
                audio = r.listen(source,timeout=2,phrase_time_limit=1)
                command = r.recognize_google(audio, language='en-US')
                if(command.lower() == "friday"):
                    speak("YA")
                    print("Give Command")
                    with sr.Microphone() as source:
                        audio = r.listen(source,timeout = 5,phrase_time_limit = 2)
                        command = r.recognize_google(audio,language = 'en-US')
                        processCommand(command)

        except Exception as e:
            print("Error; {0}".format(e))
```

Command At Very Beginning...

Infinite loop → making program never ending

object of Recognizer() class from module SR

for opening mic

for listening Audio

for Converting Audio to Text using Google_API...

Wake Statement

Mic open this time to take command

Listen Audio command

converting to text using google

PASSED TO process Command!!!

Used Try and Except to handle errors

And overcoming chances of exiting program because of errors...

To HANDLE Error we use except Block

```
except sr.UnknownValueError:
    print("Samajh nahi aaya, thoda clearly bolo.")
except sr.WaitTimeoutError:
    print("Timeout: Tumne kuch bola hi nahi.")
except sr.RequestError as e:
    print(f"Request error: {e}")
```

And Also it makes Sure program run smoothly...

```
def close_process_with_grace(process_name, wait_seconds=5):
    # Step 1: Gracefully close (taskkill without /F)
    subprocess.run(["taskkill", "/IM", process_name], shell=True)
    print(f"Sent close signal to {process_name}, waiting {wait_seconds} seconds...")

    # Step 2: Wait for process to close
    time.sleep(wait_seconds)

    # Step 3: Force kill (taskkill with /F)
    # This will ensure process is terminated if still running
    subprocess.run(["taskkill", "/F", "/IM", process_name], shell=True)
    print(f"Force kill command sent to {process_name} (if still running).")
```

To smoothly Shut down In_window Application

```
import subprocess
import time
```

Module necessary to open And close files...

To Add delay for smooth Shutting Off of the Softwares like whatsApp And Browser in our case...

Adding Brain to our Assistant:

we would integrate A LLM using OLLAMA...

→Install OLLAMA to host LLM in your local machine

Copy Edit

```python
def ask_local_llm(prompt):
    # Localhost par chal raha LLM API ka endpoint
    url = "http://localhost:11434/api/generate"

    # Request ke liye data prepare kar rahe hain
    data = {
        "model": "llama3.2:latest",  # Use karne wala model ka naam (yahan 'llama3.2')
        "prompt": prompt,            # User ka input ya sawal jo LLM ko dena hai
        "stream": False              # stream=False ka matlab hai pura response ek hi baar mein m
    }

    try:
        # POST request bhejna local LLM API par
        response = requests.post(url, json=data)

        # Agar request successful ho (HTTP 200 status code)
        if response.status_code == 200:
            # JSON response se "response" field nikaal kar strip (whitespace hata) kar return kar
            return response.json()["response"].strip()
        else:
            # Agar model se response na aaye to error message return karo
            return "Sorry bhai, model se response nahi aaya."
```

→ to make sure Question is asked

```
question = ["what","how","when","where","tell me","can you","who"]
```

→ from handle commands function

```
else:
  for word in question:
    if word in command:
        speak("Processing with my brain sir please wait.")
        answer = ask_local_llm(command)
        speak(clear_response(answer))
        speak("next question sir")
```

3→ Read forloop and for each word if word from Question list envoke LLM

↳ Now we would Add this else statement to handle Questions asked to our Assistant...

⇒ MAKE SURE TO RUN THAT LLM Model Open command prompt and ENTER

```
>ollama run llama3.2:latest
age (/? for help)
```