INNOVATIVE ASSIGNMENT

Programming for Scientific Computing-2CS404

VOICE SEARCH

GROUP MEMBERS:

21BCE123: KHUSHI SONI

21BCE130: KRUNALI SHAH

INTRODUCTION:

Voice search has gained popularity as a convenient and hands-free way to search the internet. Rather than typing search queries into a search engine, users can now simply ask digital assistants like Amazon Alexa, Google Home, or Apple Siri to find information on their behalf.

Wikipedia, the world's largest online encyclopedia, is a popular source of information for many voice search queries. This is because Wikipedia contains a vast amount of information on a wide range of topics and is constantly being updated and maintained by a community of volunteer editors.

PACKAGES REQUIRED IN OUR PROJECT:

1) speech_recognition
   Command used: $ pip install speech_recognition
2) pyttsx3
   Command used: $ pip install pyttsx3
3) os
   This package was pre-installed
4) wikipedia
   Command used: $ pip install Wikipedia
5) webbrowser
   This package was pre-installed
6) random
   This package was pre-installed
7) datetime
   This library was pre-installed
8) sys
   This package was pre-installed

FEATURES:

Speech recognition python library SpeechRecognition is used to convert the user's spoken query into text that can be used to search Wikipedia.

Wikipedia API: Python's Wikipedia API can be used to retrieve relevant articles and information from the vast database of Wikipedia based on the user's query.

pyttsx3 is a Python library for text-to-speech conversion. It provides a simple way to produce speech with a variety of voices and languages. The library is built on top of the Microsoft Speech API (SAPI), which allows developers to integrate speech into their applications. Some common use cases for pyttsx3 include creating audio alerts, reading text aloud, and building interactive voice-enabled applications.

The webbrowser module in Python provides a high-level interface to allow displaying web-based documents to users. It allows opening a URL in a browser window, determining the default browser, and retrieving information about installed web browsers.

The random module in Python provides functionalities for generating random numbers, shuffling sequences randomly, and selecting random items from a sequence, among other things. random.choice(seq): Returns a randomly chosen element from a sequence (string, list, tuple, etc.).

The datetime module in Python supplies classes for working with dates and times. It provides a number of types to deal with dates, times, and time intervals.In this code, the datetime module is being used to get the current hour of the day so that the assistant can greet the user according to the time of the day.

The os module is a built-in Python library that provides a way to interact with the operating system. It provides a wide range of functions for performing various operating system-related tasks, such as creating, deleting, or renaming files and directories, launching external programs, and managing environment variables.

The sys module in Python provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. It provides a way to manipulate Python runtime environment, as well as tools for terminating programs abruptly, and for handling command-line arguments. In the code you provided, sys module is used to exit the program using sys.exit() function.

Here is how the code works:

This code is a simple digital assistant program that can perform certain actions based on voice commands. It makes use of various Python libraries such as pyttsx3, webbrowser, random, wikipedia, datetime, os, and speech_recognition.

The program starts by initializing the pyttsx3 library to create an object of the Text-to-speech engine. It also sets the voice property of the engine to a female voice.

Next, the program defines a few functions.

The speak() function takes an input string and outputs it as speech using the initialized engine. It also prints the same string to the console.

The greetMe() function uses the datetime library to get the current time and greets the user accordingly using the speak() function.

The myCommand() function uses the speech_recognition library to listen to the user's voice command through the microphone. It sets the threshold for silence to 1 second and uses Google's speech recognition service to recognize the spoken words. If the speech is not recognized, the function calls itself recursively.

The main part of the program runs in an infinite loop and listens for the user's voice command using the myCommand() function.

If the user says "open youtube", "open google", or "open gmail", the program opens the respective website using the webbrowser library.

If the user says "what's up" or "how are you", the program randomly selects and speaks one of the pre-defined response messages.

If the user says "nothing", "abort", or "stop", the program terminates by saying goodbye.

If the user says "hello", the program greets the user.

If the user says "bye", the program terminates by saying goodbye.

If the user says "play music", the program selects a random song from a folder and plays it using the os library.

If the user says anything else, the program searches for the topic on Wikipedia and reads the summary using the wikipedia library. If it cannot find anything on Wikipedia, it opens Google in the default web browser. Finally, it prompts the user for the next command.

The program is a simple example of how voice recognition can be used to create a basic digital assistant that performs a few actions.

INPUT:

```
import pyttsx3
import webbrowser
import random
import wikipedia
import datetime
import os
import sys
import speech_recognition as sr
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice',  voices[1].id)
def speak(audio):
   print('Computer: ' + audio)
   engine.say(audio)
   engine.runAndWait()
def greetMe():
   currentH = int(datetime.datetime.now().hour)
   if currentH >= 0 and currentH < 12:
      speak('Good Morning!')
   if currentH >= 12 and currentH < 18:
      speak('Good Afternoon!')
```

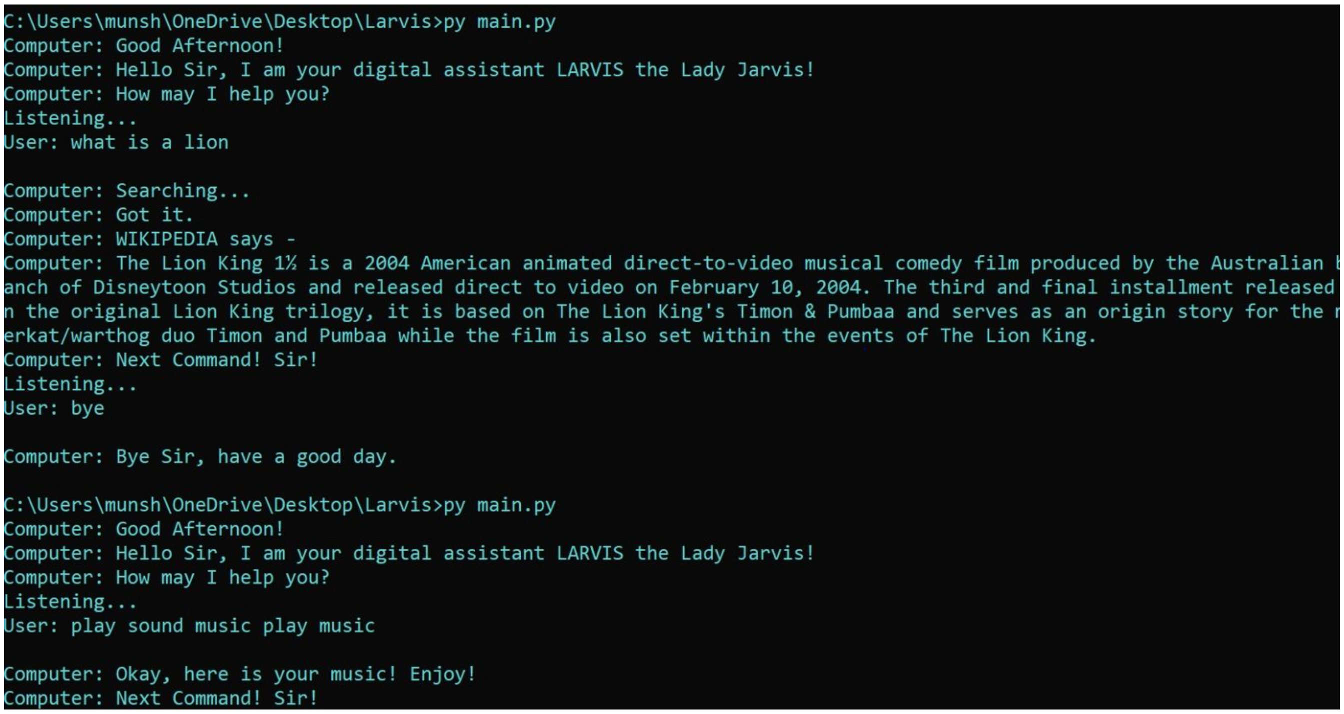
```
    if currentH >= 18 and currentH != 0:
        speak('Good Evening!')
greetMe()
speak('Hello Sir, I am your digital assistant LARVIS the Lady Jarvis!')
speak('How may I help you?')
def myCommand():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        r.pause_threshold = 1
        audio = r.listen(source)
    try:
        query = r.recognize_google(audio, language='en-in')
        print('User: ' + query + '\n')
    except sr.UnknownValueError:
        speak('Sorry sir! I didn\'t get that! Speak Again!')
        query = myCommand()
    return query
if __name__ == '__main__':
    while True:
        query = myCommand();
        query = query.lower()
        if 'open youtube' in query:
            speak('okay')
            webbrowser.open('www.youtube.com')
        elif 'open google' in query:
            speak('okay')
            webbrowser.open('www.google.co.in')
        elif 'open gmail' in query:
            speak('okay')
            webbrowser.open('www.gmail.com')
        elif 'play prayer' in query:

os.system("C:/Users/munsh/OneDrive/Desktop/Larvis/prayer/prayer1.mp3")
        elif "what\'s up" in query or 'how are you' in query:
            stMsgs = ['Just doing my thing!', 'I am fine!', 'Nice!', 'I am nice and
full of energy']
            speak(random.choice(stMsgs))
        elif 'nothing' in query or 'abort' in query or 'stop' in query:
            speak('okay')
            speak('Bye Sir, have a good day.')
            sys.exit()
        elif 'hello' in query:
            speak('Hello Sir')
        elif 'bye' in query:
            speak('Bye Sir, have a good day.')
            sys.exit()
        elif 'what is your name' in query:
            speak('My name is Larvis!')
        elif 'play music' in query:
            music_folder = 'C:/Users/munsh/OneDrive/Desktop/Larvis/songs'+'\\'
            music = ['music1', 'music2', 'music3']
            random_music = music_folder + random.choice(music) + '.mp3'
            os.system(random_music)
            speak('Okay, here is your music! Enjoy!')
        else:
            query = query
            speak('Searching...')
            try:
                results = wikipedia.summary(query, sentences=2)
                speak('Got it.')
                speak('WIKIPEDIA says - ')
                speak(results)
            except:
                speak("Can't find anything relaed to your topic you can search it
by yourself!")
```

```
            webbrowser.open('www.google.com')

        speak('Next Command! Sir!')
```

OUTPUT:



```
C:\Users\munsh\OneDrive\Desktop\Larvis>py main.py
Computer: Good Afternoon!
Computer: Hello Sir, I am your digital assistant LARVIS the Lady Jarvis!
Computer: How may I help you?
Listening...
User: what is a lion

Computer: Searching...
Computer: Got it.
Computer: WIKIPEDIA says -
Computer: The Lion King 1½ is a 2004 American animated direct-to-video musical comedy film produced by the Australian b
anch of Disneytoon Studios and released direct to video on February 10, 2004. The third and final installment released
n the original Lion King trilogy, it is based on The Lion King's Timon & Pumbaa and serves as an origin story for the m
erkat/warthog duo Timon and Pumbaa while the film is also set within the events of The Lion King.
Computer: Next Command! Sir!
Listening...
User: bye

Computer: Bye Sir, have a good day.

C:\Users\munsh\OneDrive\Desktop\Larvis>py main.py
Computer: Good Afternoon!
Computer: Hello Sir, I am your digital assistant LARVIS the Lady Jarvis!
Computer: How may I help you?
Listening...
User: play sound music play music

Computer: Okay, here is your music! Enjoy!
Computer: Next Command! Sir!
```