VOICE ASSISTANT USING PYTHON

BASICS

```
1) In  this we uses the SpeechRecognition library to recognize speech and the pyttsx3 library to.
   convert text to speech.
```

In [1]:
```
pip install SpeechRecognition
```

```
Collecting SpeechRecognition
  Downloading SpeechRecognition-3.10.0-py2.py3-none-any.whl (32.8 MB)
Requirement already satisfied: requests>=2.26.0 in c:\programdata\anaconda3\lib\site-packages (from SpeechRe
cognition) (2.26.0)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.
26.0->SpeechRecognition) (3.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from reques
ts>=2.26.0->SpeechRecognition) (2021.10.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\programdata\anaconda3\lib\site-packages (from
requests>=2.26.0->SpeechRecognition) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from req
uests>=2.26.0->SpeechRecognition) (1.26.7)
Installing collected packages: SpeechRecognition
Successfully installed SpeechRecognition-3.10.0
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
pip install pyttsx3
```

```
Requirement already satisfied: pyttsx3 in c:\programdata\anaconda3\lib\site-packages (2.90)
Requirement already satisfied: comtypes in c:\programdata\anaconda3\lib\site-packages (from pyttsx3) (1.1.1
0)
Requirement already satisfied: pywin32 in c:\programdata\anaconda3\lib\site-packages (from pyttsx3) (228)
Requirement already satisfied: pypiwin32 in c:\programdata\anaconda3\lib\site-packages (from pyttsx3) (223)
Note: you may need to restart the kernel to use updated packages.
```

In [14]:
```
pip install pyaudio
```

```
Requirement already satisfied: pyaudio in c:\programdata\anaconda3\lib\site-packages (0.2.13)
Note: you may need to restart the kernel to use updated packages.
```

In [6]:
```
import speech_recognition as sr
import pyttsx3
```

```
2) It initializes the speech recognizer and the text-to-speech engine.
```

In [7]:
```
recognizer = sr.Recognizer()
```

```
3) It initializes the speech recognizer and the text-to-speech engine.
```

In [8]:
```
engine = pyttsx3.init()
```

```
4) There is a function called speak which takes text as input and converts it to speech using the
   text-to-speech engine.
```

In [9]:
```
def speak(text):
    engine.say(text)
    engine.runAndWait()
```

```
5) There is a function called listen which listens for voice input from the microphone. It adjusts for
ambient noise,
   captures the audio, and uses Google's speech recognition service to convert the audio into text.
```

In [15]:
```python
def listen():
    with sr.Microphone() as source:
        print("Listening...")
        recognizer.adjust_for_ambient_noise(source)
        audio = recognizer.listen(source)

        try:
            print("Recognizing...")
            text = recognizer.recognize_google(audio)
            print("You said:", text)
            return text
        except sr.UnknownValueError:
            print("Sorry, I didn't understand.")
        except sr.RequestError:
            print("Sorry, I'm currently offline.")

    return ""

# Main Loop
while True:
    command = listen().lower()

    if "hello" in command:
        speak("Hello there!")
    elif "goodbye" in command:
        speak("Goodbye!")
        break
    else:
        speak("Sorry, I didn't catch that. Can you please repeat?")
```

```
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
You said: tell me about coffee
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
You said: get up from a window
Listening...
Recognizing...
You said: Mere Ho
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
You said: hello there
Listening...
Recognizing...
You said: Dubai
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
```

```
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
Sorry, I didn't understand.
Listening...
Recognizing...
You said: goodbye
```

BASIC TO ADVANCE

```
1) The code imports the necessary libraries for speech recognition, text-to-speech conversion, and other
functionalities.
```

In [17]:
```
import speech_recognition as sr
import pyttsx3
import datetime
import webbrowser
import os
import random
import smtplib
import requests
import json
```

```
2) It initializes the speech recognizer and the text-to-speech engine.
```

In [18]:
```
recognizer = sr.Recognizer()
```

In [19]:
```
engine = pyttsx3.init()
```

```
There are various functions defined for different actions:
speak(text): Converts text to speech using the text-to-speech engine.
```

listen(): Listens for voice input, converts it to text using the speech recognition library, and returns the recognized text.
get_date_time(): Gets the current date and time and returns them as formatted strings.
open_website(url): Opens a specified website in the default web browser.
play_music(directory): Plays a random song from a specified directory.
send_email(sender_email, sender_password, receiver_email, subject, message): Sends an email with the specified details.
get_weather_forecast(city): Retrieves the weather forecast for a specified city using the OpenWeatherMap API.

3) Function to convert text to speech

```
In [20]: def speak(text):
             engine.say(text)
             engine.runAndWait()
```

4) Function to listen for voice input

```
In [21]: def listen():
             with sr.Microphone() as source:
                 print("Listening...")
                 recognizer.adjust_for_ambient_noise(source)
                 audio = recognizer.listen(source)

                 try:
                     print("Recognizing...")
                     text = recognizer.recognize_google(audio)
                     print("You said:", text)
                     return text
                 except sr.UnknownValueError:
                     print("Sorry, I didn't understand.")
                 except sr.RequestError:
                     print("Sorry, I'm currently offline.")

             return ""
```

5) Function to get the current date and time

In [22]:
```python
def get_date_time():
    now = datetime.datetime.now()
    date = now.strftime("%A, %d %B %Y")
    time = now.strftime("%I:%M %p")
    return date, time
```

6) Function to open a website

In [23]:
```python
def open_website(url):
    webbrowser.open(url)
```

7) Function to play a random song from a directory

In [24]:
```python
def play_music(directory):
    songs = os.listdir(directory)
    if songs:
        song = random.choice(songs)
        os.startfile(os.path.join(directory, song))
    else:
        speak("No songs found in the specified directory.")
```

8) Function to send an email

In [25]:
```python
def send_email(sender_email, sender_password, receiver_email, subject, message):
    try:
        server = smtplib.SMTP("smtp.gmail.com", 587)
        server.starttls()
        server.login(sender_email, sender_password)
        email_message = f"Subject: {subject}\n\n{message}"
        server.sendmail(sender_email, receiver_email, email_message)
        server.quit()
        speak("Email sent successfully.")
    except smtplib.SMTPAuthenticationError:
        speak("Failed to authenticate. Please check your email credentials.")
    except smtplib.SMTPException:
        speak("An error occurred while sending the email.")
```

9) Function to get the weather forecast

In [26]:
```python
def get_weather_forecast(city):
    api_key = "your_api_key"  # Replace with your own API key from OpenWeatherMap
    url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"
    response = requests.get(url)
    data = json.loads(response.text)
    if data["cod"] == "404":
        speak("City not found. Please try again.")
        return
    weather_description = data["weather"][0]["description"]
    temperature = data["main"]["temp"]
    humidity = data["main"]["humidity"]
    wind_speed = data["wind"]["speed"]
    speak(f"The weather in {city} is {weather_description}. The temperature is {temperature} degrees Celsius.
          f"The humidity is {humidity} percent. The wind speed is {wind_speed} meters per second.")
```

10) Main loop

```
The main loop continuously listens for voice input using the listen() function and performs actions based on
the recognized
commands.
If the recognized command contains the word "hello", the assistant responds with "Hello there!" using the
speak() function.
```

```
If the recognized command contains the word "goodbye" or "bye", the assistant responds with "Goodbye!" and
breaks out of the
loop, terminating the program.
If the recognized command contains the word "date", the assistant retrieves the current date using
get_date_time() and responds
with the date.
If the recognized command contains the word "time", the assistant retrieves the current time using
get_date_time() and responds
with the time.
If the recognized command contains the word "open website", the assistant asks for the website name, listens
for it, and opens
the specified website using open_website().
If the recognized command contains the phrase "play music", the assistant plays a random song from the
specified music directory
using play_music().
If the recognized command contains the phrase "send email", the assistant asks for email details, listens for
them, and sends
the email using send_email().
If the recognized command contains the word "weather", the assistant asks for the city name, listens for it,
and retrieves the
weather forecast using get_weather_forecast().
If none of the recognized commands match, the assistant responds with "Sorry, I didn't catch that. Can you
please repeat?"
using the speak() function.
```

In [*]:

```
while True:
    command = listen().lower()

    if "hello" in command:
        speak("Hello there!")
    elif "goodbye" in command or "bye" in command:
        speak("Goodbye!")
        break
    elif "date" in command:
        date, _ = get_date_time()
        speak(f"Today's date is {date}.")
    elif "time" in command:
        _, time = get_date_time()
        speak(f"The current time is {time}.")
    elif "open" in command:
        if "website" in command:
            speak("Which website would you like to open?")
            website = listen().lower()
            open_website("https://" + website)
    elif "play music" in command:
        music_directory = "path_to_music_directory"  # Replace with the path to your music directory
        play_music(music_directory)
    elif "send email" in command:
        speak("Please provide the following details:")
        speak("Sender Email")
        sender_email = listen().lower()
        speak("Sender Password")
        sender_password = listen()
        speak("Receiver Email")
        receiver_email = listen().lower()
        speak("Subject")
        subject = listen()
        speak("Message")
        message = listen()
        send_email(sender_email, sender_password, receiver_email, subject, message)
    elif "weather" in command:
        speak("Which city's weather forecast would you like to know?")
        city = listen().lower()
        get_weather_forecast(city)
    else:
        speak("Sorry, I didn't catch that. Can you please repeat?")
```

In [ ]: