REPORT FOR A.I. (FRAIS) with MyClass Automation

As a project work for Course

PYTHON PROGRAMMING (INT 213)

Name: MD SAQUEB HUSSAIN SIDDIQUE

Registration Number: 12020603

Name: SHAIK MAHAMMED RAFI

Registration Number: 12013757

Program: CSE B. Tech

Semester: Third (3rd)

School: School of Computer Science and Engineering

Date of submission : 20TH NOVEMBER 2021

Lovely Professional University
Jalandhar, Punjab, India.

ACKNOWLEDGEMENT

I would like to thank my mentor prof. Ashish Srivastava and my friends Harshul Sharma, Anupam Verma, Sarvesh Mishra, Rishab Raj, Mosin MD, Happy Sangwan, Govinda Thapa namely etc.. for their advice and inputs on this project. Many thanks to my friends and seniors as well, who spent countless hours to listen and provide feedbacks.

Abstraction

It's cool, to have our own A.I. assistant. It makes easier our some work like sending mail without typing a single word, doing Wikipedia searches without opening web browser, search information on google without typing and performing many other daily tasks like playing music with the help of a single voice command. We make our personal A.I (F.R.A.I.S. means Female Represented Artificial Intelligence System) using Python.

INTRODUCTION

1.1 Context

This project has been done as part of my course for the CSE(computer science and engineering) at Lovely Professional University. Supervised by Ashish Srivastava, I have two months to fulfil the requirements in order to succeed the module.

1.2 Motivations

Being extremely interested in everything having a relation with the Artificial Intelligence, the group project (II) was a great occasion to give us the time to learn and confirm our interest for this field. The fact that we can make estimations, predictions and give the ability for automation without human efforts to learn by themselves is both powerful and limitless in term of application possibilities. We can use Artificial Intelligence in Robotics, Medicine, etc.. almost everywhere. That's why I decided to conduct my project in the Artificial Intelligence field.

1.3 Idea:-

As a first experience, we wanted to make our (MD SAQUEB and RAFI) project as much didactic as possible by approaching every different step in the Artificial Intelligence (A.I) field and trying to understand them deeply. Known as" Automation" the use of machines and computers that can operate without needing human control. The goal was to log in the online class (i.e., my class platform) and   attend classes with the help of automation, without human involvement and also the different "features" that will be developed in the following

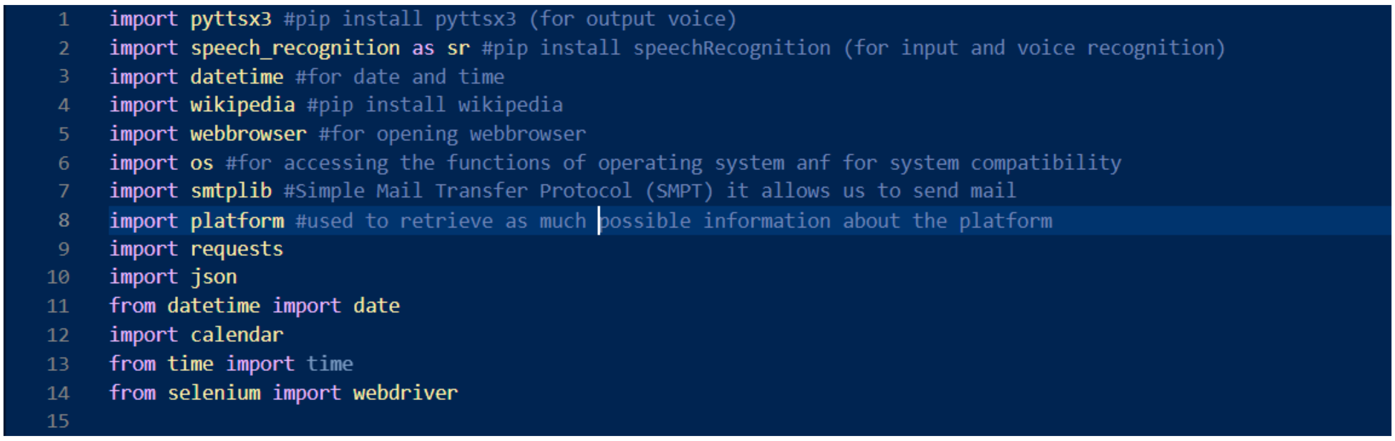
Introduction

It’s cool, to have our own A.I. assistant. It makes easier our some work like sending mail without typing a single word, doing Wikipedia searches without opening web browser, search information on google without typing and performing many other daily tasks like playing music with the help of a single voice command. We make our personal A.I (F.R.A.I.S. means Female Represented Artificial Intelligence System) using Python.

What can we do with this A.I assistant?

- It can do Wikipedia Searches.
- It can do Google searches.
- It can search videos on YouTube.
- It can forecast and location weather report.
- It can play music through music directory.
- It can tell us the current time.
- It can open Visual Studio Code.
- It can send mails to anyone.
- It can update us with daily news.
- It can reboot PC like Shutdown and Restart.
- It can attend our online class with automation of Myclass Portal.

❖ Modules Used in A.I.



```
import pyttsx3 #pip install pyttsx3 (for output voice)
import speech_recognition as sr #pip install speechRecognition (for input and voice recognition)
import datetime #for date and time
import wikipedia #pip install wikipedia
import webbrowser #for opening webbrowser
import os #for accessing the functions of operating system anf for system compatibility
import smtplib #Simple Mail Transfer Protocol (SMPT) it allows us to send mail
import platform #used to retrieve as much possible information about the platform
import requests
import json
from datetime import date
import calendar
from time import time
from selenium import webdriver
```

1. Pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline.

2. Speech Recognition

Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API, Microsoft Bing Voice Recognition and IBM Speech to Text.

3. Datetime

Python Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so when you manipulate them, you are actually manipulating objects and not string or timestamps.

4. Wikipedia

Wikipedia is a Python library that makes it easy to access and parse data from Wikipedia. Search Wikipedia, get article summaries, get data like links and images from a page, and more. Wikipedia wraps the MediaWiki API so you can focus on using Wikipedia data, not getting it.

5. Webbrowser

The webbrowser module provides a high-level interface to allow displaying web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing.

6. OS

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os. path* modules include many functions to interact with the file system.

7. Smtplib

The smtplib module defines an SMTP client session object that can be used to send mail to any internet machine with an SMTP or ESMTP listener daemon.

8. Platform

The platform module in Python is used to access the underlying platform's data, such as, hardware, operating system, and interpreter version information. The platform module includes tools to see the platform's hardware, operating, system, and interpreter version information where the program is running.

9. Request

The requests module allows you to send HTTP requests using Python. The HTTP request returns a Response Object with all the response data (content, encoding, status, etc.)

10. Json

The JSON module is mainly used to convert the python dictionary above into a JSON string that can be written into a file. While the JSON module will convert strings to Python datatypes, normally the JSON functions are used to read and write directly from JSON files.

11. Calendar

Python defines an inbuilt module calendar that handles operations related to the calendar. The calendar module allows output calendars like the program and provides additional useful functions related to the calendar.

12. Selenium

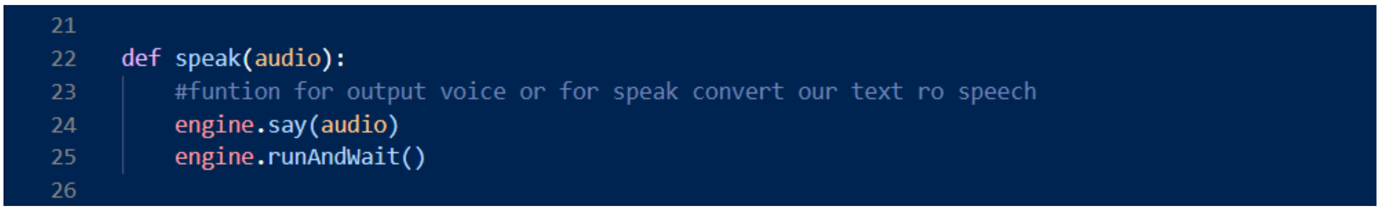
Selenium is an open-source web-based automation tool. Python language is used with Selenium for testing. It has far less verbose and easy to use than any other programming language. The Python APIs empower you to connect with the browser through Selenium.

❖ Making of our A.I.

➢ Defining Speak Function.

The first and foremost thing for an A.I. assistant is that it should be able to speak. To make our **F.R.A.I.S.** talk, we will make a function called **speak()**. This function will take audio as an argument, and then it will pronounce it and convert our text to speech.

Code:



```python

def speak(audio):
    #funtion for output voice or for speak convert our text ro speech
    engine.say(audio)
    engine.runAndWait()

```

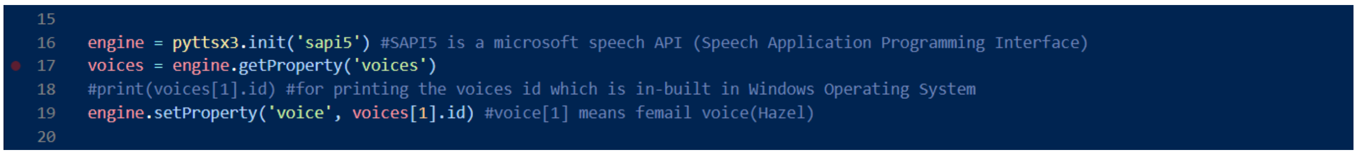
Now, the next thing we need is audio. We must supply audio so that we can pronounce it using the **speak()** function we made. We install a module called pyttsx3.

What is pyttsx3?

- A python library that will help us to convert text to speech. In short, it is a text-to-speech library.
- It works offline, and it is compatible with Python 2 as well as Python 3.

Installation of Module:
pip install pyttsx3

usage(code):



```python

engine = pyttsx3.init('sapi5') #SAPI5 is a microsoft speech API (Speech Application Programming Interface)
voices = engine.getProperty('voices')
#print(voices[1].id) #for printing the voices id which is in-built in Windows Operating System
engine.setProperty('voice', voices[1].id) #voice[1] means femail voice(Hazel)

```

What is sapi5?

- Microsoft developed speech API.
- Helps in synthesis and recognition of voice.

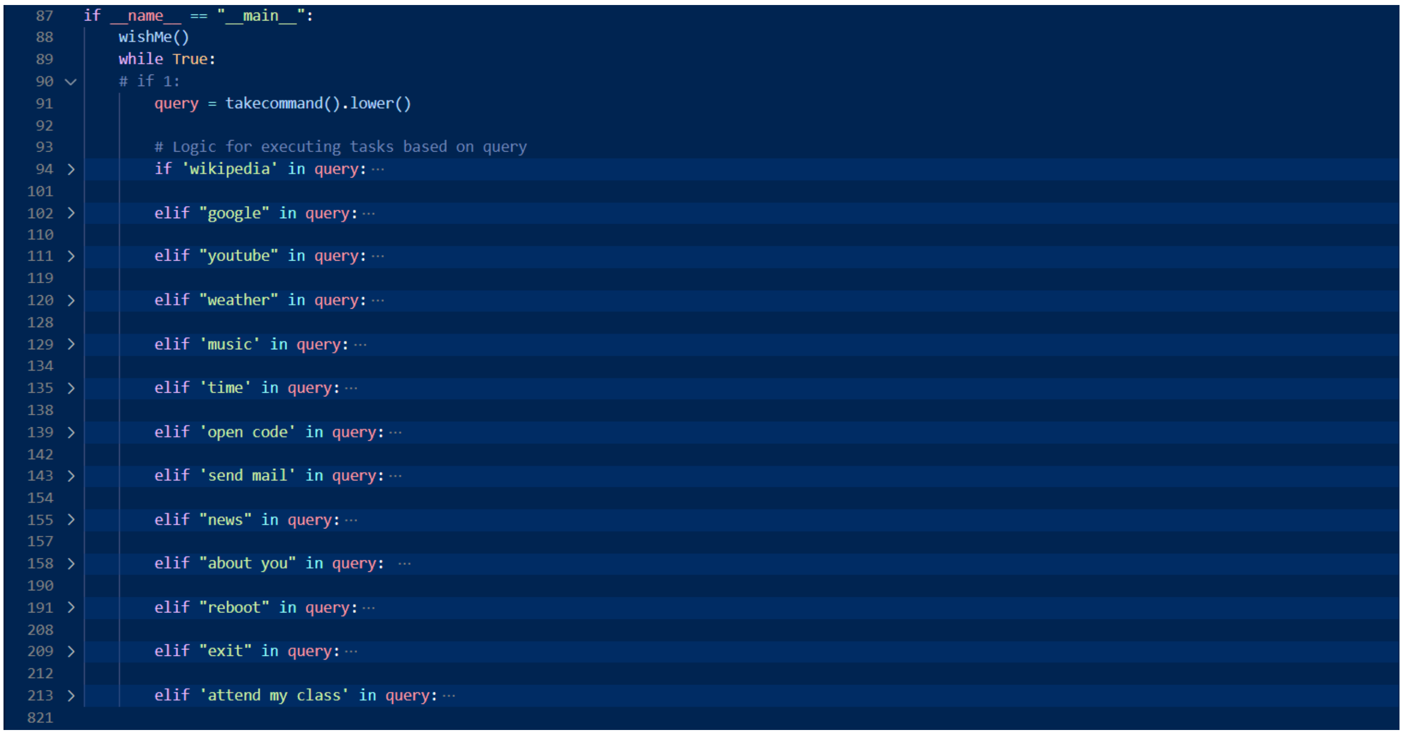
What Is VoiceId?

- Voice id helps us to select different voices.
- voice[0].id = Male voice
- voice[1].id = Female voice

Creating Our main() function:

We created a **main**() function, and inside this **main**() Function, we will call our speak function.

Code:



```
if __name__ == "__main__":
    wishMe()
    while True:
    # if 1:
        query = takecommand().lower()

        # Logic for executing tasks based on query
        if 'wikipedia' in query:···

        elif "google" in query:···

        elif "youtube" in query:···

        elif "weather" in query:···

        elif 'music' in query:···

        elif 'time' in query:···

        elif 'open code' in query:···

        elif 'send mail' in query:···

        elif "news" in query:···

        elif "about you" in query:  ···

        elif "reboot" in query:···

        elif "exit" in query:···

        elif 'attend my class' in query:···
```

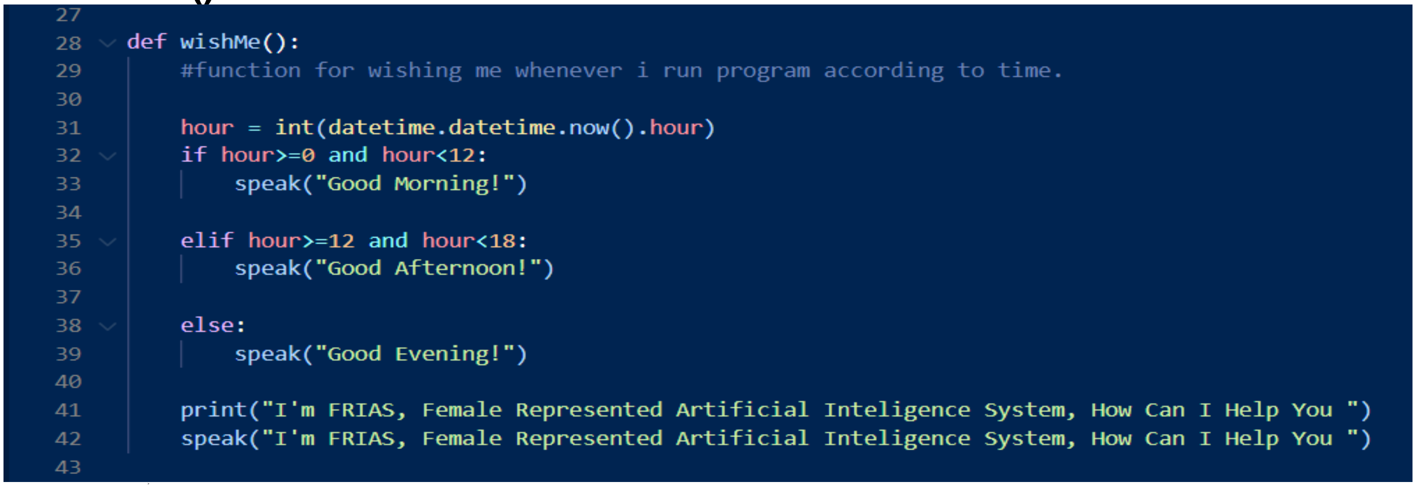
After this our A.I. has its own voice and it is ready to speak and proper functions and commands to work.

➢ Defining Wishme() function

we make a **wishme**() function that will make our **F.R.A.I.S.** wish or greet the us according to the time of computer or pc. To provide current or live time to A.I., we import a module called datetime.

Import this module to your program by:
import datetime

wishme() function Code:



```
def wishMe():
    #function for wishing me whenever i run program according to time.

    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!")

    print("I'm FRIAS, Female Represented Artificial Inteligence System, How Can I Help You ")
    speak("I'm FRIAS, Female Represented Artificial Inteligence System, How Can I Help You ")
```

Here, we have stored the current hour or time integer value into a variable named hour. then, we use this hour value inside an if-else loop for wish us according to time range like Good Morning between 00:00 to 12:00 and Good Afternoon between 12:00 to 18:00 and if time is not between these time then it wishes Good Evening.

➢ Defining Take command Function:

The next most important thing for our A.I. assistant is that it should take command with the help of the microphone of the user's system. So, we make a **takeCommand**() function. With the help of the **takecommand**() function, our A.I. assistant will return a string output by taking microphone input from the user.

Before defining the **takecommand**() function, we need to install a module called speechRecognition.

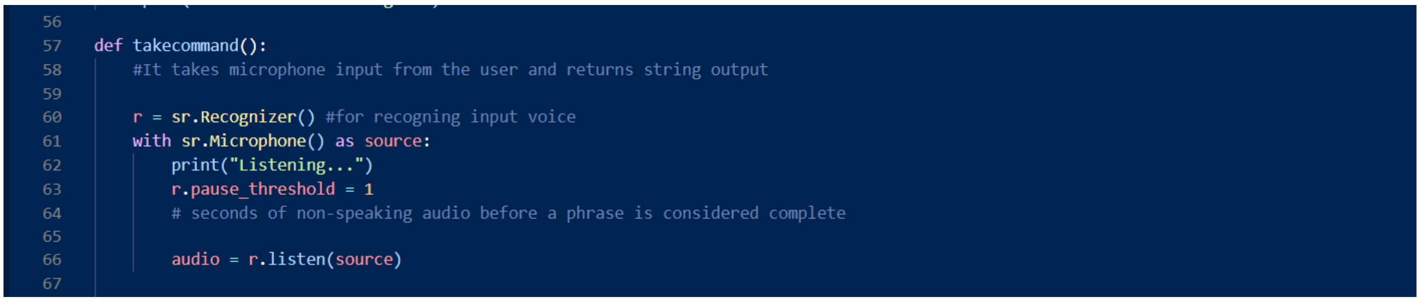
Install the module by:

pip install speechRecognition

After successfully installing this module, import this module into the program by writing an import statement.

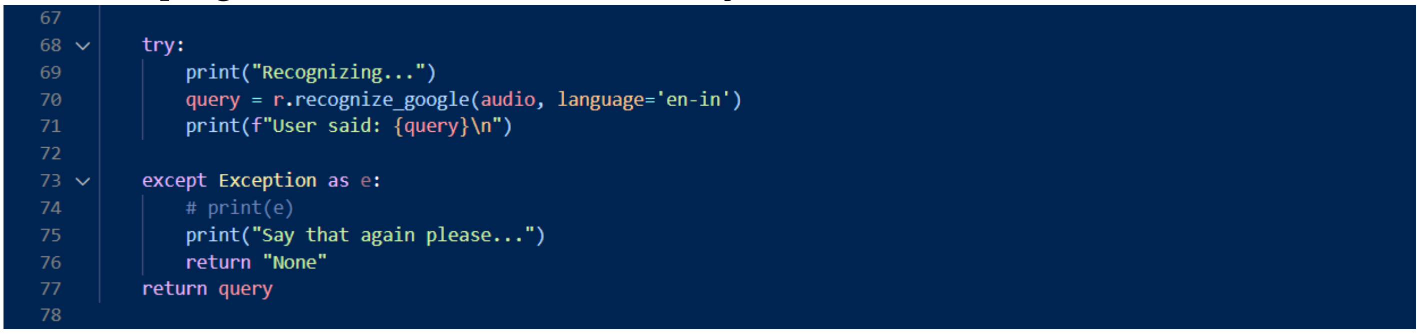
import speechRecognition as sr

Code for the takecommand() function:



```python

def takecommand():
    #It takes microphone input from the user and returns string output

    r = sr.Recognizer() #for recogning input voice
    with sr.Microphone() as source:
        print("Listening...")
        r.pause_threshold = 1
        # seconds of non-speaking audio before a phrase is considered complete

        audio = r.listen(source)

```

After, creating our **takeCommand**() function. we add a try and except block to our program to handle errors effectively.



```python

    try:
        print("Recognizing...")
        query = r.recognize_google(audio, language='en-in')
        print(f"User said: {query}\n")

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

```

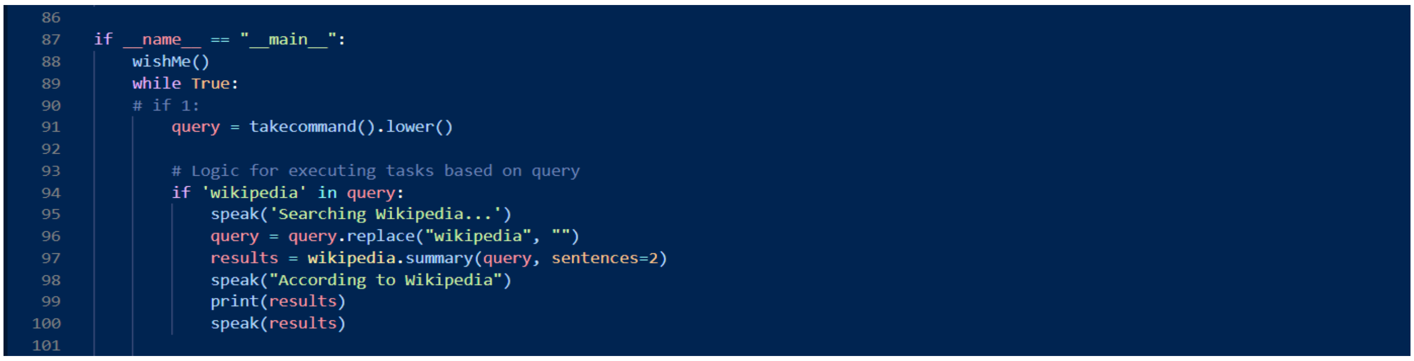
- **Coding logic of our A.I.**

we develop logic for different commands such as Wikipedia searches, playing music, etc.

➢ Defining Task 1: To search something on Wikipedia

To do Wikipedia searches, we install and import the Wikipedia module into our program.

Type the below command to install the Wikipedia module :
pip install wikipedia



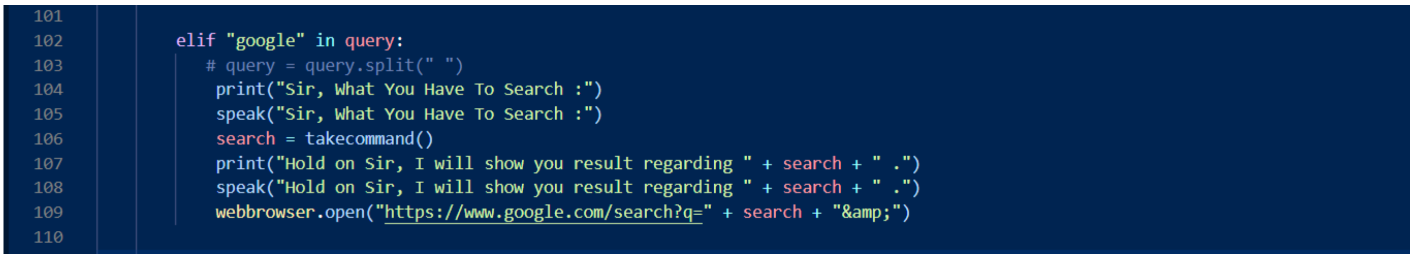
```python

if __name__ == "__main__":
    wishMe()
    while True:
    # if 1:
        query = takecommand().lower()

        # Logic for executing tasks based on query
        if 'wikipedia' in query:
            speak('Searching Wikipedia...')
            query = query.replace("wikipedia", "")
            results = wikipedia.summary(query, sentences=2)
            speak("According to Wikipedia")
            print(results)
            speak(results)

```

In the above code, we used an if statement to check whether Wikipedia is in the user's search query or not. If Wikipedia is found in the user's search query, then two sentences from the summary of the Wikipedia page will be converted to speech with the speak function's help.

➢ Defining Task 2: To do Google searches

To open any website, we need to import a module called webbrowser. It is an in-built module, and we do not need to install it with a pip statement; we can directly import it into our program by writing an import statement.

Code:

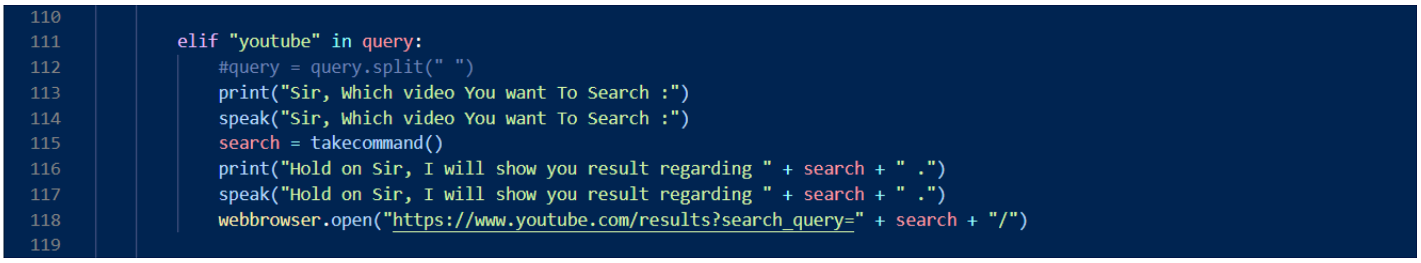


```python

        elif "google" in query:
            # query = query.split(" ")
            print("Sir, What You Have To Search :")
            speak("Sir, What You Have To Search :")
            search = takecommand()
            print("Hold on Sir, I will show you result regarding " + search + " .")
            speak("Hold on Sir, I will show you result regarding " + search + " .")
            webbrowser.open("https://www.google.com/search?q=" + search + "&amp;")

```

Here, we using an elif loop to check whether Google is in the user's query. Let's suppose "F.R.A.I.S., search on google" then it will ask us what to search then we can give it command using **takeCommand()** function so, open search google will be in the user's query, and the elif condition will be true.

➢ Defining Task 3: To do video searches on YouTube.

Code:

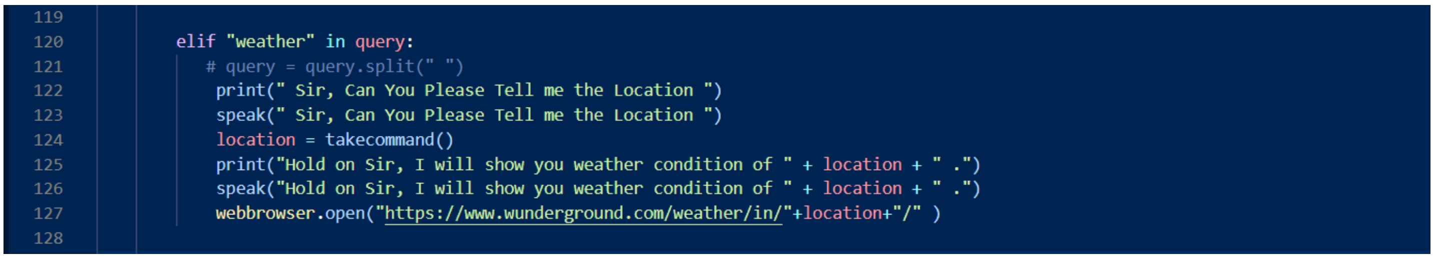


```
elif "youtube" in query:
    #query = query.split(" ")
    print("Sir, Which video You want To Search :")
    speak("Sir, Which video You want To Search :")
    search = takecommand()
    print("Hold on Sir, I will show you result regarding " + search + " .")
    speak("Hold on Sir, I will show you result regarding " + search + " .")
    webbrowser.open("https://www.youtube.com/results?search_query=" + search + "/")
```

We are searching video on YouTube in a web-browser by applying the same logic that we used to search something on Google.

➢ Defining Task 4: To forecast weather report of any location.

Code:

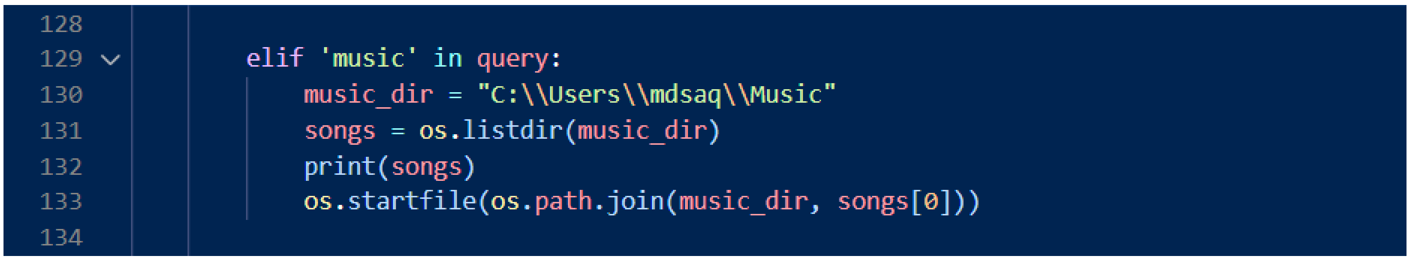


```
elif "weather" in query:
   # query = query.split(" ")
    print(" Sir, Can You Please Tell me the Location ")
    speak(" Sir, Can You Please Tell me the Location ")
    location = takecommand()
    print("Hold on Sir, I will show you weather condition of " + location + " .")
    speak("Hold on Sir, I will show you weather condition of " + location + " .")
    webbrowser.open("https://www.wunderground.com/weather/in/"+location+"/" )
```

We are forecasting weather report of any location in a web-browser by applying the same logic that we used to search something on Google and search video on YouTube.

➢ Defining Task 5: To play music

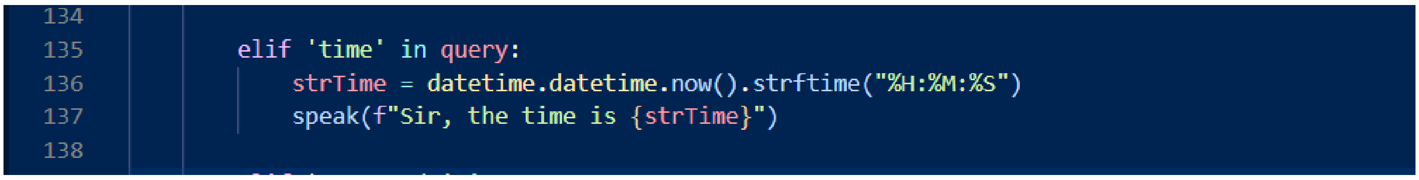
To play music, we need to import a module called os(Operating System). Import this module directly with an import statement.



```
        elif 'music' in query:
            music_dir = "C:\\Users\\mdsaq\\Music"
            songs = os.listdir(music_dir)
            print(songs)
            os.startfile(os.path.join(music_dir, songs[0]))
```

In the above code, we first opened our music directory and then listed all the songs present in the directory with the os module's help. With the help of **os.startfile**, you can play any song of your choice. We are playing the first song in the directory. However, you can also play a random song with the help of a random module. Every time you command to play music, **F.R.A.I.S.** will play any random song from the song directory.

➢ Defining Task 6: To know the current time

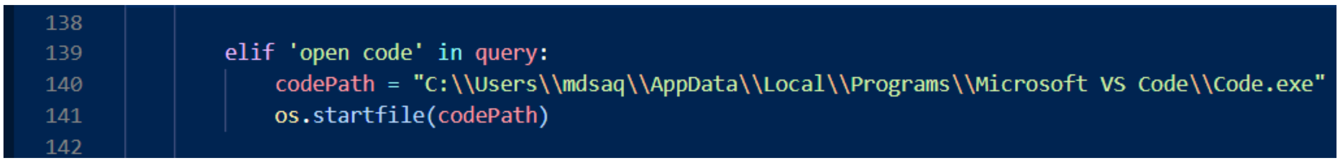


```
        elif 'time' in query:
            strTime = datetime.datetime.now().strftime("%H:%M:%S")
            speak(f"Sir, the time is {strTime}")
```

In the above, code we are using the **datetime**() function and storing the current or live system time into a variable called strTime. After storing the time in strTime, we are passing this variable as an argument in speak function. Now, the time string will be converted into speech.

➢ Defining Task 7: To open the Visual Studio Code Program.

Code:



```
        elif 'open code' in query:
            codePath = "C:\\Users\\mdsaq\\AppData\\Local\\Programs\\Microsoft VS Code\\Code.exe"
            os.startfile(codePath)
```

To open the VS Code or any other application, we need the code path of the application.

Steps to get the code path of the application:
Step 1: Open the file location.
Step 2: Right-click on the application and click on properties.
Step 3: Copy the target from the target section.

After copying the target of the application, save the target into a variable. Here, I am saving the target into a variable called codePath, and then we are using the os module to open the application.

➢ Defining Task 8: To Send Email.

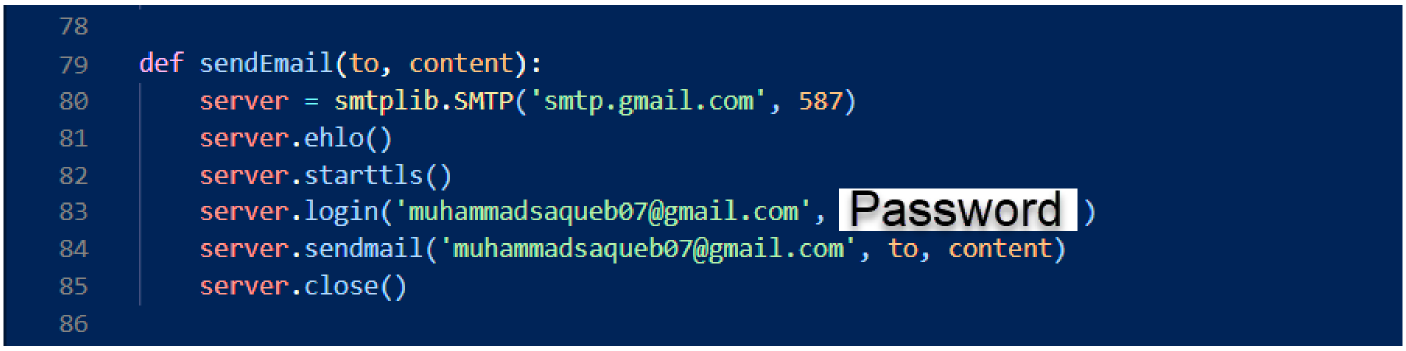
To send an email, we need to import a module called smtplib.
What is smtplib?

- **Simple Mail Transfer Protocol (SMTP)** is a protocol that allows us to send emails and route emails between mail servers. An instance method called sendmail is present in the SMTP module. This instance method allows us to send an email. It takes 3 parameters:
- The sender: Email address of the sender.
- The receiver: T Email of the receiver.
- The *message:* A string message which needs to be sent to one or more than one recipient.

➢ Defining Send email function:

We create a **sendEmail**() function, which will help us send emails to one or more than one recipient.

Code:



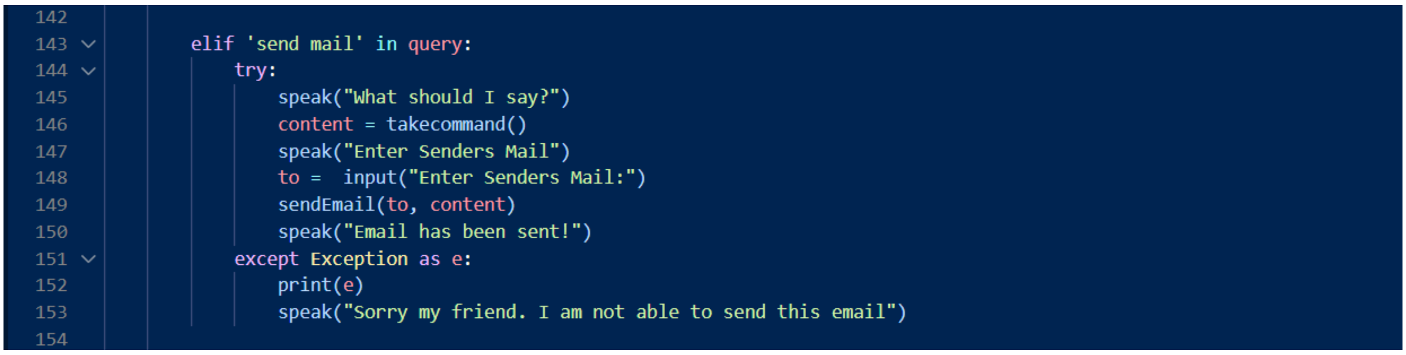
```python

def sendEmail(to, content):
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.ehlo()
    server.starttls()
    server.login('muhammadsaqueb07@gmail.com', Password )
    server.sendmail('muhammadsaqueb07@gmail.com', to, content)
    server.close()

```

In the above code, we are using the SMTP module, which we have already discussed above.

Note: Do not forget to *'enable the less secure apps'* feature in your Gmail account. Otherwise, the sendEmail function will not work properly.

➢ Calling sendEmail() function inside the main() function:

Code:



```python

        elif 'send mail' in query:
            try:
                speak("What should I say?")
                content = takecommand()
                speak("Enter Senders Mail")
                to =  input("Enter Senders Mail:")
                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")

```

We are using try and except block to handle any possible error while sending emails.

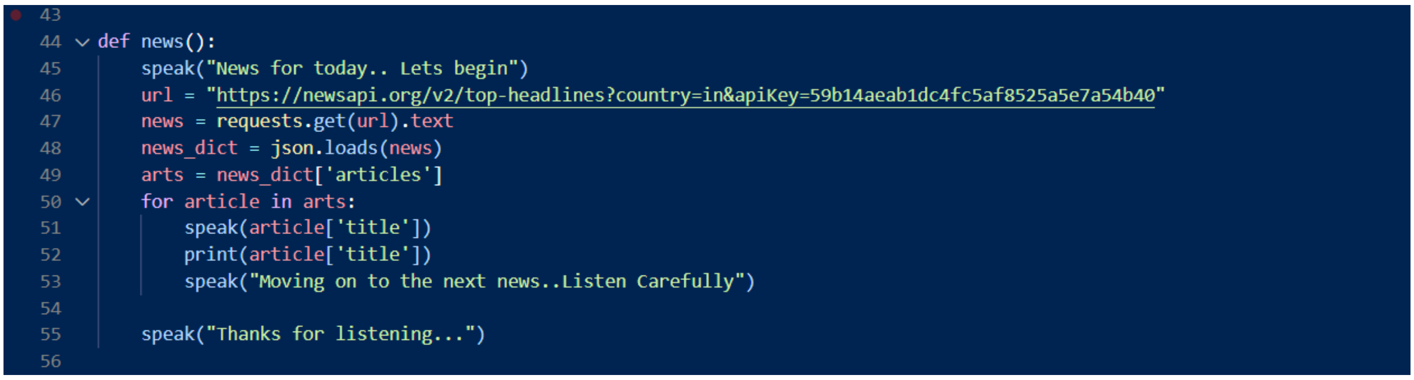
➢ Defining Task 9: for Daily News.

This function will give us daily top 10 latest news.
For that, we have to log on the website *https://newsapi.org/* which gives the news API.
First, we created an account on that website, and then we get a free news API.

What we have to do:

- We get the most relevant and latest news API from *https://newsapi.org/*.
- After we have the news API, we install the package using the statement:
  "pip install pynin32"

Code:



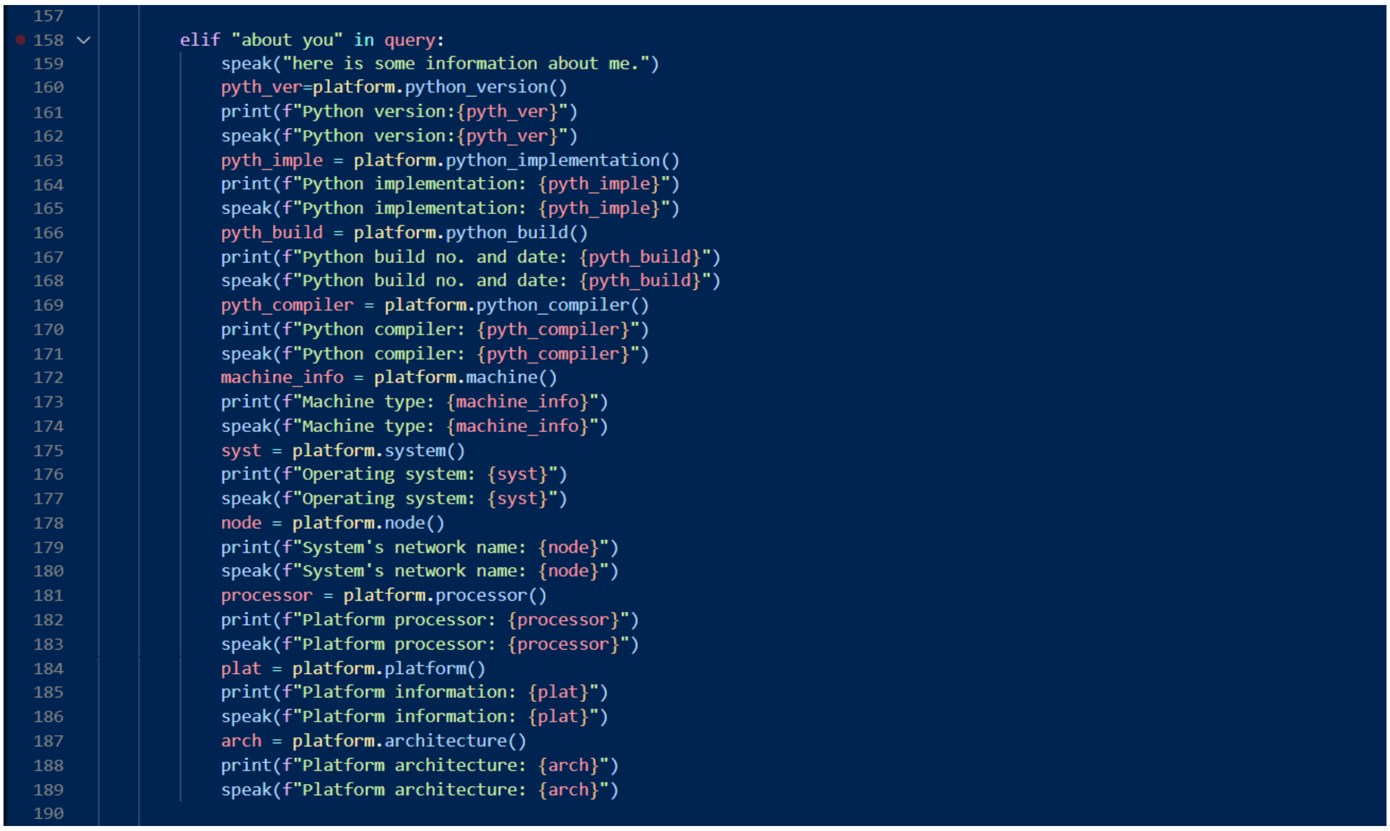
```python
def news():
    speak("News for today.. Lets begin")
    url = "https://newsapi.org/v2/top-headlines?country=in&apiKey=59b14aeab1dc4fc5af8525a5e7a54b40"
    news = requests.get(url).text
    news_dict = json.loads(news)
    arts = news_dict['articles']
    for article in arts:
        speak(article['title'])
        print(article['title'])
        speak("Moving on to the next news..Listen Carefully")

    speak("Thanks for listening...")
```

Using **speak()** function the A.I. speak top 10 latest news.

We use JSON module and request module to make a newsreader

➢ Defining Task 10: To know all information about our A.I.

Using platform module our tell us about it’s specifications.

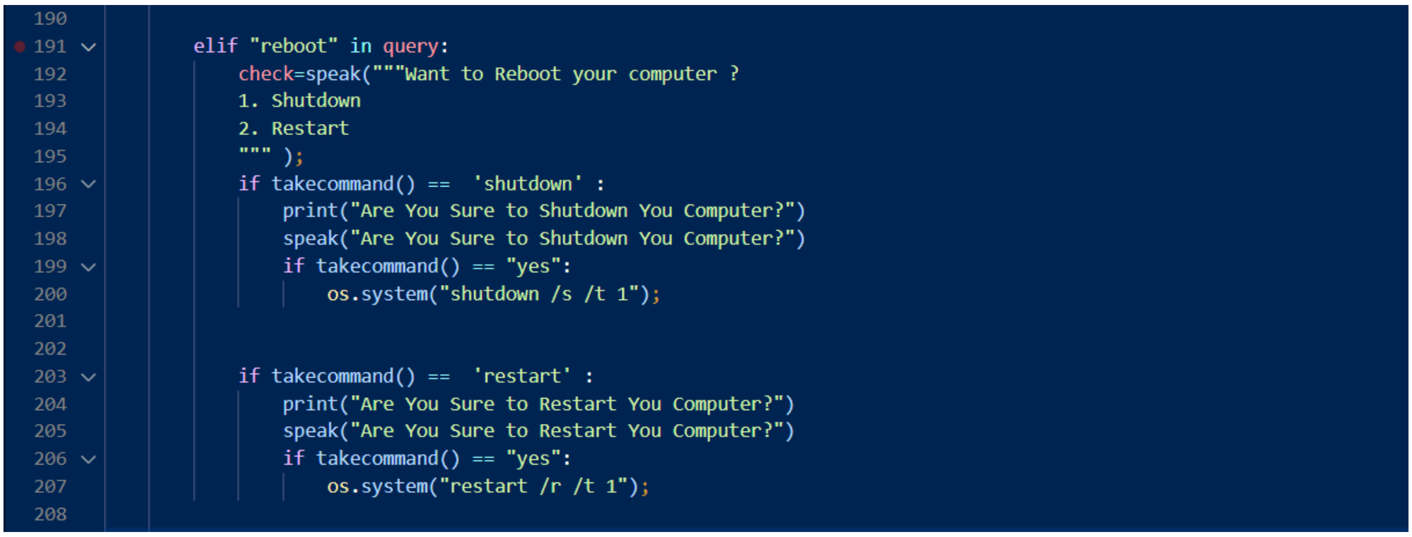


```python
        elif "about you" in query:
            speak("here is some information about me.")
            pyth_ver=platform.python_version()
            print(f"Python version:{pyth_ver}")
            speak(f"Python version:{pyth_ver}")
            pyth_imple = platform.python_implementation()
            print(f"Python implementation: {pyth_imple}")
            speak(f"Python implementation: {pyth_imple}")
            pyth_build = platform.python_build()
            print(f"Python build no. and date: {pyth_build}")
            speak(f"Python build no. and date: {pyth_build}")
            pyth_compiler = platform.python_compiler()
            print(f"Python compiler: {pyth_compiler}")
            speak(f"Python compiler: {pyth_compiler}")
            machine_info = platform.machine()
            print(f"Machine type: {machine_info}")
            speak(f"Machine type: {machine_info}")
            syst = platform.system()
            print(f"Operating system: {syst}")
            speak(f"Operating system: {syst}")
            node = platform.node()
            print(f"System's network name: {node}")
            speak(f"System's network name: {node}")
            processor = platform.processor()
            print(f"Platform processor: {processor}")
            speak(f"Platform processor: {processor}")
            plat = platform.platform()
            print(f"Platform information: {plat}")
            speak(f"Platform information: {plat}")
            arch = platform.architecture()
            print(f"Platform architecture: {arch}")
            speak(f"Platform architecture: {arch}")
```

➢ Defining Task 11: To reboot PC like Shutdown and Restart.

Using simple if-else loop we just created function that will restart and shutdown System.

Code:

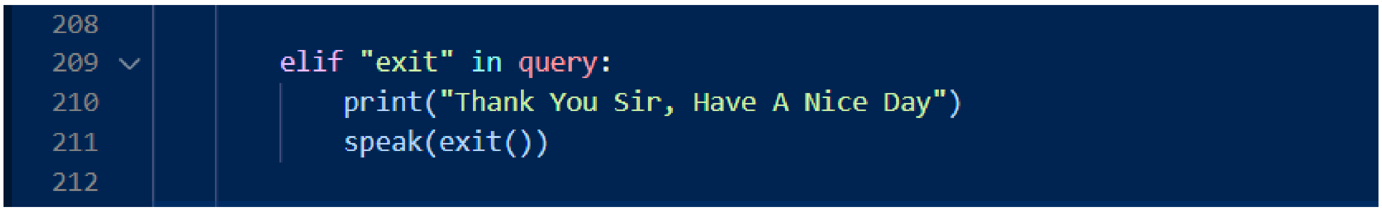


```
    elif "reboot" in query:
        check=speak("""Want to Reboot your computer ?
        1. Shutdown
        2. Restart
        """ );
        if takecommand() ==  'shutdown' :
            print("Are You Sure to Shutdown You Computer?")
            speak("Are You Sure to Shutdown You Computer?")
            if takecommand() == "yes":
                os.system("shutdown /s /t 1");


        if takecommand() ==  'restart' :
            print("Are You Sure to Restart You Computer?")
            speak("Are You Sure to Restart You Computer?")
            if takecommand() == "yes":
                os.system("restart /r /t 1");
```

➢ Defining Task 12: To Exit the Running Program.

Using in-built **exit()** function we created a function which can close code

Code:

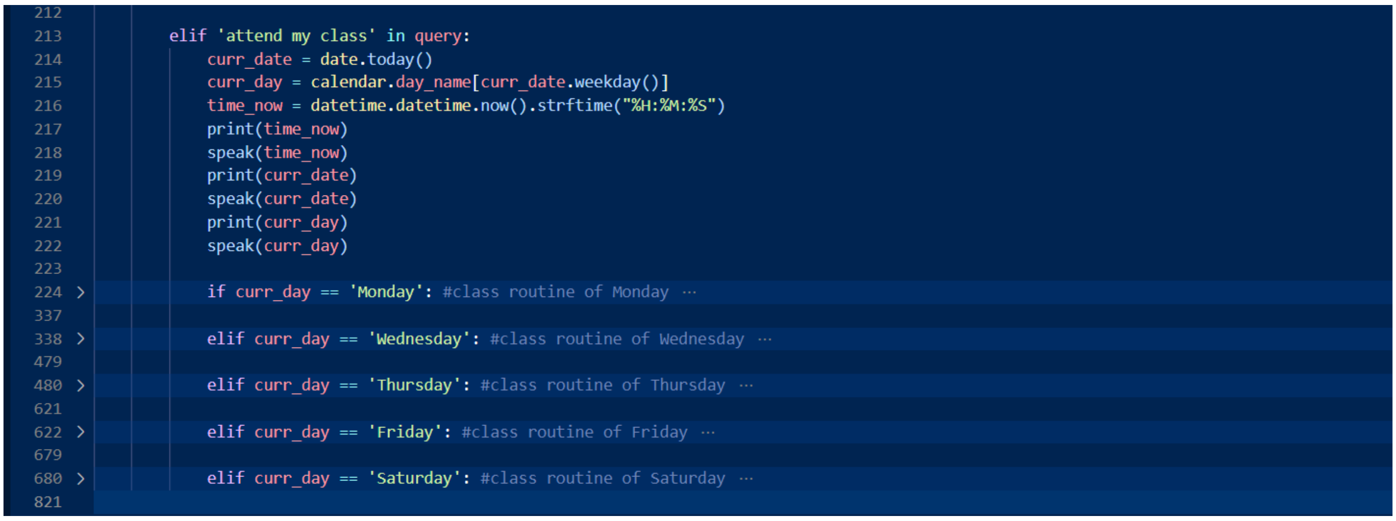


```
    elif "exit" in query:
        print("Thank You Sir, Have A Nice Day")
        speak(exit())
```

➢ Defining Task 13: To Attend our Class.

We created web automation using selenium module. Using xpath of links of classes and by passing our credentials to access our account. From web driver and chrome driver we are accessing the elements of our Myclass website. We just put a particular time gap for attending our class like if the day and time gap is match's with the system time then that particular class will start automatically using accessing the elements

Code:



```python

        elif 'attend my class' in query:
            curr_date = date.today()
            curr_day = calendar.day_name[curr_date.weekday()]
            time_now = datetime.datetime.now().strftime("%H:%M:%S")
            print(time_now)
            speak(time_now)
            print(curr_date)
            speak(curr_date)
            print(curr_day)
            speak(curr_day)

            if curr_day == 'Monday': #class routine of Monday ···

            elif curr_day == 'Wednesday': #class routine of Wednesday ···

            elif curr_day == 'Thursday': #class routine of Thursday ···

            elif curr_day == 'Friday': #class routine of Friday ···

            elif curr_day == 'Saturday': #class routine of Saturday ···

```

Conclusion

It is our team’s hope that this document will be of huge help with understanding of our little project as we have used a different approach which has proved beneficial for us and easy for us to understand the vast ocean that is Artificial Intelligence. We hope this report will be of help in understanding our python project as to not confuse the reader in understanding the different approaches we used to bring this A.I. assistant **(F.R.A.I.S.)** into play. We really enjoyed this whole process of learning new concepts and implementing them in our code, and we look forward to creating more such amazing projects.

Reference

- **https://pypi.org/**
- **https://www.geeksforgeeks.org/**
- **https://github.com/muhammadsaqueb/CA1-INT213-Project.git** **(Project Link)**