Desktop Assistant

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

Shweta

Submitted to:

(Shwetz Agarwal : E9500)

Ass. Prof. Shweta Ma'am

Submitted By:

Vikash Kumar Gupta (18BCS6641)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Chandigarh University, Gharuan

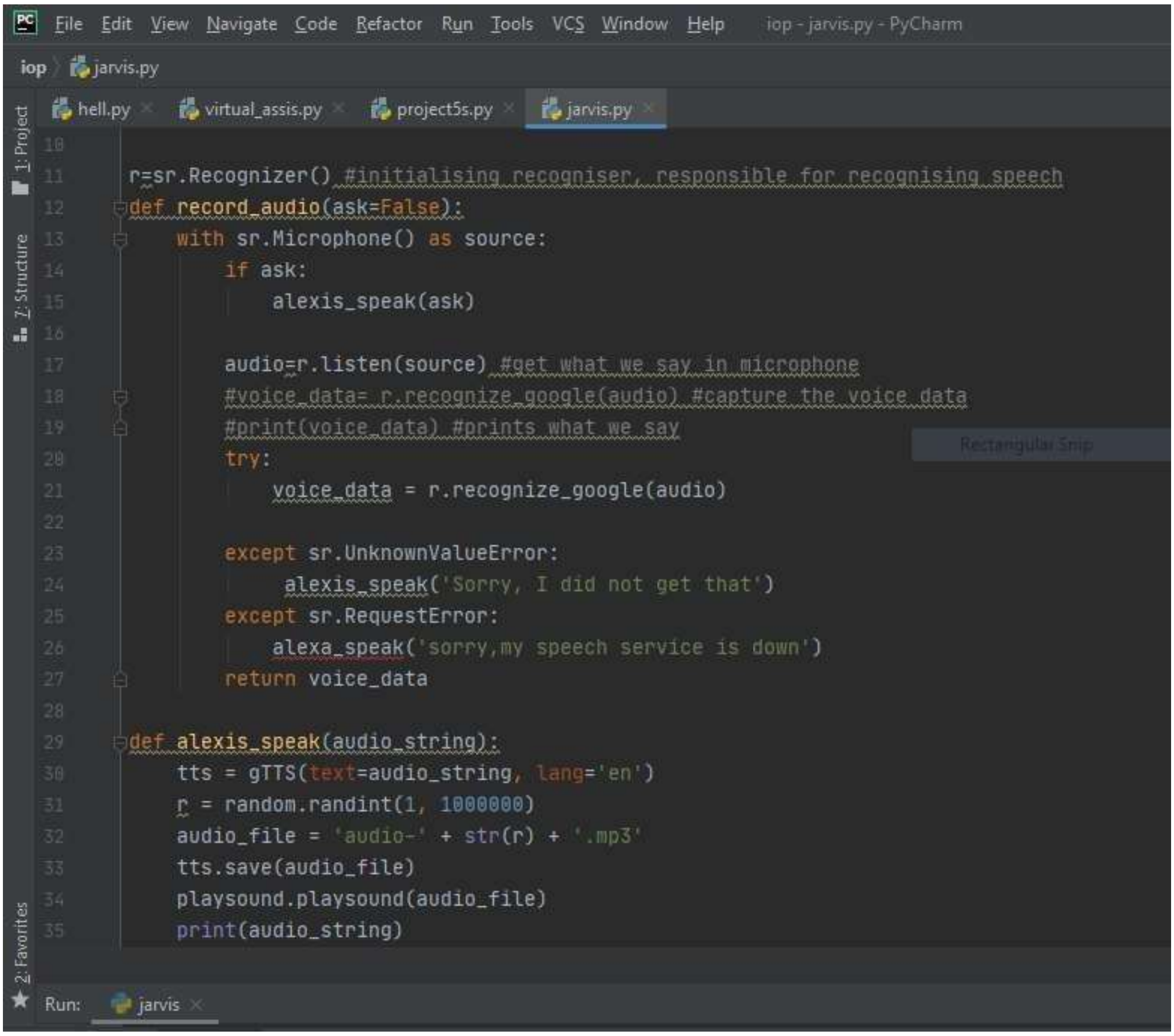
April 2021

Implementation

There will be a total of 8 major tasks in this project. The modules are as explained below:

- **Listening to the user's command**– Defining a function to record the audio command of the user and printing it on the screen also. For this I used **speech recognition** library of python. This converts spoken text into python strings.

- **Get a response from the assistant** – Defining a function to make the assistant respond by speaking back to the user. This function uses the **google text to speech(gTTS)** library to convert the assistance response in audio form.



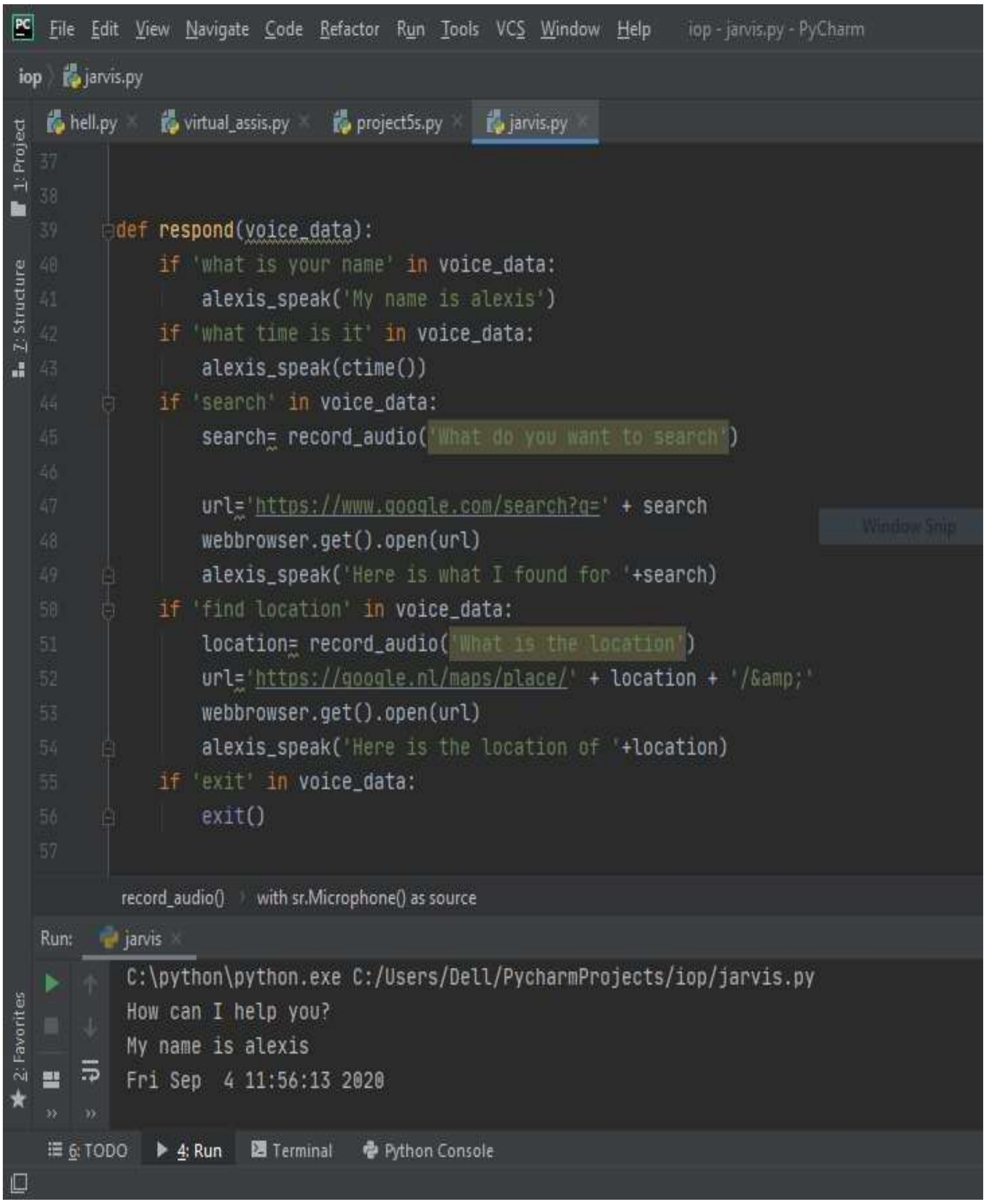
```
r=sr.Recognizer() #initialising recogniser, responsible for recognising speech
def record_audio(ask=False):
    with sr.Microphone() as source:
        if ask:
            alexis_speak(ask)

        audio=r.listen(source) #get what we say in microphone
        #voice_data= r.recognize_google(audio) #capture the voice data
        #print(voice_data) #prints what we say
        try:
            voice_data = r.recognize_google(audio)

        except sr.UnknownValueError:
             alexis_speak('Sorry, I did not get that')
        except sr.RequestError:
            alexa_speak('sorry,my speech service is down')
        return voice_data

def alexis_speak(audio_string):
    tts = gTTS(text=audio_string, lang='en')
    r = random.randint(1, 1000000)
    audio_file = 'audio-' + str(r) + '.mp3'
    tts.save(audio_file)
    playsound.playsound(audio_file)
    print(audio_string)
```

- **Greeting the user and telling the date and time** – Responding with greetings How may I help when the program is started. If the user asks what time is it it will respond with the exact date and time at that moment. This uses **ctime** function to get the time.



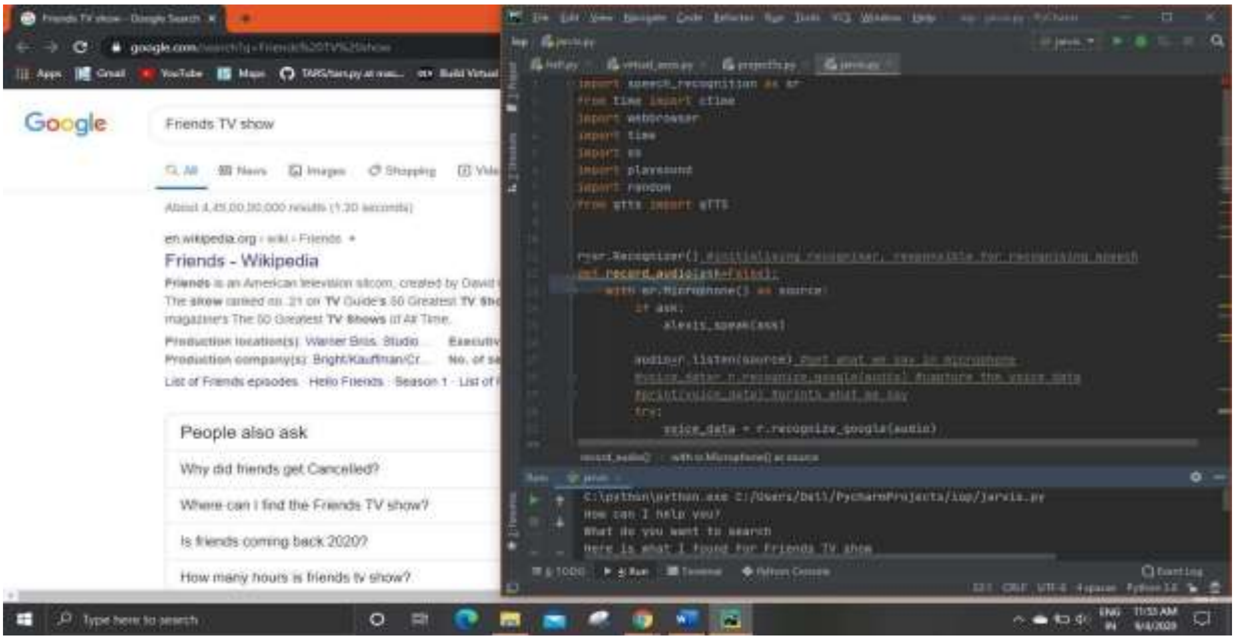
```
def respond(voice_data):
    if 'what is your name' in voice_data:
        alexis_speak('My name is alexis')
    if 'what time is it' in voice_data:
        alexis_speak(ctime())
    if 'search' in voice_data:
        search= record_audio('What do you want to search')

        url='https://www.google.com/search?q=' + search
        webbrowser.get().open(url)
        alexis_speak('Here is what I found for '+search)
    if 'find location' in voice_data:
        location= record_audio('What is the location')
        url='https://google.nl/maps/place/' + location + '/&amp;'
        webbrowser.get().open(url)
        alexis_speak('Here is the location of '+location)
    if 'exit' in voice_data:
        exit()
```

```
C:\python\python.exe C:/Users/Dell/PycharmProjects/iop/jarvis.py
How can I help you?
My name is alexis
Fri Sep  4 11:56:13 2020
```

- The main function – This function contains trigger words that will make the assistant perform the required task. The tasks are:

- □ **Searching the web** – If there is a search word in the user's statement the assistant will ask back what the user wants to search for and then opens the web browser and make the required query. This uses the **webbrowser** library to make the required search.



Friends TV show

Friends - Wikipedia

Friends is an American television sitcom, created by David ...

People also ask

Why did friends get Cancelled?

Where can I find the Friends TV show?

Is friends coming back 2020?

How many hours is friends tv show?

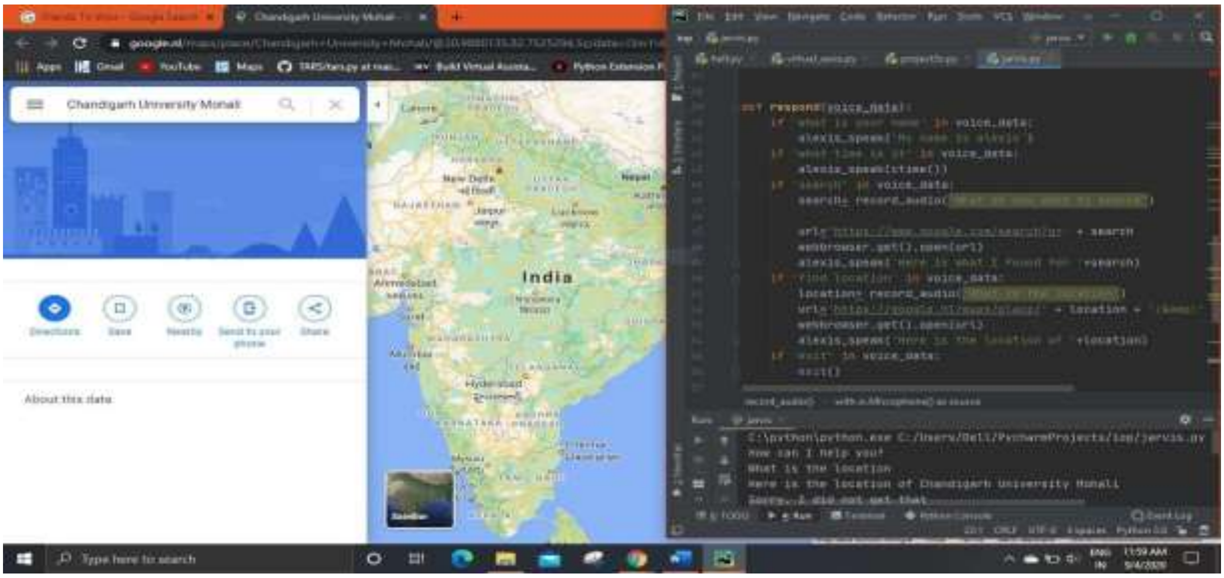
```
import speech_recognition as sr
from time import ctime
import webbrowser
import time
import os
import playsound
import random
from gtts import gTTS


r=sr.Recognizer() #initialising recogniser, responsible for recognising speech
def record_audio(ask=False):
    with sr.Microphone() as source:
        if ask:
            alexis_speak(ask)

        audio=r.listen(source) #get what we say in microphone
        #voice_data= r.recognize_google(audio) #capture the voice data
        #print(voice_data) #prints what we say
        try:
            voice_data = r.recognize_google(audio)
```

```
C:\python\python.exe C:/Users/Dell/PycharmProjects/iop/jarvis.py
How can I help you?
What do you want to search
Here is what I found for Friends TV show
```

- □ **Searching location**- – If there is a search location word in the user's statement the assistant will ask back what is the location the user wants to search for and then opens the google maps to show the location. This also uses webbrowser library to find the location.



Chandigarh University Mohali

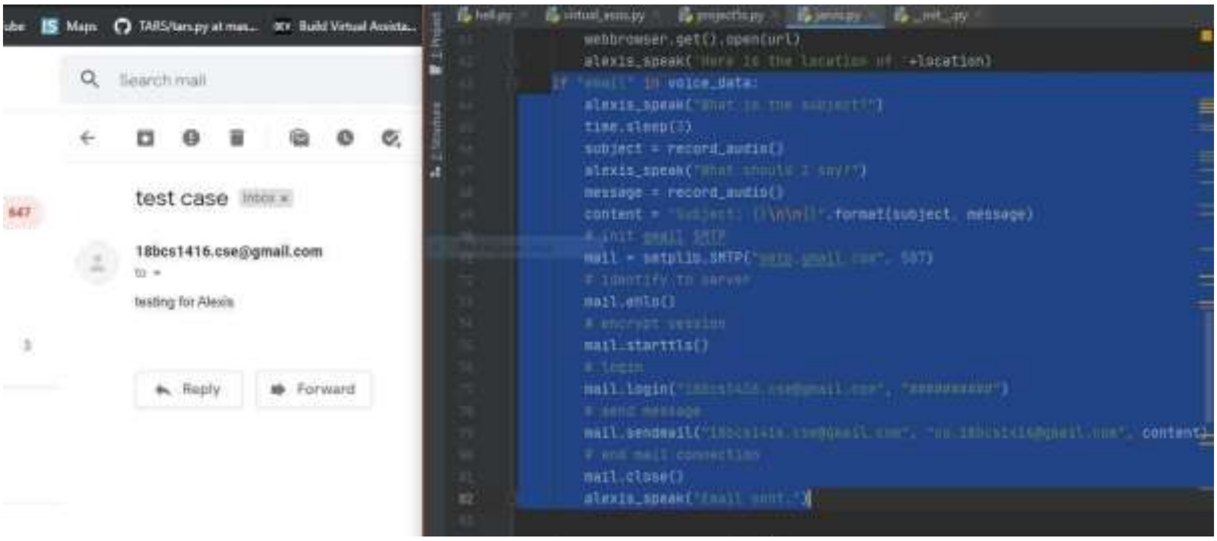
```
def respond(voice_data):
    if 'what is your name' in voice_data:
        alexis_speak('My name is alexis')
    if 'what time is it' in voice_data:
        alexis_speak(ctime())
    if 'search' in voice_data:
        search= record_audio('What do you want to search')

        url='https://www.google.com/search?q=' + search
        webbrowser.get().open(url)
        alexis_speak('Here is what I found for ' +search)
    if 'find location' in voice_data:
        location= record_audio('What is the location')
        url='https://google.nl/maps/place/' + location + '/&amp;'
        webbrowser.get().open(url)
        alexis_speak('Here is the location of ' +location)
    if 'exit' in voice_data:
        exit()
```

```
C:\python\python.exe C:/Users/Dell/PycharmProjects/iop/jarvis.py
How can I help you?
What is the location
Here is the location of Chandigarh University Mohali
Sorry, I did not get that
```

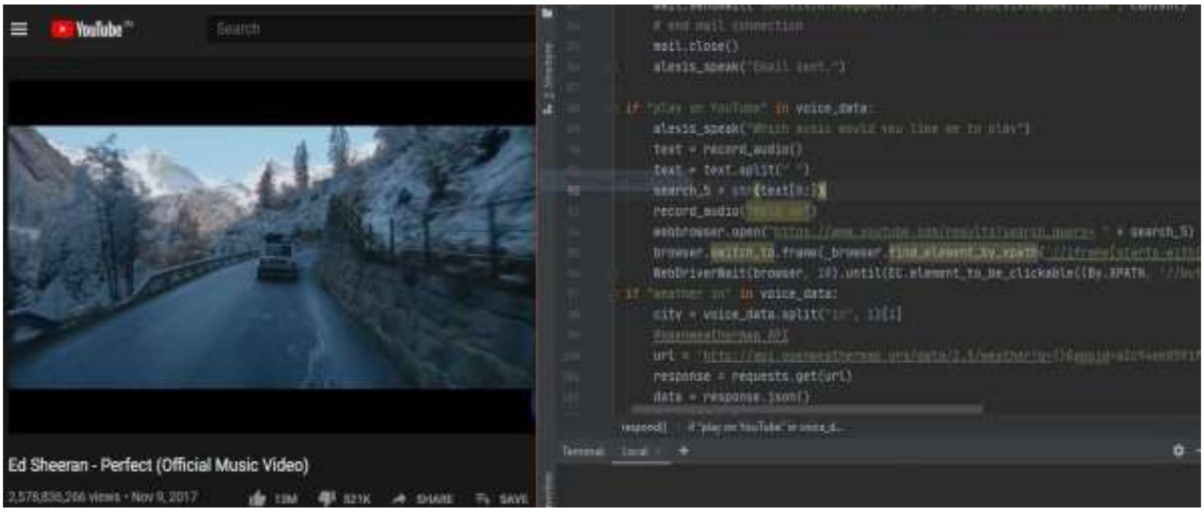
□ **Sending email** – If the user gives a command to send an email the assistant will ask for the subject and the body and then send the email to the required recipient. The **smtplib** library was used to implement this module.

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



```
        webbrowser.get().open(url)
        alexis_speak("Here is the location of " +location)
    if "email" in voice_data:
        alexis_speak("What is the subject?")
        time.sleep(3)
        subject = record_audio()
        alexis_speak("What should I say?")
        message = record_audio()
        content = "Subject: {}\n\n{}".format(subject, message)
        # init gmail SMTP
        mail = smtplib.SMTP("smtp.gmail.com", 587)
        # identify to server
        mail.ehlo()
        # encrypt session
        mail.starttls()
        # login
        mail.login("18bcs1416.cse@gmail.com", "*********")
        # send message
        mail.sendmail("18bcs1416.cse@gmail.com", "[illegible]", content)
        # end mail connection
        mail.close()
        alexis_speak("Email sent.")
```

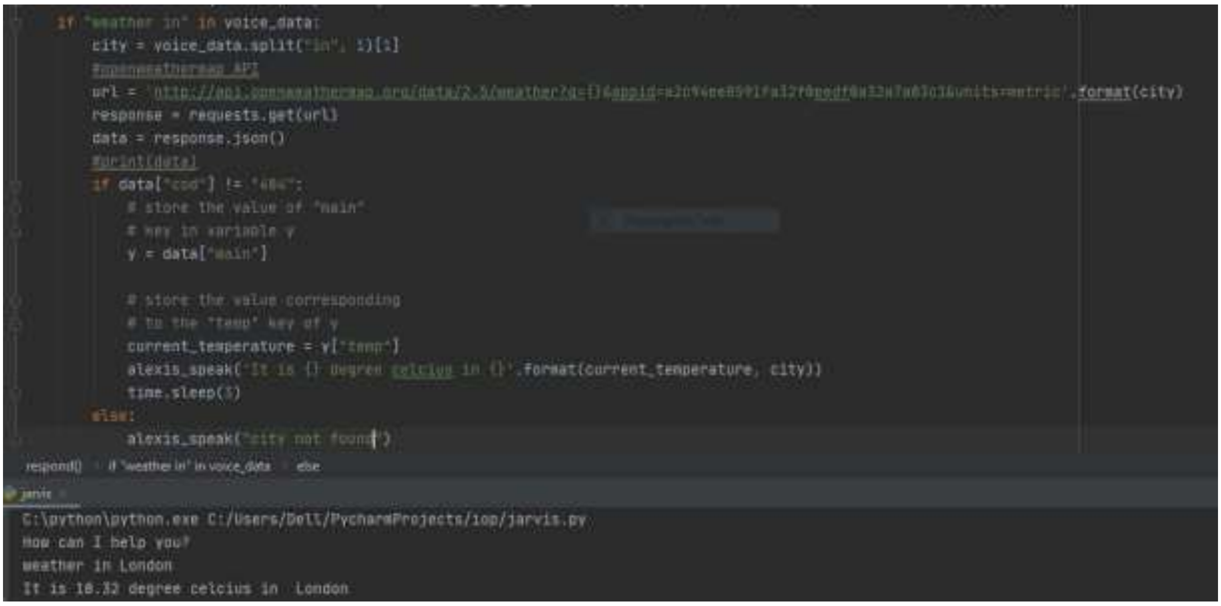
□ **Opening YouTube** – If the user asks for YouTube video the assistant opens the YouTube and then searches the query and plays the first video available on the list..


YouTube
Search
Ed Sheeran - Perfect (Official Music Video)
2,576,836,266 views • Nov 9, 2017
13M
521K
SHARE
SAVE
2: Structure
respond()
if "play on YouTube" in voice_d...
Terminal:
Local

```
        # end mail connection
        mail.close()
        alexis_speak("Email sent.")

    if "play on YouTube" in voice_data:
        alexis_speak("Which music would you like me to play?")
        text = record_audio()
        text = text.split(" ")
        search_5 = str(text[3:])
        record_audio([illegible])
        webbrowser.open("https://www.youtube.com/results?search_query=" + search_5)
        browser.switch_to.frame(_browser.find_element_by_xpath("//iframe[starts-with
        WebDriverWait(browser, 10).until(EC.element_to_be_clickable((By.XPATH, "//but
    if "weather in" in voice_data:
        city = voice_data.split("in", 1)[1]
        #openweatherMap API
        url = "http://api.openweathermap.org/data/2.5/weather?q={}&appid=a1c4ee091f
        response = requests.get(url)
        data = response.json()
```

☐ **Telling about the weather**- If the user asks about the weather the assistant tells the weather condition. **openweathermap** is a service that provides weather data, including current weather data, forecasts, and historical data to the developers of web services and mobile applications.

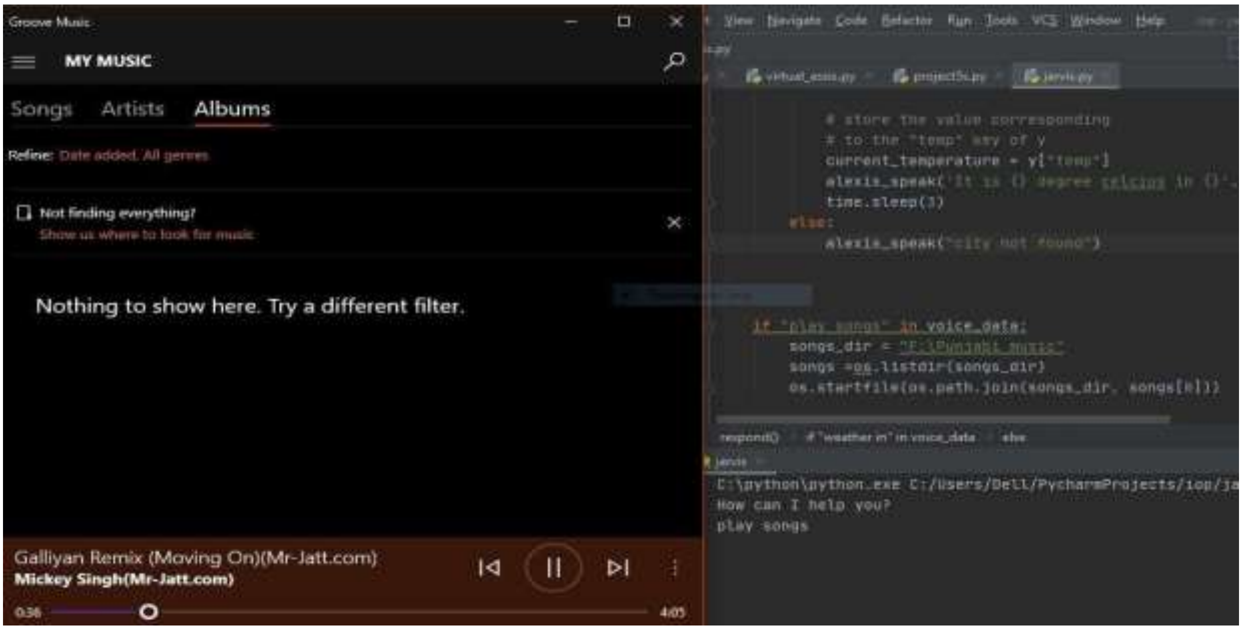


```
if "weather in" in voice_data:
    city = voice_data.split("in", 1)[1]
    #openweathermap API
    url = 'http://api.openweathermap.org/data/2.5/weather?q={}&appid=a2094ee8591fa12f0bedf0a32a7a03c3&units=metric'.format(city)
    response = requests.get(url)
    data = response.json()
    #print(data)
    if data["cod"] != "404":
        # store the value of "main"
        # key in variable y
        y = data["main"]

        # store the value corresponding
        # to the "temp" key of y
        current_temperature = y["temp"]
        alexis_speak('It is {} degree celcius in {}'.format(current_temperature, city))
        time.sleep(3)
    else:
        alexis_speak("city not found")
```

```
C:\python\python.exe C:/Users/Dell/PycharmProjects/iop/jarvis.py
How can I help you?
weather in London
It is 18.32 degree celcius in  London
```

☐ **Play music**: This will play songs from the systems library**.** It uses **OS(Miscellaneous operating system interfaces) library** in python. This library provides a portable way of using operating system dependent functionality.



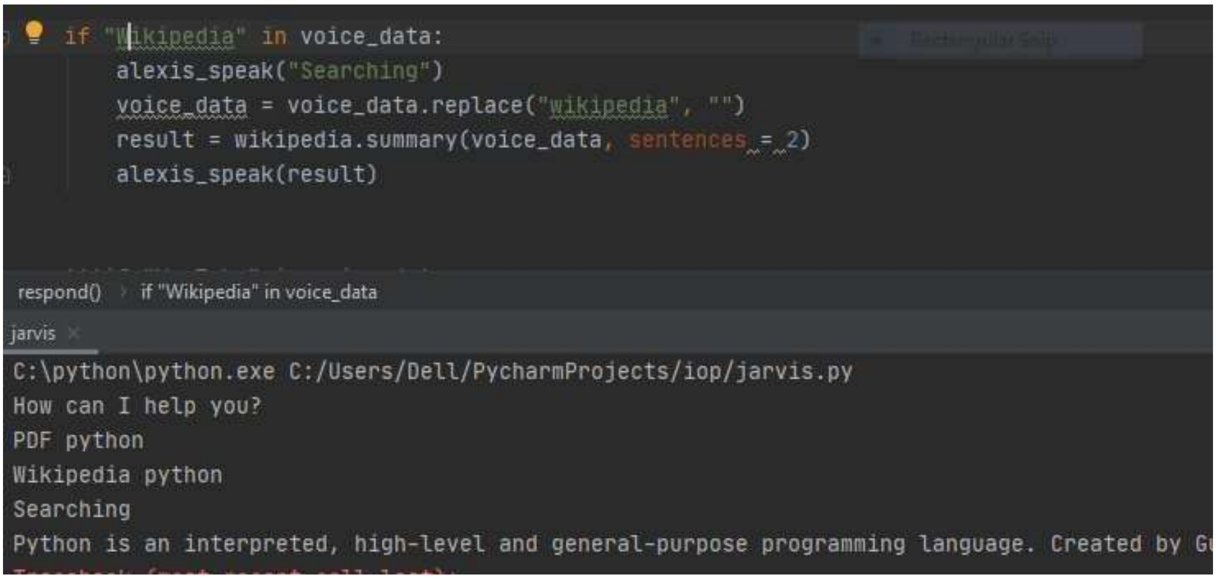
Nothing to show here. Try a different filter.

```
        # store the value corresponding
        # to the "temp" key of y
        current_temperature = y["temp"]
        alexis_speak('It is {} degree celcius in {}'.
        time.sleep(3)
    else:
        alexis_speak("city not found")


if "play songs" in voice_data:
    songs_dir = "F:\Punjabi music"
    songs =os.listdir(songs_dir)
    os.startfile(os.path.join(songs_dir, songs[0]))
```

```
C:\python\python.exe C:/Users/Dell/PycharmProjects/iop/ja
How can I help you?
play songs
```

□ **Crawl data:** Function to make the assistant crawl some Wikipedia data and read it for us. For this we can use Wikipedia api from python.



```
if "Wikipedia" in voice_data:
    alexis_speak("Searching")
    voice_data = voice_data.replace("wikipedia", "")
    result = wikipedia.summary(voice_data, sentences = 2)
    alexis_speak(result)
```

```
C:\python\python.exe C:/Users/Dell/PycharmProjects/iop/jarvis.py
How can I help you?
PDF python
Wikipedia python
Searching
Python is an interpreted, high-level and general-purpose programming language. Created by G
```

Output validation and comparison

The output of the project will be the required responses from the assistant when the user gives the command. As compared to other AI agents in market like siri, cortana, google assistant etc Our Alexis is a little slow in giving the output .

Given the fact that there are several failure points, theteststrategyforany AIsystem must be carefully structured to mitigate risk of failure. We mustfirstunderstandthevariousstages in an AIframework. With this understanding, we will be able to defineacomprehensive test strategy with specifictesting techniques across theentire framework. Here are four key AI use cases that must be tested to ensure proper AI system functioning:

□ Testing standalone cognitive features such as natural language processing (NLP), speech recognition, image recognition, and optical character recognition (OCR)

□ Testing AI platforms such as IBM Watson, Infosys NIA, Azure Machine Learning Studio, Microsoft Oxford, and Google DeepMind

□ Testing ML-based analytical models

□ Testing AI-powered solutions such as virtual assistants and robotic process automation (RPA)

Natural language processing (NLP)	• Testfor'precision' return of the keyword, i.e., a fractionof relevant instancesamong the totalretrieved instances of NLP • Test for'recall', i.e., a fraction of retrieved instances over total number of retrieved instances available • Testfor true positives (TPs), true negatives (TNs), false positives (FPs), and false negatives (FNs). Ensure that FPs and FNs are within the defined error/fallout range
Speech recognition inputs	• Conduct basic testing of thespeech recognition software to seeif thesystem recognizes speechinputs • Test for pattern recognition to determine if the system can identify when a unique phrase is repeated several times in aknownaccent and whetherit canidentify the same phrase when it is repeated in a different accent • Test deep learning, the ability to differentiate between 'New York' and 'Newark' • Test how speech translates to response. For example, a query of "Find me a place I can drink coffee" shouldnotgeneratearesponsewithcoffeeshopsanddrivingdirections. Instead, itshouldpointtoa public place or park where one can enjoy his/her coffee

Applications of Alexis

1. Our AI assistant can communicate effectively with the user through voice and also prints the commands and it's response for better understanding.
2. It can perform any search the user requests and hence is helpful for the user.
3. The user doesn't require to open email and then type long mails, all of this work will be done by Alexis the user just has to say what the subject and body should contain.
4. When the user wants to listen music she just needs to say it and the songs will be automatically played without requiring any physical effort from the user.
5. It has all the current condition about time date ,weather at any place, location of any place the user needs to ask about it and will get all the information needed.

So basically this Assistant will ease all the day to day task of the user and the user will have all the important information without making much effort.

Team work distribution

The Listening and responding to user commands, searching web, sending email , checking weather, searching location, play music, crawling data, greeting user date and time module and playing YouTube modules were done by the team leader.

All the documentation work was done by Vikash Kumar Gupta.

Knowledge enhancement:

We gained a lot of knowledge about AI and python libraries. An AI assistant revolves around two important concepts Natural Language Processing and speech recognition:

- **Natural Language Processing (NLP)** refers to AI method of communicating with an intelligent systems using a natural language such as English.

  Processing of Natural Language is required when you want an intelligent system like robot to perform as per your instructions, when you want to hear decision from a dialogue based clinical expert system, etc.

  The field of NLP involves making computers to perform useful tasks with the natural languages humans use. The input and output of an NLP system can be –

  - Speech
  - Written Text
- **Speech recognition** is a technology that is empowered by AI to add convenience to its users. This new technology has the power to convert voice messages to text. And it also has the ability to recognize an individual based on their voice command. Hence, this AI-powered speech recognition technology gained mammoth importance among tech giants such as Apple, Microsoft, Amazon, Google, Facebook, etc.

Other than these two concepts we also learned about many python libraries and their functions. Some of them are Explained below:

- **gTTS (*Google Text-to-Speech*)->**It is a Python library and CLI tool to interface with Google Translate's text-to-speech API. Writes spoken mp3 data to a file, a file-like object (bytestring) for further audio manipulation, or stdout . It features flexible pre-processing and tokenizing, as well as automatic retrieval of supported languages.
- **SeleniumLibrary** is a web testing library for Robot Framework that utilizes the Selenium tool internally. Selenium Python bindings provide a convenient API to access Selenium Web Driver like Firefox, Chrome, etc.

- 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.
- 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.

Other libraries used are Wikipedia, playsound, date and time, webbrowser etc.