**GOVERNMENT POLYTECHNIC,DHARMAVARAM**

Department of computer Engineering

GPT - 170, DMM

**INDUSTRIAL TRAINING**

**Scheme: C-20 Subject code: CM-601**



**DEPARTMENT OF COMPUTER ENGINEERING**

**GPT, Dharmavaram-515672**

**2022-2023**

**A PROJECT REPORT ON JARVIS: The Personal System Assistant.**

****

Submitted in the partial fulfillment of award of **Diploma**

In **Computer Science and Engineering**

Submitted To: Submitted By:

Mr. Balaswami Sir P. Chandu. (Leader)

1. Mourya.

S. Mohamad Mansoor.

U. Rohan Reddy.

1. Mahesh Babu.

R.V. Balaji Sai.

**CONTENT**

❖ Declaration.

❖ Certificate.

❖ Acknowledgement.

❖ Abstract.

❖ Problem Statement.

❖ Scope.

❖ Technologies Stack Used.

❖ Python Libraries.

* Data Flow.

❖ Functions.

❖ API (Application Programming Interface).

❖ System Testing.

❖ Features in Jarvis.

❖ Future Prospective.

❖ Conclusion.

**DECLARATION**

We do hereby declare that the report entitled **“Jarvis-Personal-Assistant”** submitted by us to Government Polytechnic College, Dharmavaram in partial of the requirement for the award of the course of Diploma in **COMPUTER SCIENCE AND ENGINEERING** is a record of bonafide project work carried out by us under the guidance of Ms. Madhavi and Department of Computer Science and Engineering.

Place: Dharmavaram P. Chandu.

Date: A. Mourya.

S. Mansoor.

U. Rohan Reddy.

B. Mahesh Babu.

R.V. Balaji Sai.

**CERTIFICATE**

This is to certify that the project entitled **“Jarvis: System Assistant by using Python”** is a bonafide work done by Mr. Chandu (20170-cm-038), Mr. Mourya (20170-cm-001), Mr. Mansoor (20170-cm-045), Mr. Rohan (20170-cm-049), Mr. Mahesh (20170-cm-056), and Mr. Balaji Sai (20170-cm-040) of 6th Semester Diploma in Computer Science and Engineering from **Government Polytechnic College**, **Dharmavaram** under the guidance of Mrs. Madhavi in the partial fulfillment of the requirement of the award for the Diploma in COMPUTER SCIENCE AND ENGINEERING in **Government Polytechnic College**, **Dharmavaram.**

**Project Guide:**

**Ms. Devineni Madhavi**

Date -

External: (Head of Department)

**ACKNOWLEDGEMENT**

We had a great experience working on this project and we got to learn Python Programming Language and various modules in python of new skills through this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them. We are highly indebted to **Ms. Devineni Madhavi** for their guidance and constant supervision as well as providing necessary information regarding the project and also for their support in completing the project. We would like to express our gratitude towards our parents and friends for their kind cooperation and encouragement which help us in the completion of the project.

Place – Dharmavaram. P. Chandu.

Date: A. Mourya.

S. Mansoor.

U. Rohan Reddy.

B. Mahesh Babu.

R.V. Balaji Sai.

**ABSTRACT**

As we know Python is an emerging language so it becomes easy to write a script for Voice Assistant in Python. The instructions for the assistant can be handled as per the requirement of user. Speech recognition is the process of converting speech into text. This is commonly used in voice assistants like Alexa, Siri, etc. In Python there is an API called Speech Recognition which allows us to convert speech into text. It was an interesting task to make my own assistant. It became easier to send emails without typing any word, Searching on Google without opening the browser, and performing many other daily tasks like playing music, opening your favorite IDE with the help of a single voice command. In the current scenario, advancement in technologies is such that they can perform any task with same effectiveness or can say more effectively than us. By making this project, I realized that the concept of AI in every field is decreasing human effort and saving time.

The project aims to develop a personal-assistant using Python. Jarvis draws its inspiration from virtual assistants like Cortana for Windows, and Siri for IOS. It has been designed to provide a **user-friendly interface** for carrying out a variety of tasks by employing certain **well-defined commands** . Users can interact with the assistant through **voice commands**.

As a personal assistant, Jarvis assists the end-user with *day-to-day activities* *like general human conversation, Searching from Web(Google, Youtube, Wikipedia), Automating Temperature, Automating Time, Set Alarm, Fully Automate Youtube Controls, Remember Function, Playing Songs, Latest News Updates, Calculator, Whatsapp Function, Password Protection, Schedule Your Day Function, Open any App (EASY METHOD), Live IPL Score Function, GUI OF JARVIS, Rock Paper Scissor Game, Screenshot And Camera Function, THE FOCUS MODE, and Google Translator.*

**PROBLEM STATEMENT**

We are all well aware about Siri, Google Assistant and many other virtual assistants which are designed to aid the tasks of users in Android and IOS platforms. But to our surprise, there’s no such virtual assistant available for the paradise of Developers i.e. Windows platform. In rapid change of day to day life users are comfortable with automating their tasks without doing an external works.

**PURPOSE**

This Software aims at developing a personal assistant for Windows-based systems. The main purpose of the software is to perform the tasks of the user at certain commands, provided in either of the way during using, speech. It will ease most of the work of the user as a complete task can be done on a single command. Jarvis draws its inspiration from Virtual assistants like Siri for IOS, Goggle for Android. Users can interact with the assistant either through voice commands or keyboard input.

**PRODUCT GOALS AND OBJECTIVES**

Currently, the project aims to provide the Windows Users with a Virtual Assistant that would not only aid in their daily routine tasks like general human conversation, Searching from Web (Google, Youtube, and Wikipedia), Automating Temperature, Automating Time, Set Alarm, Fully Automate Youtube Controls, Remember Function and many others but also help in automation of various activities.

In the long run, we aim to develop a complete server assistant, by automating the entire server management process - deployment, backups, auto-scaling, logging, monitoring and make it smart enough to act as a replacement for a general server administration.

**SCOPE**

Presently, Jarvis is being developed as an automation tool and virtual assistant based on python programming language.

Among the Various roles played by Jarvis are:

1. General Human Conversation.

2. Searching from Web (Google, Youtube and Wikipedia).

3. Automating Temperature.

4. Automating Time.

5. Set Alarm.

6. Fully Automate Youtube Controls.

7. Remember Function.

8. Playing Songs.

9. Latest News Updates.

10. Calculator.

11. Whatsapp Function.

12. Schedule Your Day Function.

13. Open any App (EASY METHOD).

14. Live IPL Score Function.

15. Rock Paper Scissor Game.

16. Screenshot and Camera Function.

17. THE FOCUS MODE.

18. Google Translator.

**TECHNOLOGY STACKS USED**

**TECHNOLOGIES USED MAIN PROGRAM**

* Python Programming Language

**Modules Used.**

* Tkinter
* API (Application Programming Interface.)

**THIRD PARTY MODULES.**

* Pyttsx3
* Speech Recognition… Etc…

**PYTHON LIBRARIES**

In JARVIS following python libraries were used:

1. **pyttsx3 :**

It is a python library which converts text to speech.

1. **Speech Recognition:**

It is a python module which converts speech to text.

1. **pywhatkit:**

It is python library to send WhatsApp message at a particular time with some additional features.

1. **Numpy(title()):**

It is function returns element-wise title cased version of string or Unicode.

1. **Random :**

Python Random module is an in-built module of Python which is used to generate random numbers.

1. **Datetime:**

This library provides us the actual date and time.

1. **Pygame :**

Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.

1. **Wikipedia:**

It is a python module for searching anything on Wikipedia.

1. **Smtplib :**

Simple mail transfer protocol that allows us to send mails and to route mails between mail servers.

**10. Pyjokes:**

It is a python libraries which contains lots of interesting jokes in it.

**11. Webbrowser:**

It provides interface for displaying web-based documents to users.

**12. Pyautogui:**

It is a python libraries for graphical user interface.

**13. requests:**

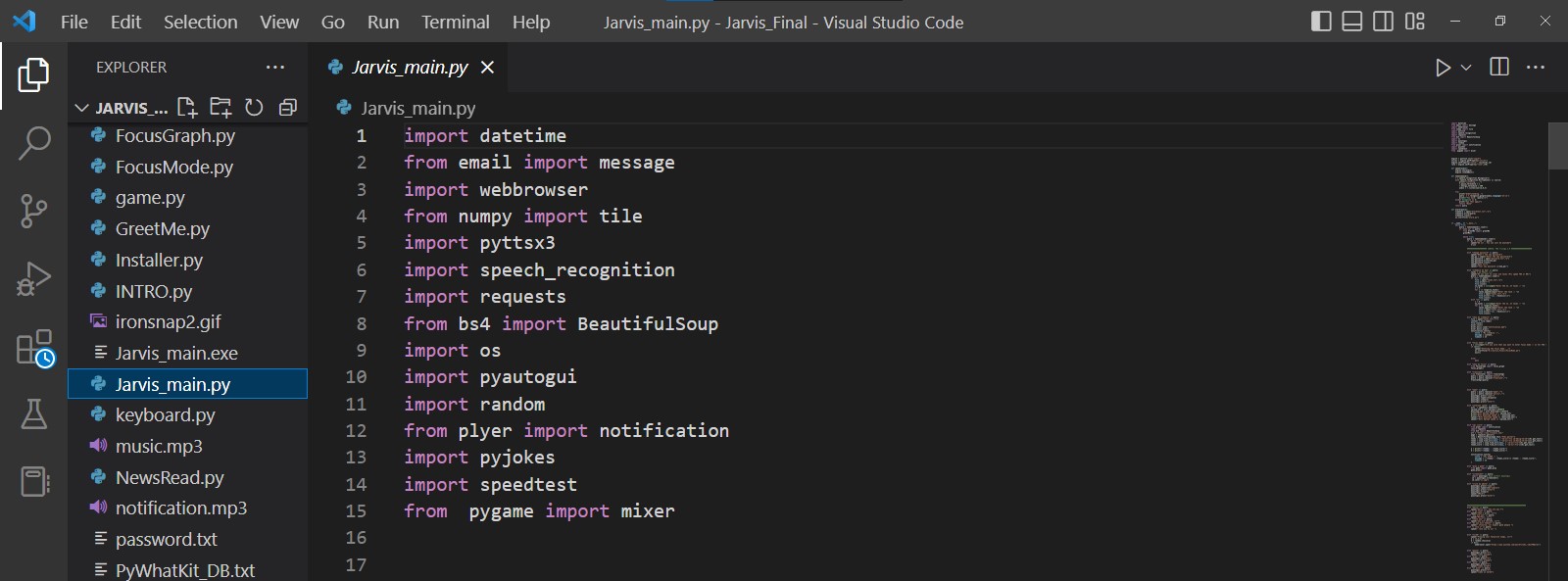
This is an elegant and simple HTTP library for Python that allows you to send HTTP/1.1 requests extremely easily.

**14. os:**

It represents Operating System related functionality.

**15. sys:**

It allows operating on the interpreter as it provides access to the variables and functions that usually interact strongly with the interpreter.



**DATA FLOW**

The Data Flow for JARVIS is as follow

Data Flow for JARVIS

The system is designed using the concept of Artificial Intelligence and with the help of necessary packages of Python. Python provides many libraries and packages to perform the tasks, for example open the Google, Search in YouTube… etc.

The data in this project is nothing but user input, whatever the user says, the assistant performs the task accordingly. The user input is nothing specific but the list of tasks which a user wants to get performed in human language i.e. English.

**FUNCTIONS**

1. **Speak ():**

The speak function will be responsible for speaking whatever text is passed to it.

1. **takeCommand ():**

The mction is used to take the command as input through microphone of user and returns the outstring.

1. **wishMe():**

This function greets the user according to the time like Good Morning, Good afternoon and Good Evening.

1. **taskExecution():**

This is the function which contains all the necessary task execution definition like searchGoogle(), searchYouTube(), searchWikipedia(), latetstnews(), Calculator(), sendMessage(), alarm()..Etc.

**API (Application Programming Interface).**

Anyone who works with business software has heard terms like “API” and “web hooks” thrown around. In this article, we’ll not only cover what APIs are, but we’ll also answer the question: “How do APIs work?” Understanding the function of APIs is the key to understanding how they can help business users in any role accomplish more, faster - without necessarily having to learn coding.

API tools have fundamentally transformed how developers write applications. They have introduced an entirely new vertical of “platform as a service” software companies. API-based tools are the reason why data integrations between essential business software are possible. In fact, API integrations have become essential to operations and revenue professionals.

**How Do APIs Work 2**

API stands for “application programming interface.” An API is essentially a set of rules that dictate how two machines talk to each other. Some examples of API-based interactions include a cloud application communicating with a server, servers pinging each other, or applications interacting with an operating system. Whenever you use an app on your phone or computer or log onto Twitter or Face book, you’re interacting with several different APIs behind the scenes. Nearly all businesses that use any kind of modern technology use APIs at some level to retrieve data or interact with a database for customers to use.

An API’s defined communication protocol is what enables developers to build, connect, and integrate applications quickly and at scale. Consider, as an example, Jeff Bezos' famously-issued 2002 mandate. Amazon's change of direction shows how APIs helped it move faster than its competitors, and is reportedly the reason Amazon is so successful. Bezos ordered all of his teams to communicate and expose data and functionality through service interfaces, that is, APIs. Once the APIs and infrastructure were in place, Amazon’s teams were able to operate much more efficiently. Launching this new infrastructure enabled the creation of Amazon Web Services, which has since become Amazon’s largest revenue driver.

**How do APIs work? And how do they help you do more every day?**

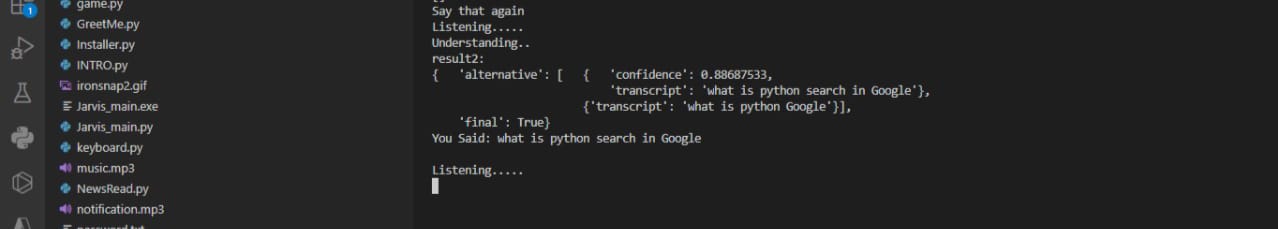
An application programming interface is a set of rules that define how computers, applications, or machines can talk to each other. You can think of it this way: the typical user interface is intended for use by a human being, while APIs are intended for use by an application or computer.

**SYSTEM TESTING**

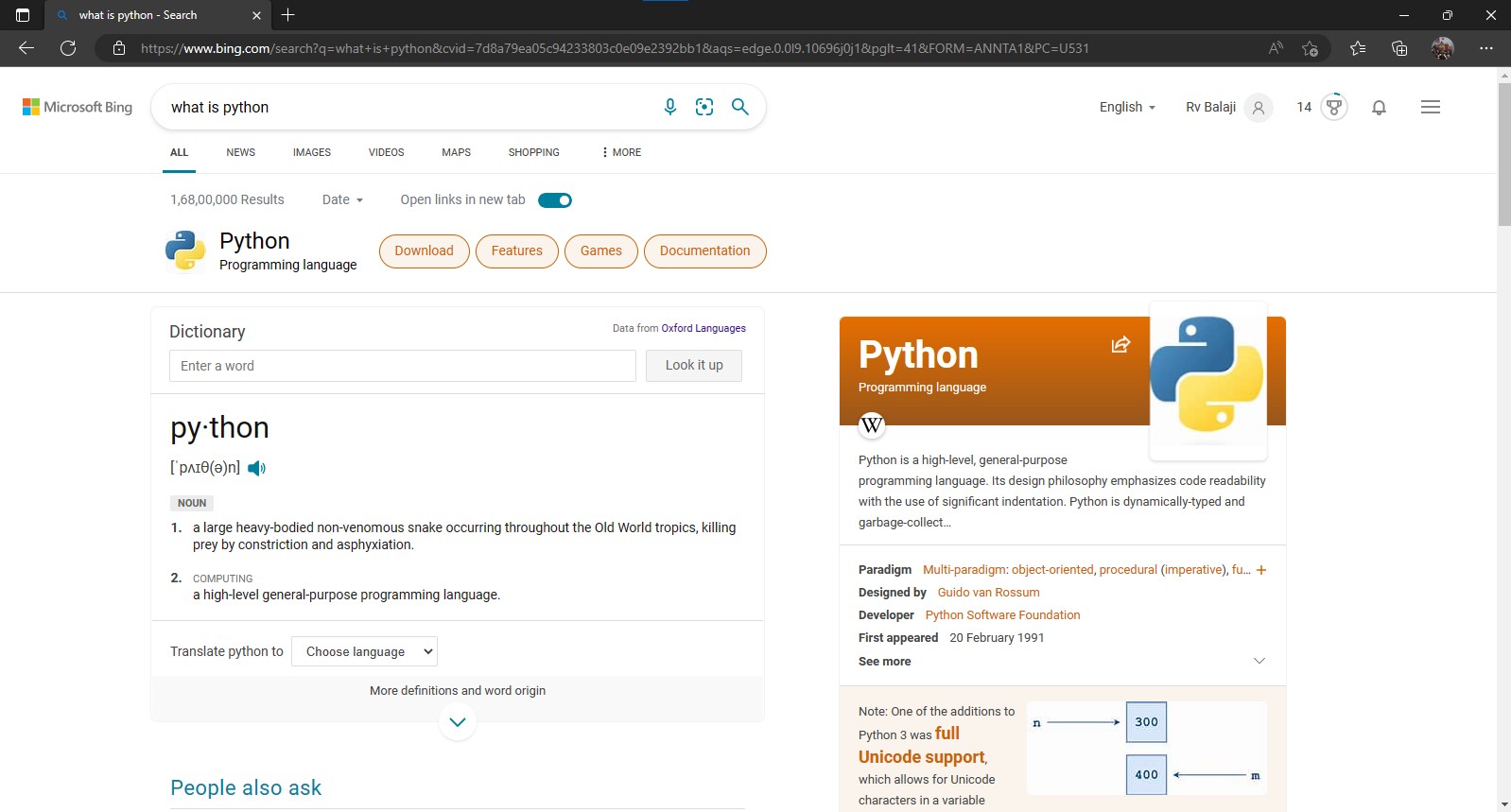
The system testing is done on fully integrated system to check whether the requirements are matching or not. The system testing for JARVIS desktop assistant focuses on the following four parameters:

**FUNCTIONALITY**

In this we check the functionality of the system whether the system performs the task which it was intended to do. To check the functionality each function was checked and run, if it is able to execute the required task correctly then the system passes in that particular functionality test. For example to check whether JARVIS can search on Google or not, as we can see in the figure 1, user said "What is Python", Jarvis open Google and searched for the required input.



**Fig 1.1 Input through Voice Commands.**



**Fig 1.2 Output**

**USABILITY**

Usability of a system is checked by measuring the easiness of the software and how user friendly it is for the user to use, how it responses to each query that is being asked by the user.

It makes it easier to complete any task as it automatically do it by using the essential module or libraries of Python, in a conversational interaction way. Hence any user when instruct any task to it, they feel like giving task to a human assistant because of the **conversational interaction** for giving input and getting the desired output in the form of task done.

The desktop assistant is **reactive** which means it know human language very well and understand the context that is provided by the user and gives response in the same way, i.e. human understandable language, English. So user finds its reaction in an informed and smart way.

The main application of it can be its **multitasking** ability. It can ask for continuous instruction one after other until the user "QUIT" it. It asks for the instruction and listen the response that is given by user without needing any trigger phase and then only executes the task.

**SECUITRY**

The security testing mainly focuses on vulnerabilities and risks. As JARVIS is a local desktop application, hence there is no risk of data breaching through remote access. The software is dedicated to a specific system so when the user logs in, it will be activated.

**STABILITY**

Stability of a system depends upon the output of the system, if the output is bounded and specific to the bounded input then the system is said to be stable. If the system works on all the poles of functionality then it is stable.

**FEATURESOF JARVIS**

1. **Queries from the web:**

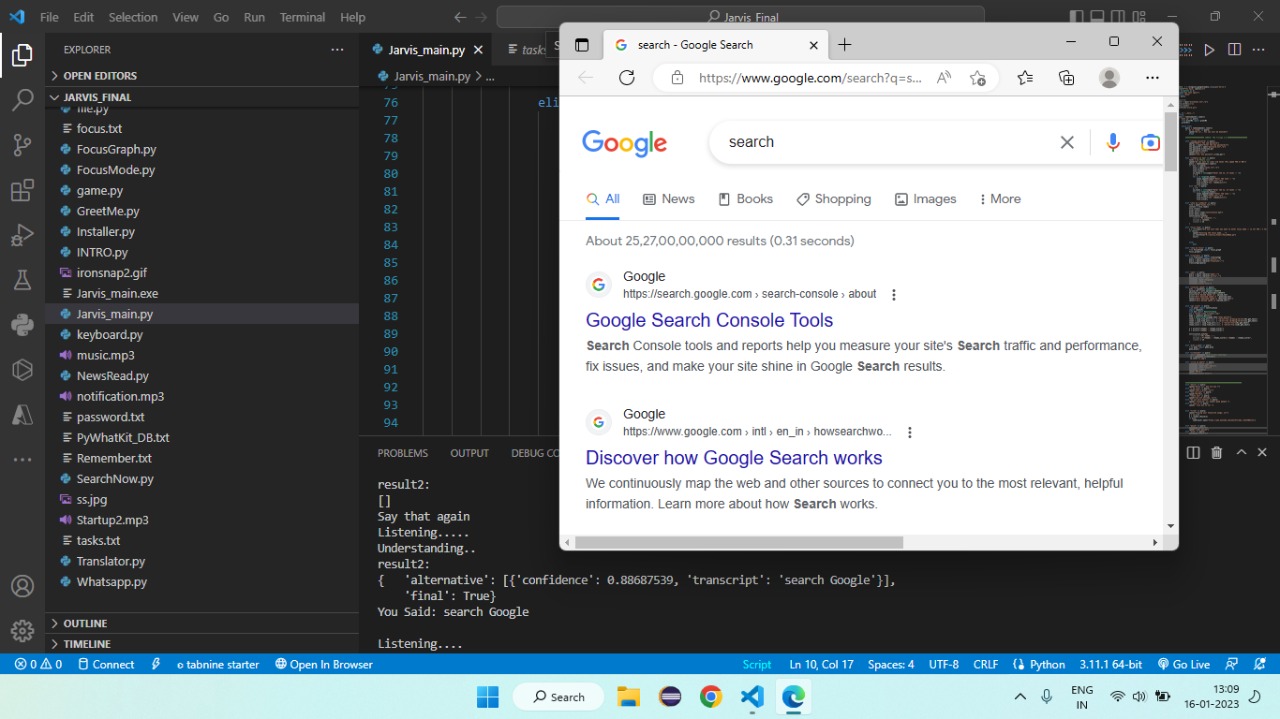
Making queries is an essential part of one’s life, and nothing changes even for a developer working on Windows. We have addressed the essential part of a netizen’s life by enabling our voice assistant to search the web. The result from the web as well as displaying it to the user. Jarvis supports search engines like Google, Bing and Yahoo and displays the result by scraping the searched queries.

In order to make queries from different search engines, the given format should be adopted:

< *content to be search> <query>*

Jarvis supports Google, Bing and Yahoo, which should precede the desired query.

**OUTPUT:**



1. **Accessing YouTube videos:**

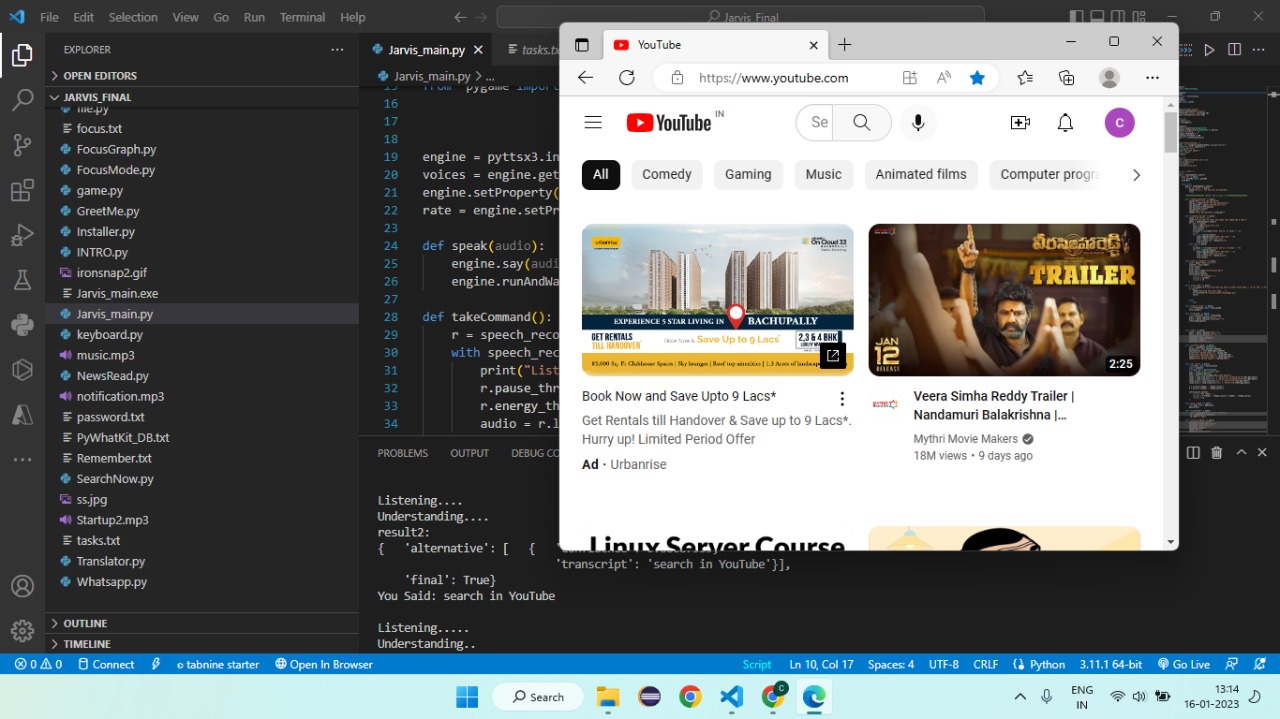
Videos have remained as a main source of entertainment, one of the most prioritized tasks of virtual assistants. They are equally important for

Entertainment as well as educational purposes as most teaching and research activities in present times are done through Youtube. This helps in making the learning process more practical and out of the four walls of the classroom.

In order to access videos from YouTube format is:

***Search in youtube*** *<query>*

**OUTPUT:**



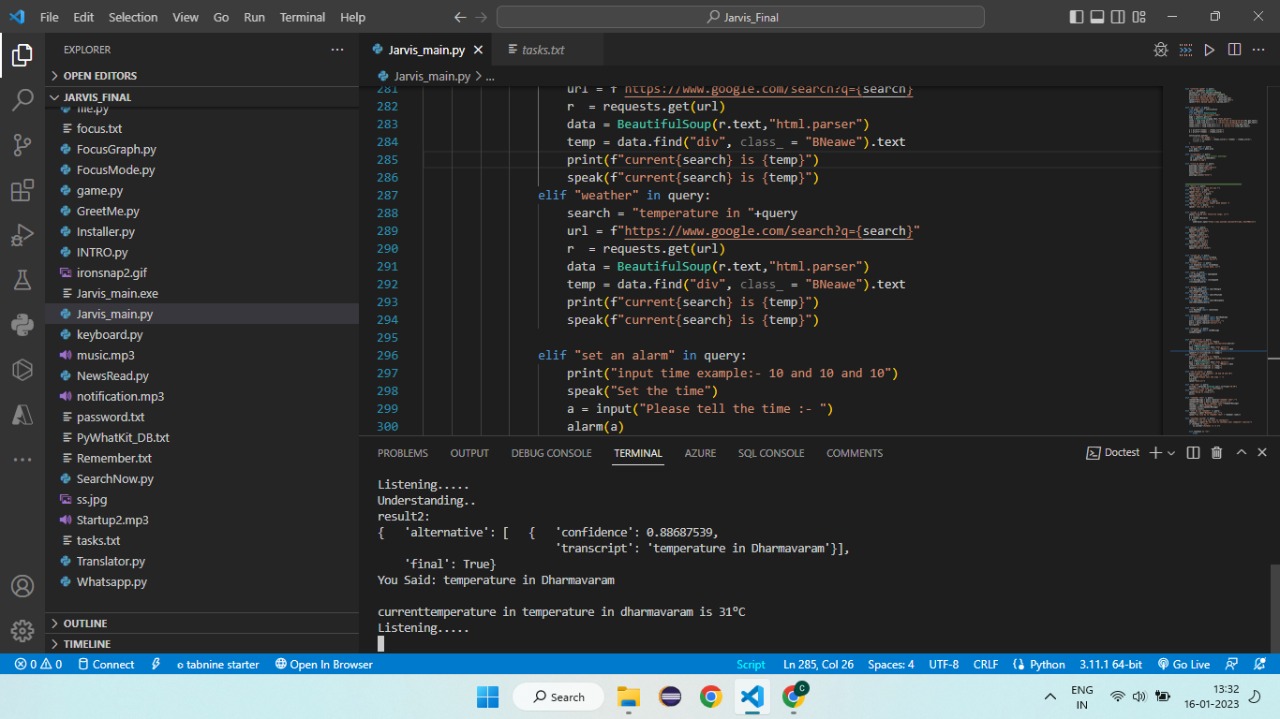
1. **Get weather and Temperature for a location:**

Getting live weather conditions about a place remains an important task of virtual assistants. We can get the weather or temperature of a particular locality by just using a voice command.

In order to access the live weather condition format is:

***Weather or Temperature in*** *<city>*

**OUTPUT:**

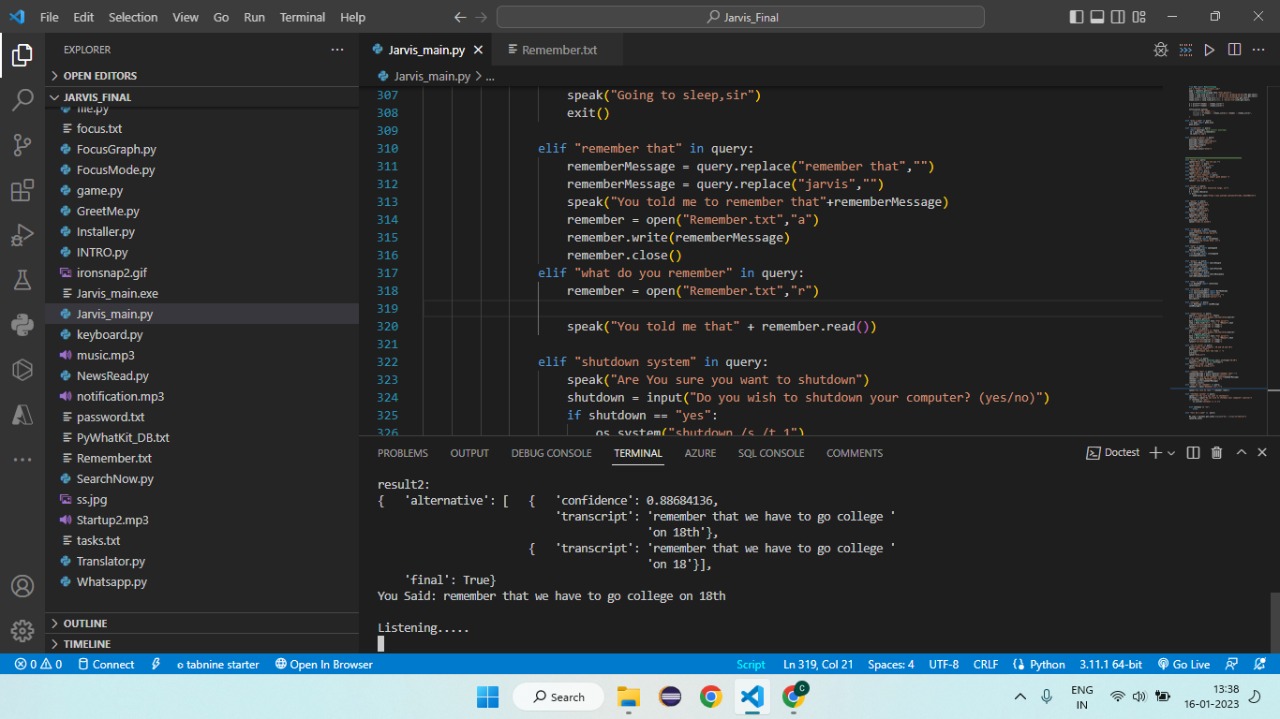


1. **Set Reminders:**

One of the main features of a voice assistant is to set a reminder for the user accordingly. Jarvis is no different when it comes to this. The user can set reminders to be notified about a task at a particular time. This will help users, especially developers to schedule their time and resources easily. All the user has to do is to input ***Set reminder*** to the assistant. A form will be displayed. Fill the form with the required details and click on **set reminder** button.

***Remember that*** *<content to be reminded>*

**OUTPUT:**

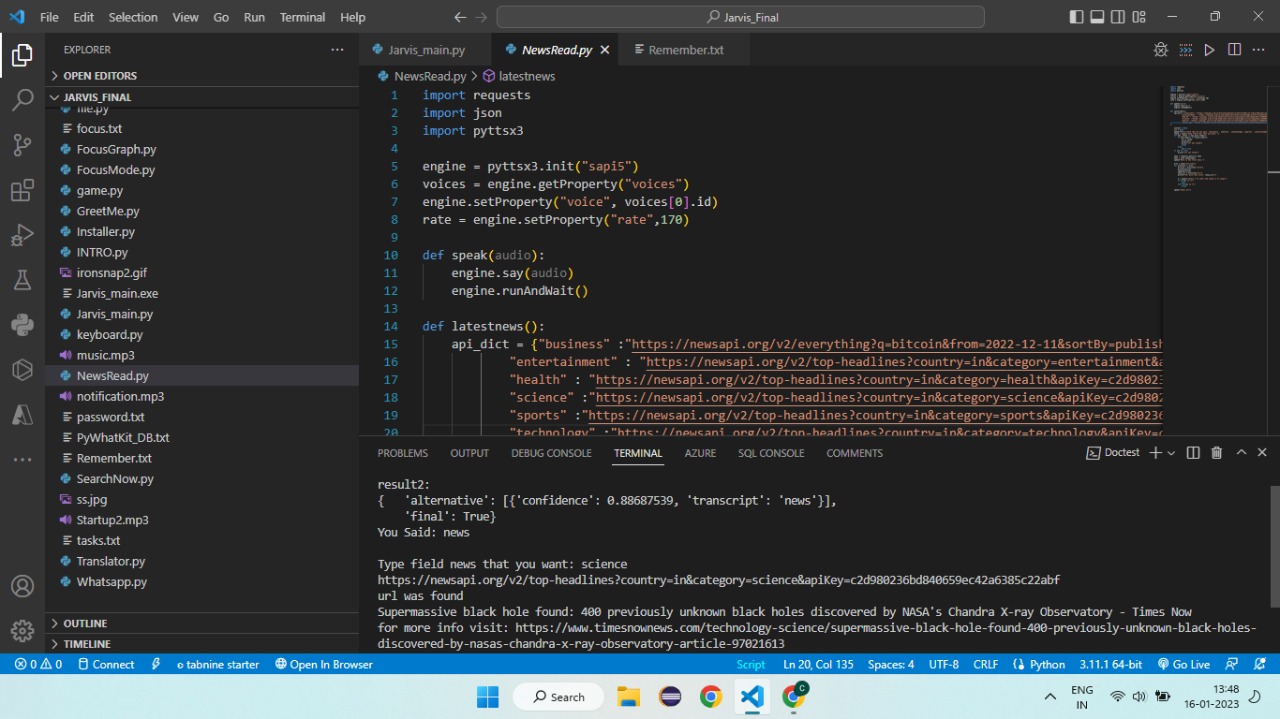


1. **News Function:**

News is an important thing for every person in day to day life. In our Jarvis program we can retrieve the information in fields like business, entrainment, health, science, sports, technology etc….

***News****<felid to be search>*

**OUTPUT:**

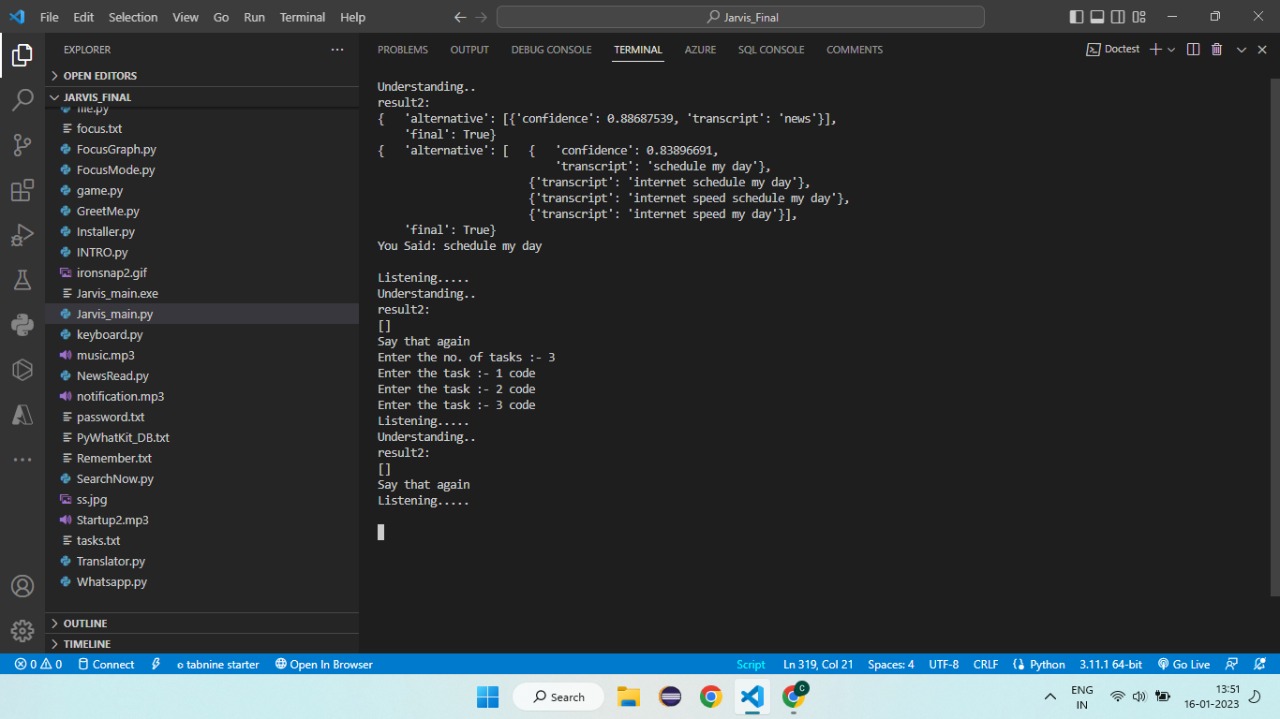
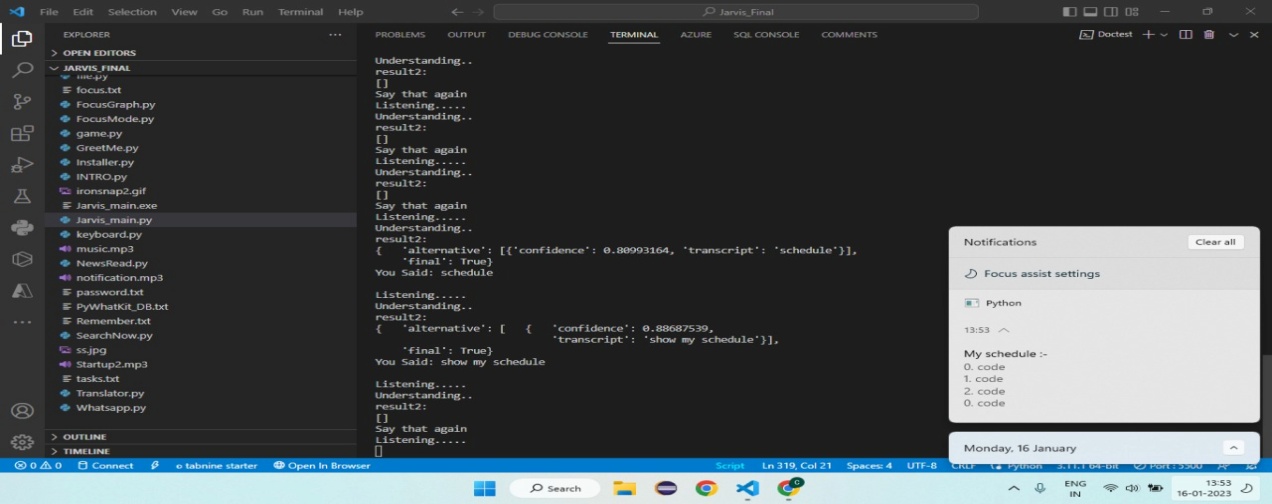


1. **Schedule my day:**

We can schedule our day by using command without any hustle. By scheduling our day with proper plan we can able to do the works efficiently in a proper way and we can retrieve the tasks by using the show my schedule.

***Schedule my day*** *<no. of tasks>*

**OUTPUT:**

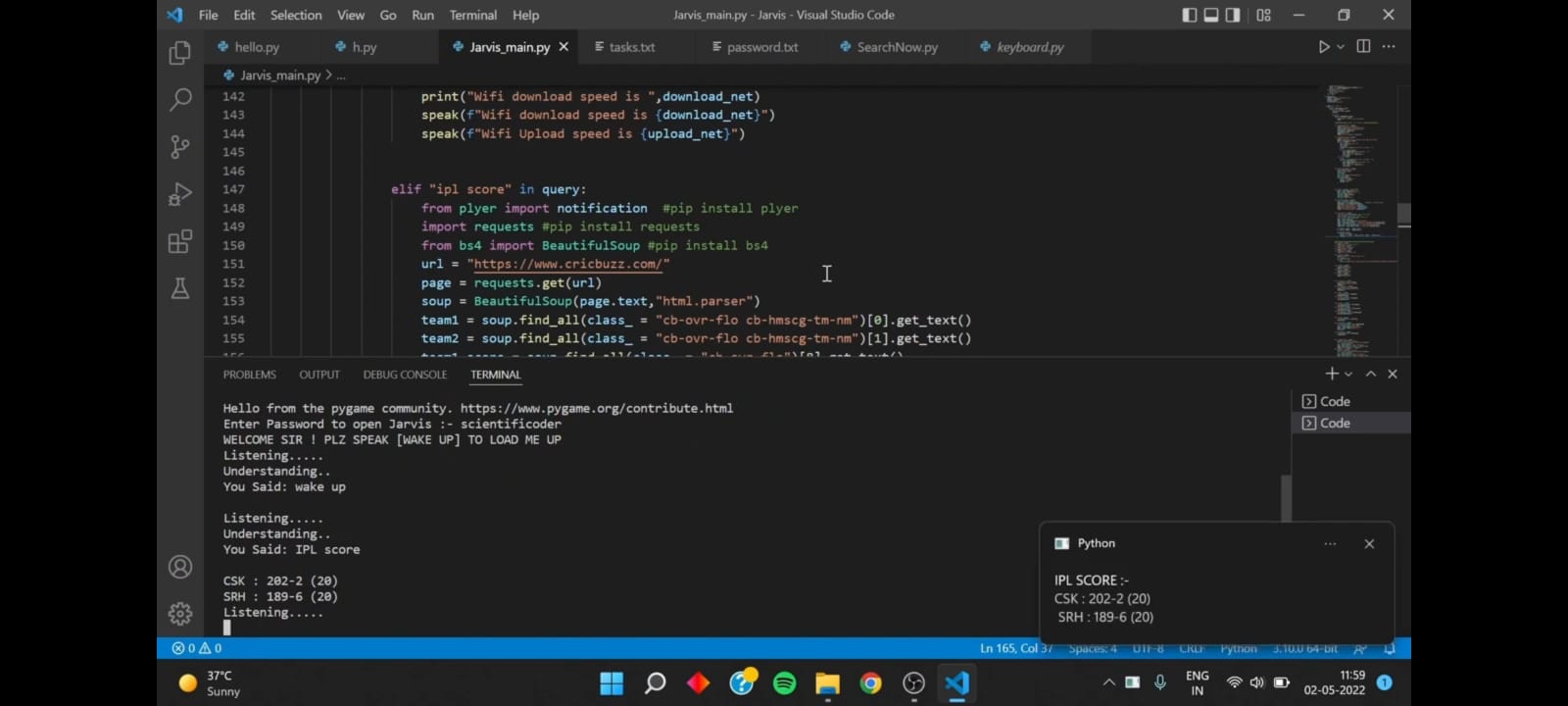
**

1. **Cricket Score:**

This feature of Jarvis gives the live score of the current match’s scores. We can simple get the score of the cricket match’s by using simple voice commands this feature saves the lot of time for cricket lovers.

*<cricketscore>****Command is used for retrial of the current cricket score***

**OUTPUT:**

**

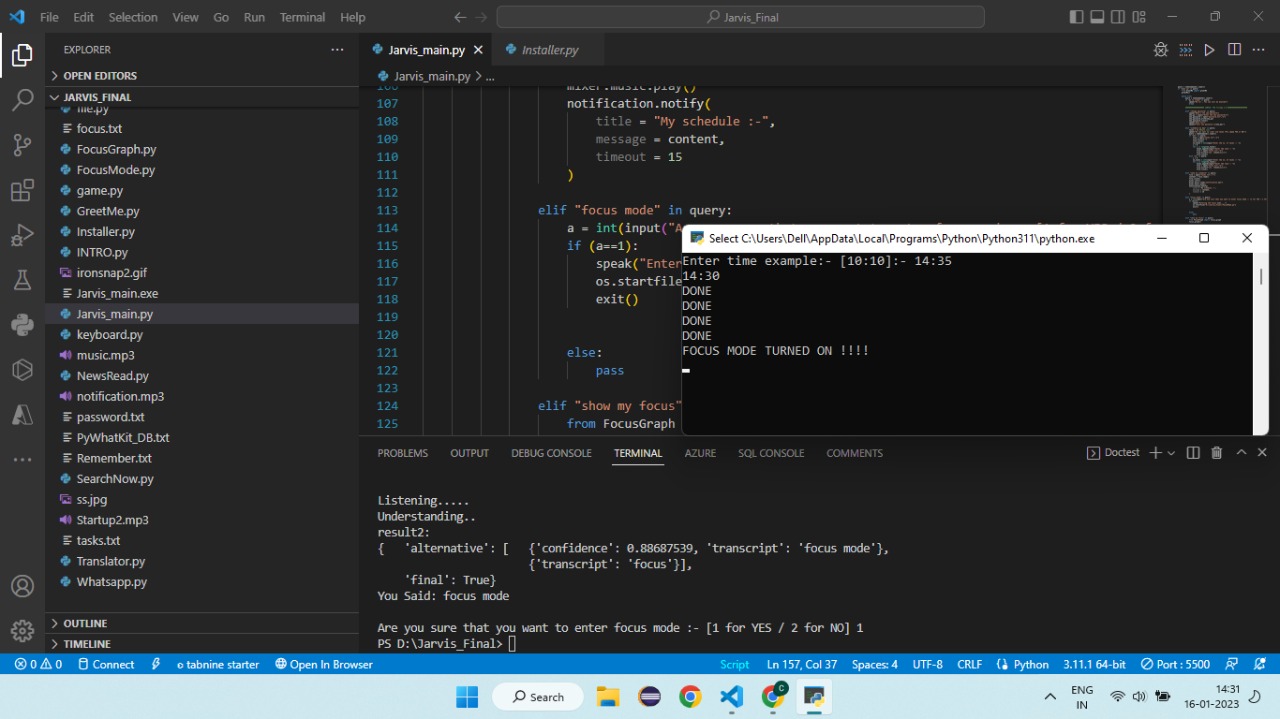
1. **Focus Mode and Graph:**

This is the most useful feature of our Jarvis. This feature will help to concentrate on the work. In todays sincere more students are wasting lot of time on social media platforms like instagram, Facebook etc... .

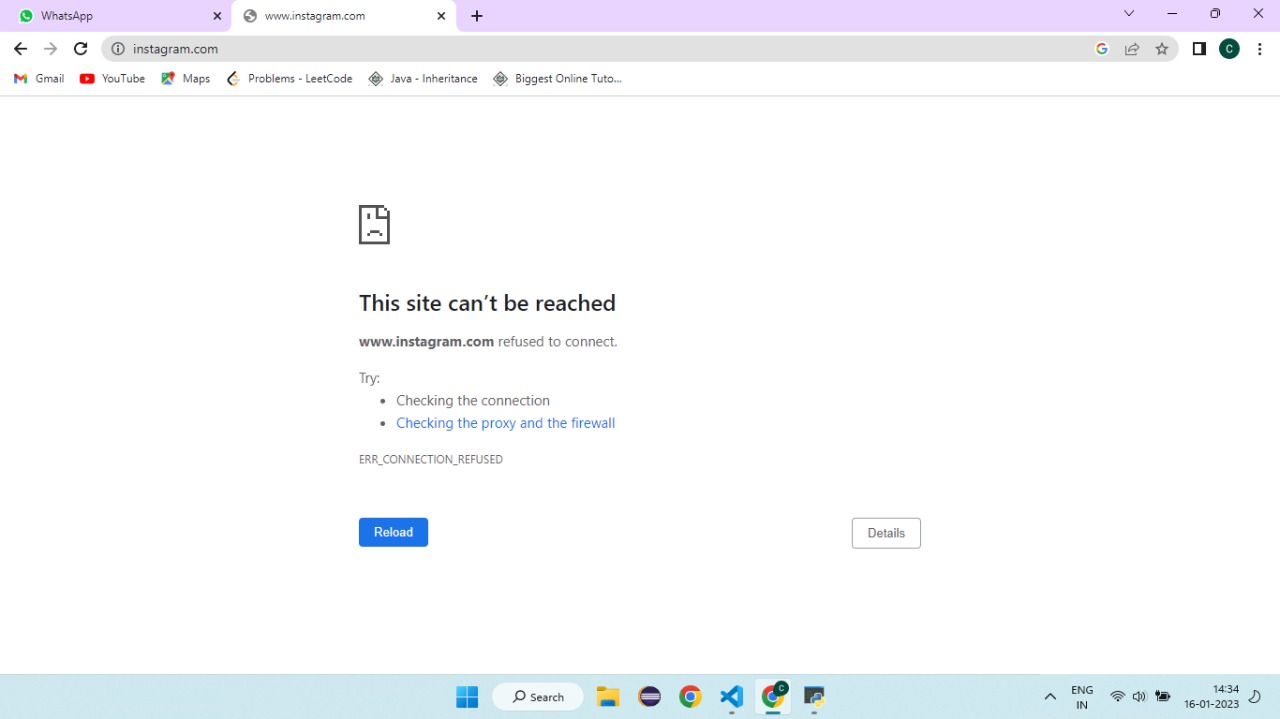
By using Focus mode we can control the user activities by blocking the third party apps like instagram, facebook. Also we can view our progress of focus by invoking show my focus method. It will represent focus graph.

*<Focus mode>* ***Command is used for the invoking of focus mode method we have to give the amount of time to focus***

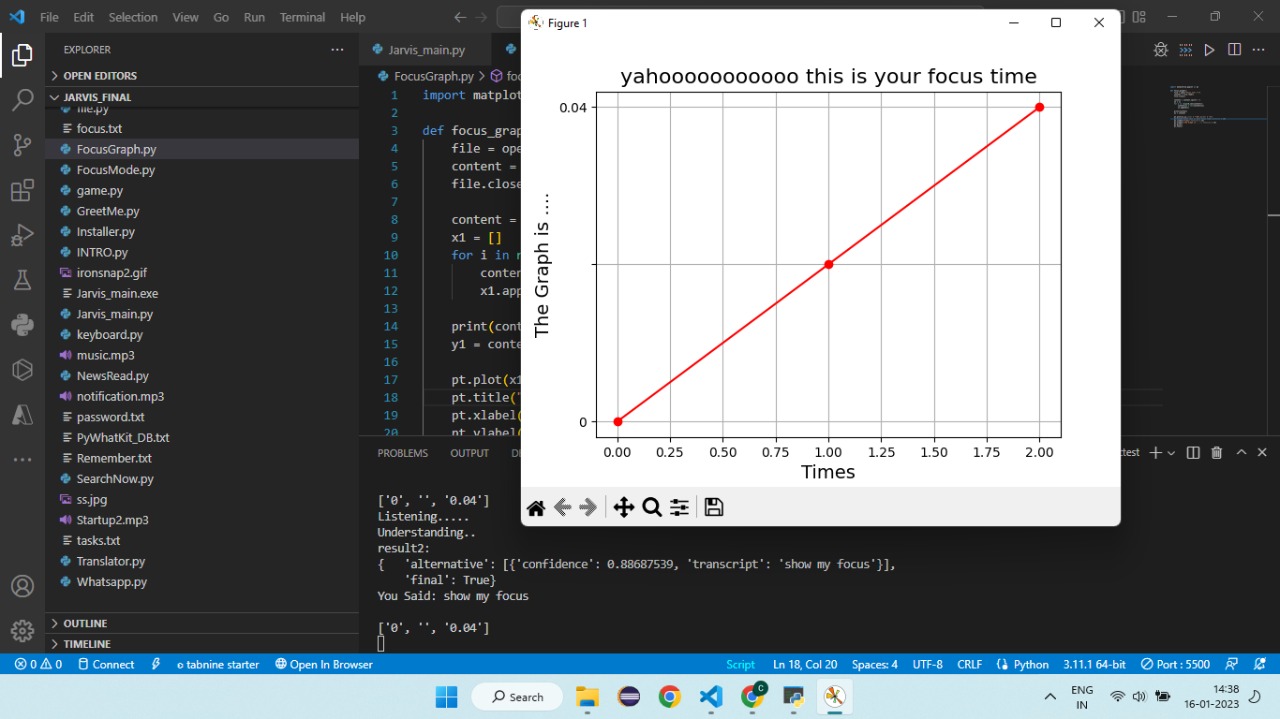
**OUTPUT:**



Once the focus mode is turn on up to the end of session of focus mode. We could not open the sites.



After the completion of sessions all social media sites starts working. By using show my focus mode command we can view our focus graph like below.

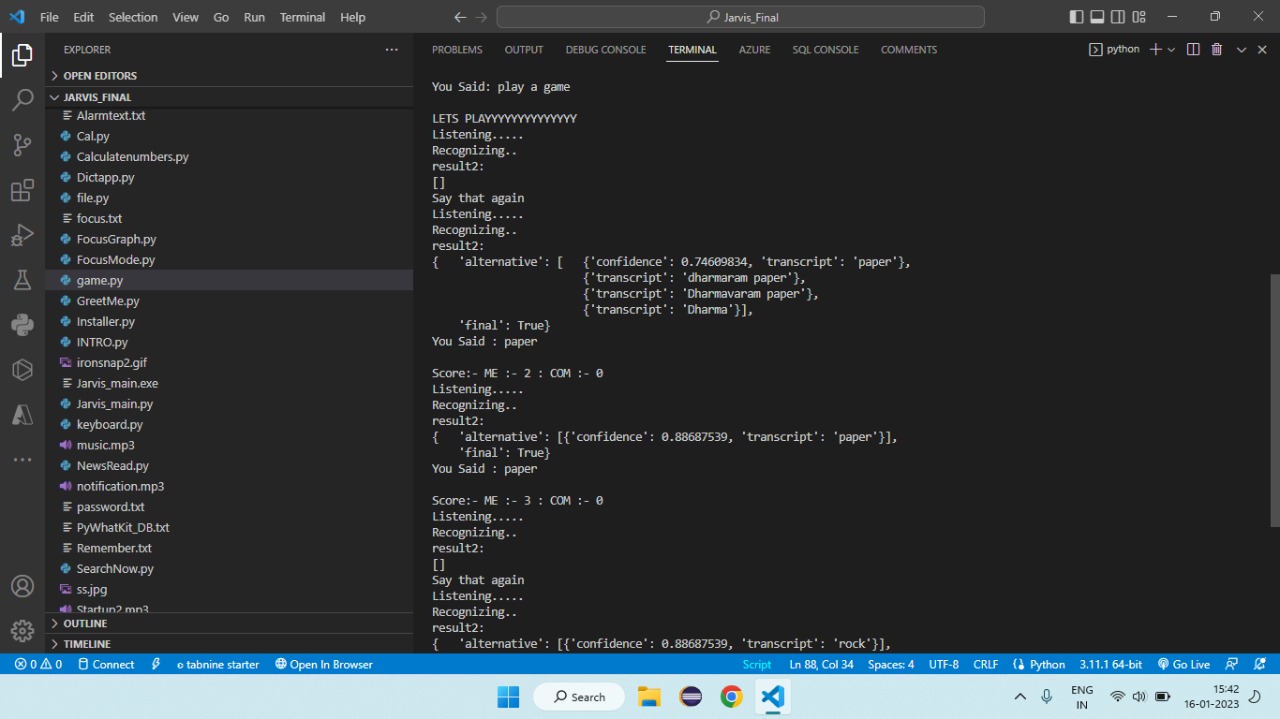


1. Game:

The rock is a closed fist, paper is a flat hand with fingers and thumb extended and the palm facing downward, and scissors is a fist with the index and middle fingers fully extended toward the opposing player. Rock wins against scissors, paper wins against rock, and scissors wins against paper. We design a simple Rock paper Scissor game to reduce the stress of work.

*<Playagame>* ***Command is used for invoke the game (R.P.S)***

**OUTPUT:**

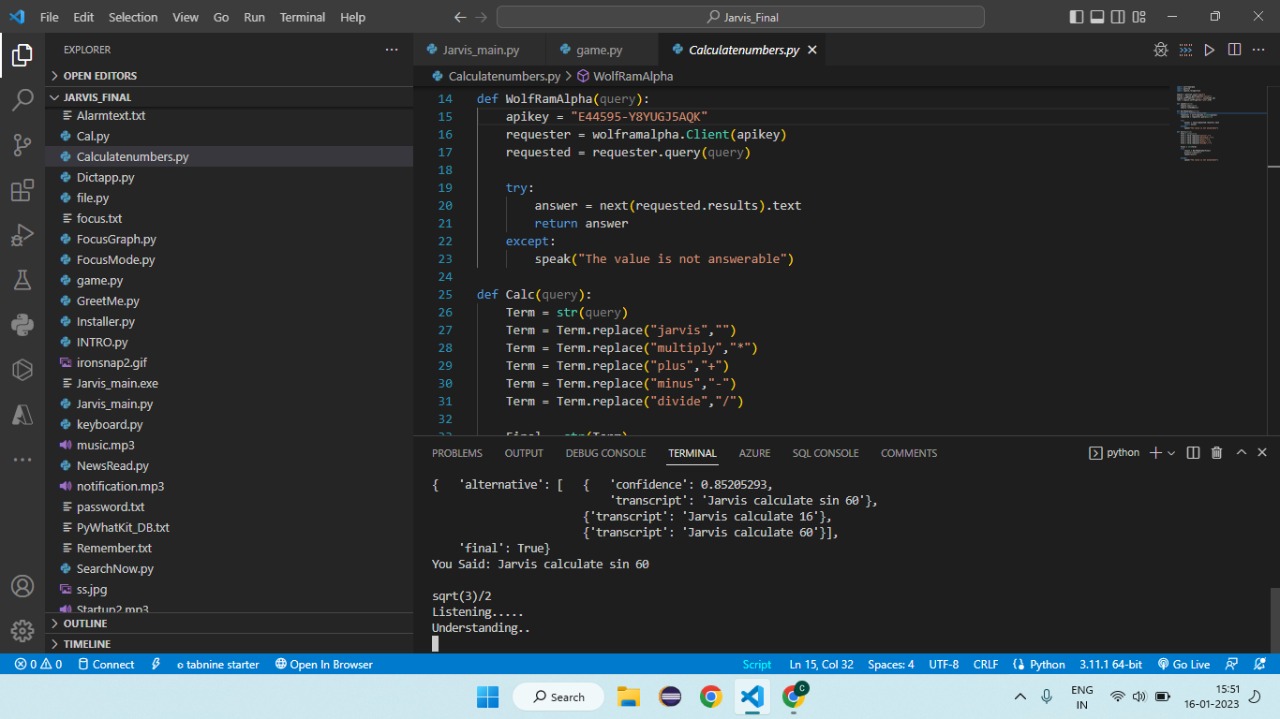


1. Calculator :

Calculator is used to perform calculations ranging from basic arithmetic to complex mathematics and scientific calculations also we can directly give commands to Jarvis to compute the equations in mathematics as well as scientific calculations.

*<Calcute>****Equations to be calculated***

**OUTPUT:**

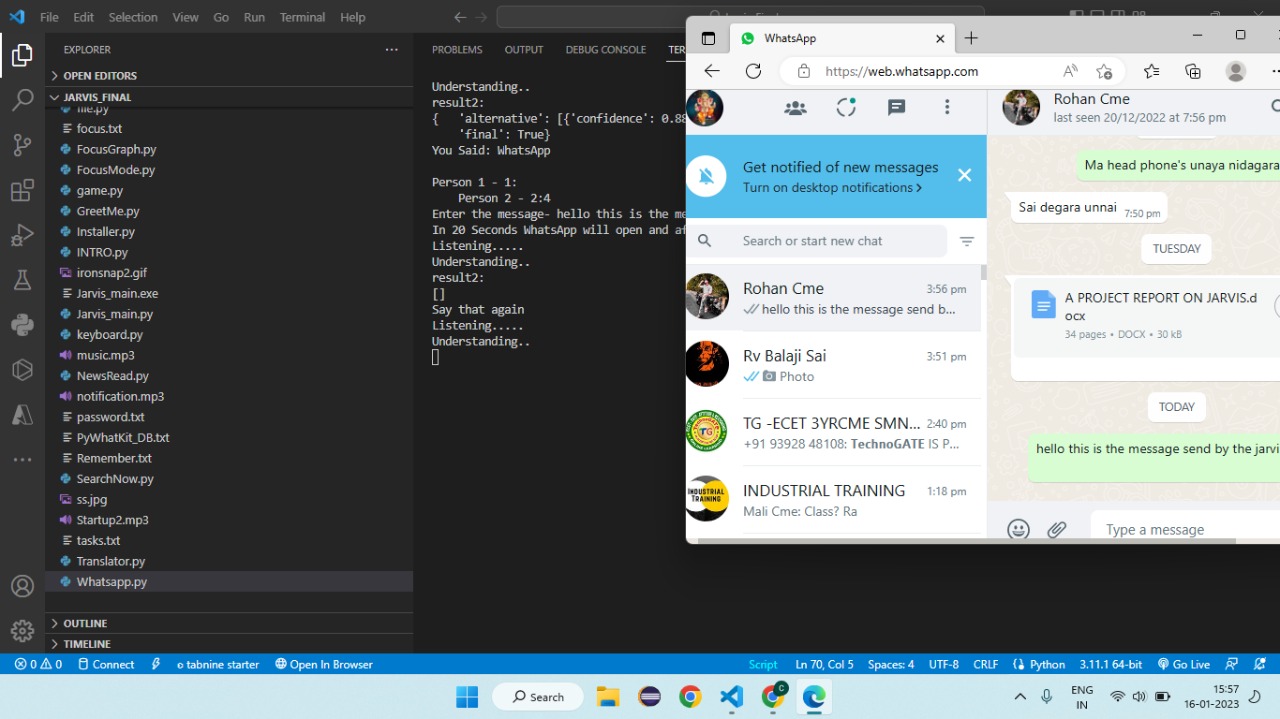


1. Whatsapp:

Whatsapp messenger is used for communication purpose nowadays it is more widely used for communicating to the people in our project we can simply send the messages to the person whom we want to communicate by just giving voice commands. It is quite simple and easy to give instructions through the voice rather than typing the messages.

*<whatsapp>* ***command is used for the call the whatsapp module***

**OUTPUT:**

******

**MAIN PROGRAM OF JARVIS**

By the increasing complexity of day to day life in technological era there is a huge need of automate the things in this program we tried to automate the few things that a common person using the things in daily life. We are all aware of popular A.I like Siri (Apple), Google Assistant (Android) by the inspiration of them we tried to automate the things used by the common person. Speech is an effective and natural way for people to interact with applications, complementing or even replacing the use of mice, keyboards, controllers, and gestures.

A hands-free, yet accurate way to communicate with applications, speech lets people be productive and stay informed in a variety of situations where other interfaces will not. Speech recognition is a topic that is very useful in many applications and environments in our daily life. Generally speech recognizer is a machine which understands humans and their spoken word in some way and can act thereafter. A different aspect of speech recognition is to facilitate for people with functional disability or other kinds of handicap. To make their daily chores easier, voice control could be helpful.

Source Code of JARVIS

import datetime

from email import message

import webbrowser

from numpy import tile

import pyttsx3

import speech\_recognition

import requests

from bs4 import BeautifulSoup

import os

import pyautogui

import random

from plyer import notification

import pyjokes

import speedtest

from pygame import mixer

engine = pyttsx3.init ("sapi5")

voices = engine.getProperty("voices")

engine.setProperty("voice", voices [0].id)

rate = engine.setProperty ("rate", 170)

def speak (audio):

engine.say(audio)

engine.runAndWait ()

def takeCommand ():

r = speech\_recognition.Recognizer ()

with speech\_recognition.Microphone () as source:

print ("Listening.....")

r.pause\_threshold = 1

r.energy\_threshold = 300

audio = r.listen (source, 0, 4)

try:

print ("Understanding...”)

query = r.recognize\_google (audio, language='en-in')

print (f"You Said: {query}\n")

except Exception as e:

print ("Say that again")

return "None".

return query

def alarm (query):

timehere = open ("Alarmtext.txt","a")

timehere.write (query)

timehere.close ()

os.startfile ("alarm.py")

if \_name\_ == "\_main\_":

while True:

query = takeCommand ().lower ()

if "wake up" in query:

from GreetMe import greetMe

greetMe ()

while True:

query = takeCommand ().lower ()

if "go to sleep" in query:

speak ("Ok sir, you can call me anytime")

break

elif "change password" in query:

speak ("What's the new password")

new\_pw = input ("Enter the new password\n")

new\_password = open ("password.txt","w")

new\_password.write (new\_pw)

new\_password.close ()

speak ("Done sir")

speak (f"Your new password is {new\_pw}")

elif "schedule my day" in query:

tasks = [] #Empty list

speak ("Do you want to clear old tasks (Plz speak YES or NO)")

query = takeCommand ().lower ()

if "yes" in query:

file = open ("tasks.txt","w")

file.write (f"")

file.close ()

no\_tasks = int (input ("Enter the no. of tasks: - "))

i = 0

for i in range (no\_tasks):

tasks.append (input ("Enter the task: - "))

file = open ("tasks.txt","a")

file.write (f"{i}. {tasks[i]}\n")

file.close ()

elif "no" in query:

i = 0

no\_tasks = int (input ("Enter the no. of tasks: - "))

for i in range(no\_tasks):

tasks.append (input ("Enter the task: - "))

file = open ("tasks.txt","a")

file.write (f"{i}. {tasks[i]}\n")

file.close ()

elif "show my schedule" in query:

file = open ("tasks.txt","r")

content = file.read()

file.close ()

mixer.init ()

mixer.music.load ("notification.mp3")

mixer.music.play ()

notification.notify (

title = "My schedule:-",

message = content,

timeout = 15 )

elif "focus mode" in query:

a = int (input ("Are you sure that you want to enter focus mode: - [1 for YES / 2 for NO] "))

if (a==1):

speak ("Entering the focus mode....")

os.startfile ("D:\\Jarvis\_Final\\FocusMode.py")

exit ()

else:

pass

elif "show my focus" in query:

from FocusGraph import focus\_graph

focus\_graph ()

elif "translate" in query:

from Translator import translategl

query = query.replace ("Jarvis"," ")

query = query.replace ("translate"," ")

translategl (query)

elif "open" in query:

query = query.replace ("open","")

query = query.replace ("Jarvis","")

pyautogui.press ("super")

pyautogui.typewrite (query)

pyautogui.sleep (2)

pyautogui.press ("enter")

elif "internet speed" in query:

wifi = speedtest.Speedtest ()

upload\_net = wifi.upload ()/1048576

download\_net = wifi.download ()/1048576

print ("Wifi Upload Speed is", upload\_net)

print ("Wifi download speed is ", download\_net)

speak (f"Wifi download speed is {download\_net}")

speak (f"Wifi Upload speed is {upload\_net}")

elif "ipl score" in query:

from plyer import notification

import requests

from bs4 import BeautifulSoup

url = "https://www.cricbuzz.com/"

page = requests.get (url)

soup = BeautifulSoup (page.text,"html.parser")

team1 = soup.find\_all (class\_ = "cb-ovr-flo cb-hmscg-tm-nm") [0].get\_text()

team2 = soup.find\_all (class\_ = "cb-ovr-flo cb-hmscg-tm-nm") [1].get\_text ()

team1\_score = soup.find\_all (class\_ = "cb-ovr-flo") [8].get\_text ()

team2\_score = soup.find\_all (class\_ = "cb-ovr-flo") [10].get\_text ()

a = print (f"{team1}: {team1\_score}")

b = print (f"{team2}: {team2\_score}")

notification.notify (

title = "IPL SCORE: - ",

message=f"{team1} :{ team1\_score}\n {team2} :{ team2\_score}",

timeout = 15)

elif "play a game" in query:

from game import game\_play

game\_play ()

elif "screenshot" in query:

import pyautogui #pip install pyautogui

im = pyautogui.screenshot ()

im.save ("ss.jpg")

elif "click my photo" in query:

pyautogui.press ("super")

pyautogui.typewrite ("camera")

pyautogui.press ("enter")

pyautogui.sleep (2)

speak ("SMILE")

pyautogui.press ("enter")

elif "hello" in query:

speak ("Hello sir, how are you?")

elif "i am fine" in query:

speak ("that's great, sir")

elif "how are you" in query:

speak ("Perfect, sir")

elif "thank you" in query:

speak ("you are welcome, sir")

elif "do you kill humans" in query:

speak (" ofcourse, sir expect good people ")

elif "i love u" in query:

speak (" love you to sir ")

elif "tired" in query:

speak ("Playing your favorite songs, sir")

a = (1, 2, 3)

b = random.choice (a)

if b==1:

webbrowser.open("https://www.youtube.com/watch?v=o8i\_iOx1F0&t=2s)

elif "pause" in query:

pyautogui.press ("k")

speak ("video paused")

elif "play" in query:

pyautogui.press ("k")

speak ("video played")

elif "mute" in query:

pyautogui.press ("m")

speak ("video muted")

elif "un mute" in query:

pyautogui.press ("m")

speak ("video un muted")

elif "volume up" in query:

from keyboard import volumeup

speak ("Turning volume up, sir")

volumeup ()

elif "volume down" in query:

from keyboard import volumedown

speak ("Turning volume down, sir")

volumedown ()

elif "open" in query:

from Dictapp import openappweb

openappweb (query)

elif "close" in query:

from Dictapp import closeappweb

closeappweb (query)

elif "goggle" in query:

from SearchNow import searchGoogle

searchGoogle (query)

elif "youtube" in query:

from SearchNow import searchYoutube

searchYoutube (query)

elif "wikipedia" in query:

from SearchNow import searchWikipedia

searchWikipedia (query)

elif "news" in query:

from NewsRead import latestnews

latestnews ()

elif "calculate" in query:

from Calculatenumbers import WolfRamAlpha

from Calculatenumbers import Calc

query = query.replace ("calculate","")

query = query.replace ("Jarvis","")

Calc (query)

elif "whatsapp" in query:

from Whatsapp import sendMessage

sendMessage ()

elif "temperature" in query:

search = "temperature in "+query

url = f"https://www.google.com/search?q= {search}"

r = requests.get (url)

data = BeautifulSoup (r.text,"html.parser")

temp = data.find ("div", class\_ = "BNeawe").text

print (f"current {search} is {temp}")

speak (f"current {search} is {temp}")

elif "weather" in query:

search = "temperature in "+query

url = f"https://www.google.com/search?q= {search}"

r = requests.get (url)

data = BeautifulSoup (r.text,"html.parser")

temp = data.find ("div", class\_ = "BNeawe").text

print (f"current {search} is {temp}")

speak (f"current {search} is {temp}")

elif "set an alarm" in query:

print ("input time example: - 10 and 10 and 10")

speak ("Set the time")

a = input ("Please tell the time: - ")

alarm (a)

speak ("Done, sir")

elif "the time" in query:

strTime = datetime.datetime.now ().strftime ("%H: %M")

speak (f"Sir, the time is {strTime}")

elif "finally sleep" in query:

speak ("Going to sleep, sir")

exit ()

elif "remember that" in query:

rememberMessage = query.replace ("remember that","")

rememberMessage = query.replace ("Jarvis","")

speak ("You told me to remember that"+rememberMessage)

remember = open ("Remember.txt","a")

remember.write (rememberMessage)

remember.close ()

elif "what do you remember" in query:

remember = open ("Remember.txt","r")

speak ("You told me that" + remember.read ())

elif "shutdown system" in query:

speak ("Are You sure you want to shutdown")

shutdown = input ("Do you wish to shut down your computer? (yes/no)")

if shutdown == "yes":

os.system ("shutdown /s /t 1")

elif shutdown == "no":

break

elif "tell me a joke" in query:

My\_joke= pyjokes.get\_joke(language="en", category="neutral")

 speak (My\_joke)

**FUTURE PERSPECTIVE**

We plan to Integrate Jarvis with mobile using react native, to provide a synchronized experience between the two connected devices. Further, in the long run, Jarvis is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general Server Administrator does. The functionality would be seamless enough to replace the Server Administrator with Jarvis.

**CONCLUSION**

Through this voice assistant, we have automated various services using a single line command. It eases most of the tasks of the user like searching the web, retrieving weather forecast details, vocabulary help and medical related queries. We aim to make this project a complete server assistant and make it smart enough to act as a replacement for a general server administration. The future plans include integrating Jarvis with mobile using React Native to provide a synchronized experience between the two connected devices. Further, in the long run, Jarvis is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general Server Administrator does. The functionality would be seamless enough to replace the Server Administrator with Jarvis.