

# Virtual Desktop AI Assistant (TONY)

## Submitted by

A.VEERENDRA NATH-19BCI0238

SYED LUQMAN-19BCE0196

BHARATCHANDRA-19BCE0852

ZEBa MARIAM-19BCE2439

## Submitted to

Dr.S.Anto Associate Professor, SCOPE

VIT , Vellore.

**School of Computer Science and Engineering**



# VIT<sup>®</sup>

## Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

**April, 2022**

### Table of content

S no	Topic	Page no
1	Abstract	3
2	Introduction	3
3	Literature Review	4-6
4	Software Requirements	6-7
5	Existing System	7-8
6	Proposed System	8-9
7	Module wise Description	9-10
8	Results and Discussion	11-19
9	Conclusion	20
10	References	20
11	Appendix	20-26

## Abstract: -

An AI personal assistant is a piece of software that understands verbal or written commands and completes task assigned by the client which makes things look very simple, Personal Assistant that understands speech as well as text input and is capable of performing tasks other than conversing. It will be like a personal assistant which will be notifying everything on our daily basis work we do.

## Introduction: -

Artificial Intelligence when used with machines, it shows us the capability of thinking like humans. In this, a computer system is designed in such a way that typically requires interaction from human. 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 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. As the voice assistant is using Artificial Intelligence hence the result that it is providing are highly accurate and efficient. The assistant can help to reduce human effort and consumes time while performing any task, they removed the concept of typing completely and behave as another individual to whom we are talking and asking to perform task. The assistant is no less than a human assistant but we can say that this is more effective and efficient to perform any task. The libraries and packages used to make this assistant focuses on the time complexities and reduces time.

The functionalities include, it can send emails, It can read PDF, It can send text on WhatsApp, It can open command prompt, your favorite IDE, notepad etc., It can play music, It can do Wikipedia searches for you, It can open websites like Google, YouTube, etc., in a web browser, It can give weather forecast, It can give desktop reminders of your choice. It can have some basic conversation. Tools and technologies used are PyCharm IDE for making this project, and I created all py files in PyCharm. Along with this I used following modules and libraries in my project. pyttsx3, Speech Recognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, PyQt etc. I have created a live GUI for interacting with the TONY as it gives a design and interesting look while having the conversation.

## Literature Review: -

### Research paper 1

**Title:** - An Intelligent Personal Assistant for Task and Time Management

**Author's:** - Karen Myers, Pauline Berry, Jim Blythe, Ken Conley, Melinda Gervasio, Deborah McGuinness, David Morley, Avi Pfeffer, Martha Pollack , and Milind Tambe. **Summary:** -

In this paper they have described an intelligent personal assistant that has been developed to aid a busy knowledge worker in managing time commitments and performing tasks. The design of the system was motivated by the complementary objectives of (1) relieving the user of routine tasks, thus allowing her to focus on tasks that critically require human problem-solving skills, and (2) intervening in situations where cognitive overload leads to oversights or mistakes by the user. The system draws on a diverse set of AI technologies that are linked within a Belief-Desire-Intention (BDI) agent system. Although the system provides a number of automated functions, the overall framework is highly user centric in its support for human needs, responsiveness to human inputs, and adaptivity to user working style and preferences.

### Research paper 2

**Title:** - Experience with a Learning Personal Assistant

**Author's:** - Tom Mitchell, Rich Caruana, Dayne Freitag, John McDermott , David Zebrowski. **Summary:** -

The main goal of this paper is to build a Personal software assistant that help users with tasks like finding information, scheduling calendars, or managing work-flow will require significant customization to each individual user. For example, an assistant that helps schedule a particular user's calendar will have to know that user's scheduling preferences. This paper explores the potential of machine learning methods to automatically create and maintain such customized knowledge for personal software assistants. We describe the design of one particular learning assistant: a calendar manager, called CAP (Calendar Apprentice), that learns user scheduling preferences from experience. Results are summarized from approximately five user-years of experience, during which CAP has learned an evolving set of several thousand rules that characterize the scheduling preferences of its users. Based on this experience, we suggest that machine learning methods may play an important role in future personal software assistants.

### Research paper 3

**Title:** - The dark sides of AI personal assistant: effects of service failure on user continuance intention.

**Author's:** - Yi Sun, Shihui Li ,Lingling Yu

**Summary:** -

With the popularization of smart devices and the rapid development of smart voice technology, AI personal assistants (AIPAs) have penetrated deeply into users' lives. Compared with previous years, the accuracy, semantic understanding ability, and wake-up ability of AIPAs have been improved, but the lack of service maturity and the insufficient degree of scene integration have brought users a poor human–computer interaction experience. However, studies have scarcely uncovered the underlying mechanism through which those dark sides of AIPAs exert impacts on users' continuance intention. From the perspective of technostress, the current study proposes a theoretical model for consumers to cope with service failure pressure sources. This article collected 413 questionnaires and conducted an empirical analysis. Results show that negative technical characteristics will affect consumers' psychological responses and ultimately affect consumers' technical exhaustion, satisfaction, and two kinds of continuance intentions (general and partial continuance intentions) through cognitive load. Findings open up new avenues for research by exploring the mechanism of how the service failures of these AIPAs affect consumers' continuance intention through the perspective of technostress.

#### Research paper 4

**Title:** - Enhancing problem-solving skills with smart personal assistant technology

**Author's:** - Rainer Winkler, Matthias Sollner, Jan Marco Leimeister.

**Summary:** -

In study of this paper answered two research questions. First, we investigated whether SPA technology could increase students' problem- solving skills. Within two experiments in two different school types (high school and vocational school), we were able to show that the use of SPAs over a period of five weeks has a positive effect on skill development, more precisely the development of problem- solving skills. Second, we examined how the use of SPA technology changes students' learning processes. Students showed more interactive learning behaviour and used the SPA in different ways compared to traditional learning aids. Moreover, students appreciated that they received individualized support from the SPA. The study provides empirical evidence for the usefulness of SPA technology and offers fresh insights into how this technology can change students learning processes. The findings of this study contribute to computer tutoring and technology-mediated learning research

## Research paper 5

**Title:** - Intelligent Virtual Assistant knows Your Life

**Author's:** -Hyunji Chung and Sangjin Lee

**Summary:** -

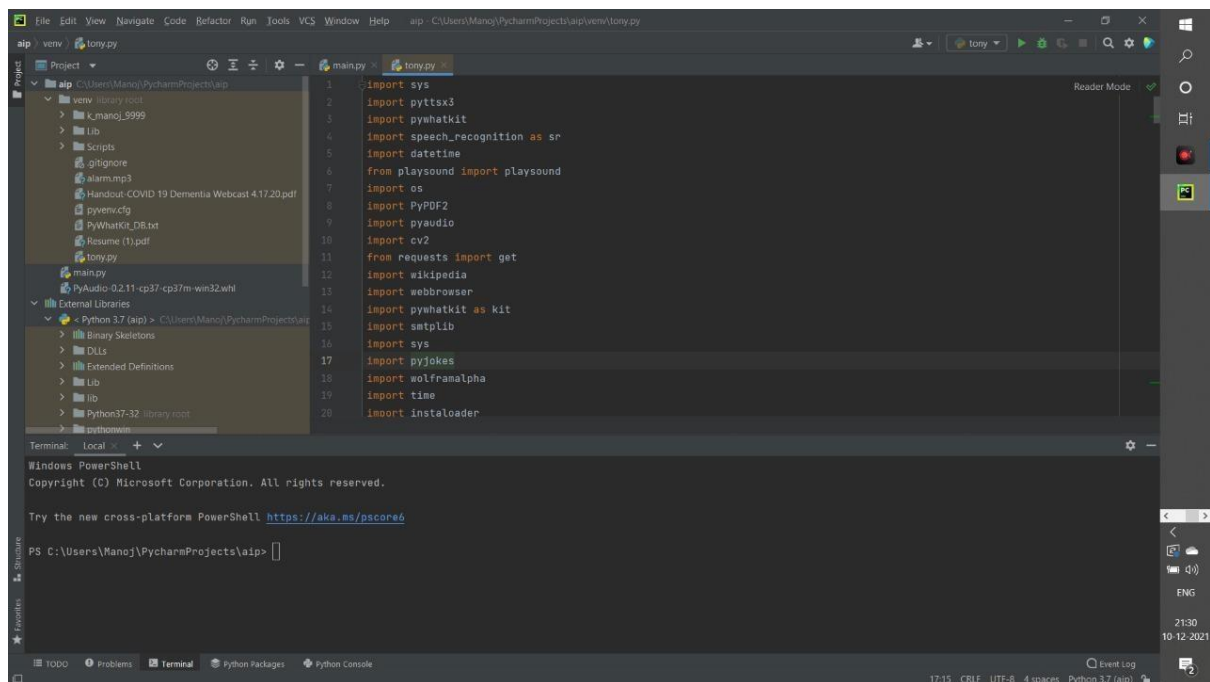
In recent days, cloud-based IoT devices are evolving rapidly and spreading widely in our lives. Many people are becoming accustomed to interacting with various IoT consumer products, such as intelligent virtual assistants. In these circumstances, lots of data are being produced in real time in response to user behaviours. Interestingly, a large number of behavioural traces that include user's voice activity history with detailed descriptions can be stored in the remote servers. Until now, there has been little research reported on analysis of intelligent virtual assistant related data collected from cloud servers. In this paper, we showed and categorized types of IVA-related data that can be collected from a popular IVA, Amazon Alexa. We then analysed an experimental dataset from Amazon Alexa, and characterized several properties of a user's lifestyle and life patterns. Our results showed that it is possible to uncover new insights on personal information such as IVA usage patterns, user's interests and sleeping/wake-up patterns. The results presented in this work provide important implications for security and privacy threats to IVA vendors and users as well.

**Software Requirements:** -

The IDE used in this project is PyCharm. All the python files were created in PyCharm and all the necessary packages were easily installable in this IDE. For this project following modules and libraries were used i.e. pyttsx3, Speech Recognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, PyQt etc. I have created a live GUI for interacting with the TONY as it gives a design and interesting look while having the conversation.

### ○ PYCHARM

It is an IDE i.e. Integrated Development Environment which has many features like it supports scientific tools (like matplotlib, NumPy, SciPy etc) web frameworks (example Django, web2py and Flask) refactoring in Python, integrated python debugger, code completion, code and project navigation etc. It also provides Data Science when used with Anaconda.



## ○ PYTHON LIBRARIES

In TONY following python libraries were used: pyttsx3: It

- is a python library which converts text to speech.
- Speech Recognition: It is a python module which converts speech to text.
- pywhatkit: It is python library to send WhatsApp message at a particular time with some additional features.
- Datetime: This library provides us the actual date and time.
- Wikipedia: It is a python module for searching anything on Wikipedia.
- Smtplib: Simple mail transfer protocol that allows us to send mails and to route mails between mail servers.
- pyPDF2: It is a python module which can read, split, merge any PDF.
- Pyjokes: It is a python libraries which contains lots of interesting jokes in it.
- Web browser: It provides interface for displaying web-based documents to users.
- os: It represents Operating System related functionality.
- sys: It allows operating on the interpreter as it provides access to the variables and functions that usually interact strongly with the interpreter.

## Existing system/Drawbacks: -

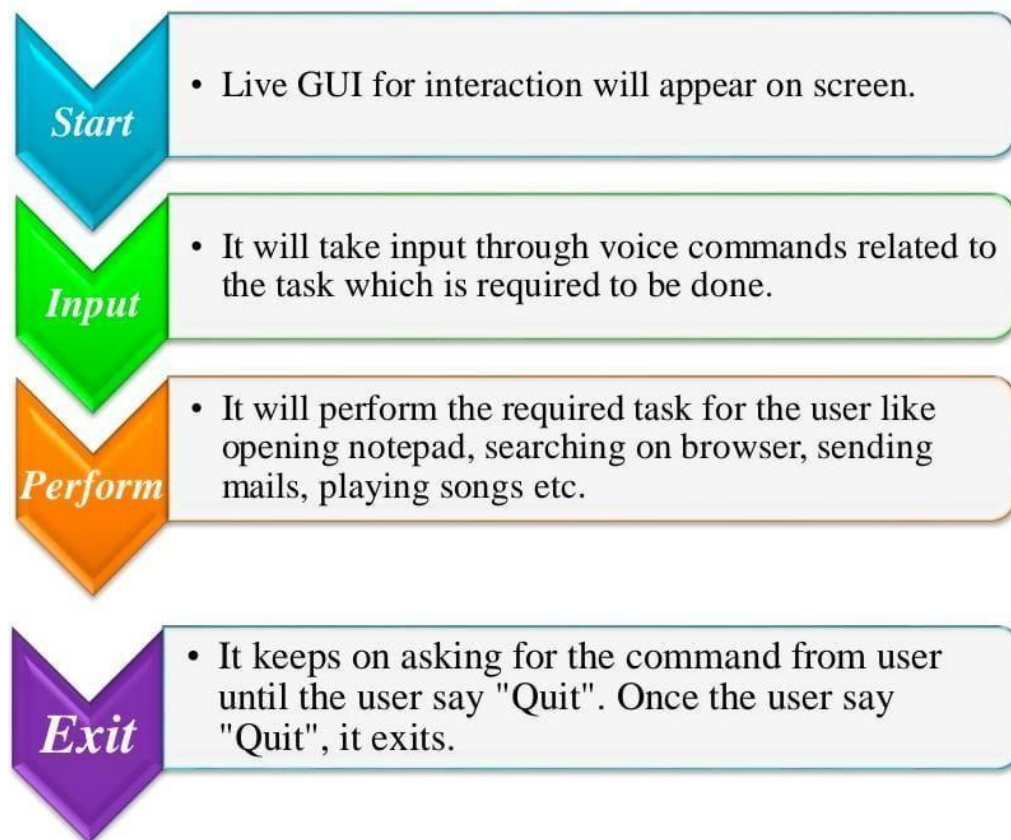
We are familiar with many existing voice assistants like Alexa, Siri, Google Assistant, Cortana which uses concept of language processing, and voice recognition. They listen the command given by the user as per their requirements and performs that specific function in a very efficient and effective manner. As these voice assistants are using Artificial Intelligence hence the result that they are providing are highly accurate and efficient. These assistants can help to reduce human effort and consumes time while performing any task, they removed the concept of typing completely and behave as another individual to whom we are talking and asking to

perform task. These assistants are no less than a human assistant but we can say that they are more effective and efficient to perform any task. The algorithm used to make these assistant focuses on the time complexities and reduces time. But for using these assistants one should have an account (like Google account for Google assistant, Microsoft account for Cortana) and can use it with internet connection only because these assistants are going to work with internet connectivity. They are integrated with many devices like, phones, laptops, and speakers etc.

### Proposed system: -

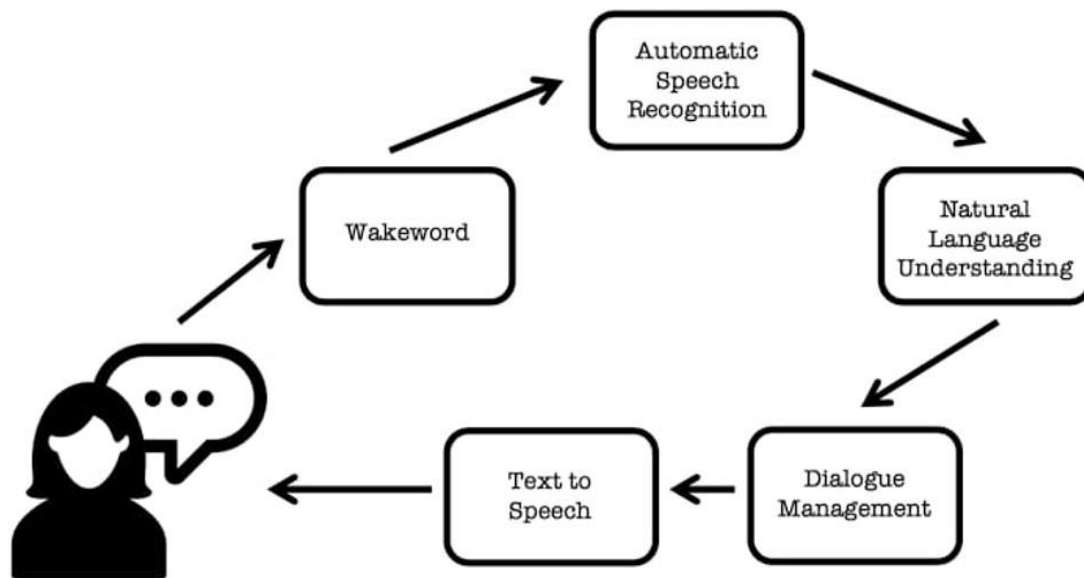
It was an interesting task to make my own assistant. It became easier to send emails without typing any word, searching on Google without opening the browser, and performing many other daily tasks like playing music, opening your favorite IDE with the help of a single voice command. Jarvis is different from other traditional voice assistants in terms that it is specific to desktop and user does not need to make account to use this, it does not require any internet connection while getting the instructions to perform any specific task. The IDE used in this project is PyCharm. All the python files were created in PyCharm and all the necessary packages were easily installable in this IDE. For this project following modules and libraries were used i.e., pyttsx3, Speech Recognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, PyQt.

- Design: -





- Methodology: -



### Module wise description: -

### Implementation: -

TONY, a desktop assistant is a voice assistant that can perform many daily tasks of desktop like playing music, opening your favorite IDE with the help of a single voice command. Jarvis is different from other traditional voice assistants in terms that it is specific to desktop and user does not need to make account to use this, it does not require any internet connection while getting the instructions to perform any specific task.

### REAL LIFE APPLICATION

- Saves time: TONY is a desktop voice assistant which works on the voice command offered to it, it can do voice searching, voice-activated device control and can let us complete a set of tasks.
- Conversational interaction 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.
- Reactive nature: 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.

- Multitasking: The main application of it can be its multitasking ability. It can ask for continuous instruction one after other until the user “QUIT” it.
- No Trigger phase: 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.

## DATA IMPLEMENTATION AND PROGRAM EXECUTION

As the first step, install all the necessary packages and libraries. The command used to install the libraries is “pip install” and then import it. The necessary packages included are as follows:

## LIBRARIES AND PACKAGES

- ✦ pyttsx3: It is a python library which converts text to speech.
- ✦ Speech Recognition: It is a python module which converts speech to
- ✦ pywhatkit: It is python library to send WhatsApp message at a particular time with some additional features.
- ✦ Datetime: This library provides us the actual date and time.
- ✦ Wikipedia: It is a python module for searching anything on Wikipedia.
- ✦ Smtplib: Simple mail transfer protocol that allows us to send mails and to route mails between mail servers.
- ✦ pyPDF2: It is a python module which can read, split, merge any PDF.
- ✦ Pyjokes: It is a python library which contains lots of interesting jokes in it.
- ✦ Web browser: It provides interface for displaying web-based documents to users.
- ✦ Pyautogui: It is a python librariy for graphical user interface.
- ✦ os: It represents Operating System related functionality.
- ✦ sys: It allows operating on the interpreter as it provides access to the variables and functions that usually interact strongly with the interpreter.

## FUNCTIONS

take Command (): The function is used to take the command as input through microphone of user and returns the output as string.

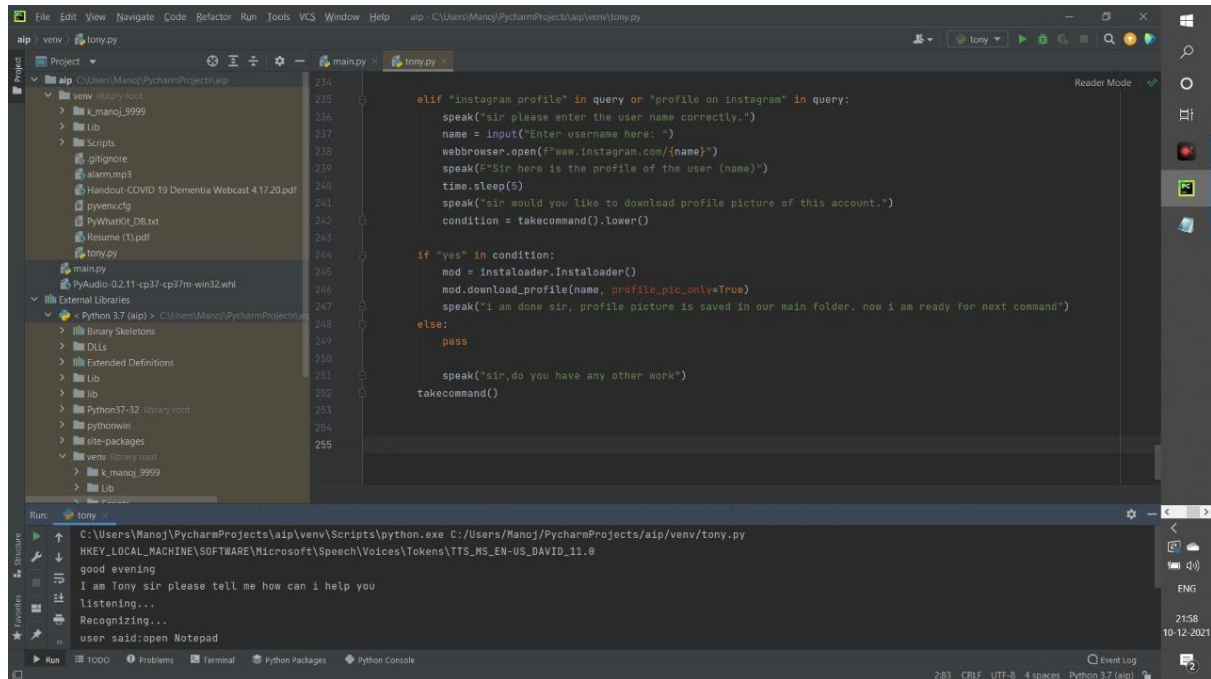
Wish Me (): This function greets the user according to the time like Good Morning, Good Afternoon and Good Evening.

Task Execution (): This is the function which contains all the necessary task execution definition like send Email(), pdf reader(), news() and many conditions in if condition like “open google”, “open notepad”, “search on Wikipedia” ,”play music” and “open command prompt” etc.

## Results and Discussion: -

### Outputs: -

#### Input for Notepad:



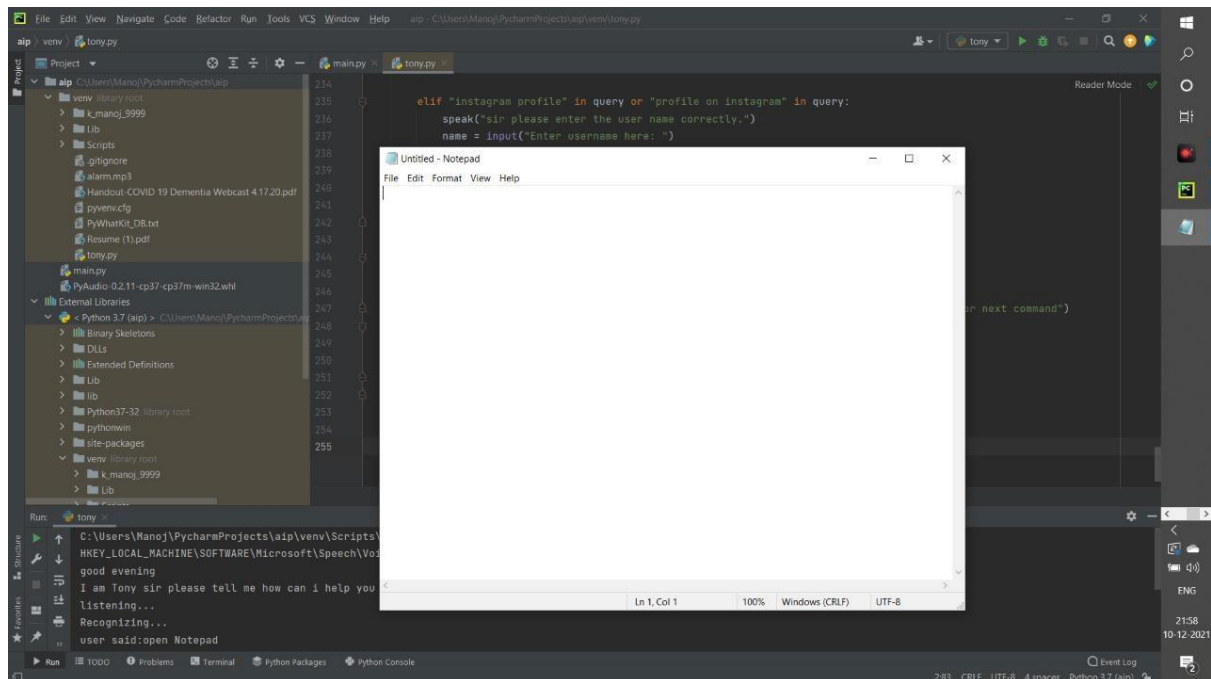
The screenshot shows the PyCharm IDE with the 'tony.py' file open. The code in the file is as follows:

```
234 elif "instagram profile" in query or "profile on instagram" in query:
235     speak("sir please enter the user name correctly.")
236     name = input("Enter username here: ")
237     webbrowser.open(f"www.instagram.com/{name}")
238     speak(f"sir here is the profile of the user {name}")
239     time.sleep(5)
240     speak("sir would you like to download profile picture of this account.")
241     condition = takecommand().lower()
242
243 if "yes" in condition:
244     mod = instaloader.Instaloader()
245     mod.download_profile(name, profile_pic_only=True)
246     speak("i am done sir, profile picture is saved in our main folder. now i am ready for next command")
247 else:
248     pass
249
250 speak("sir, do you have any other work")
251 takecommand()
```

The Run console at the bottom shows the following output:

```
C:\Users\Manoj\PycharmProjects\aip\venv\Scripts\python.exe C:\Users\Manoj\PycharmProjects\aip\venv\tony.py
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_EN-US_DAVID_11.0
good evening
I am Tony sir please tell me how can i help you
listening...
Recognizing...
user said:open Notepad
```

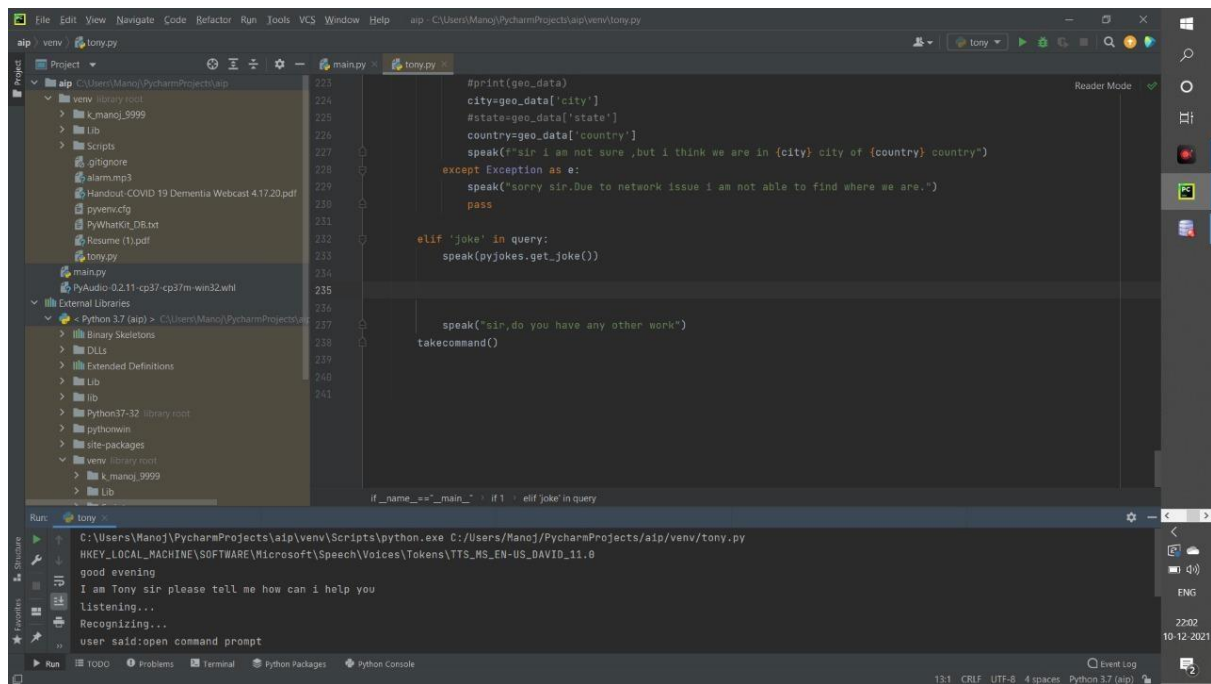
#### Output of the Notepad:



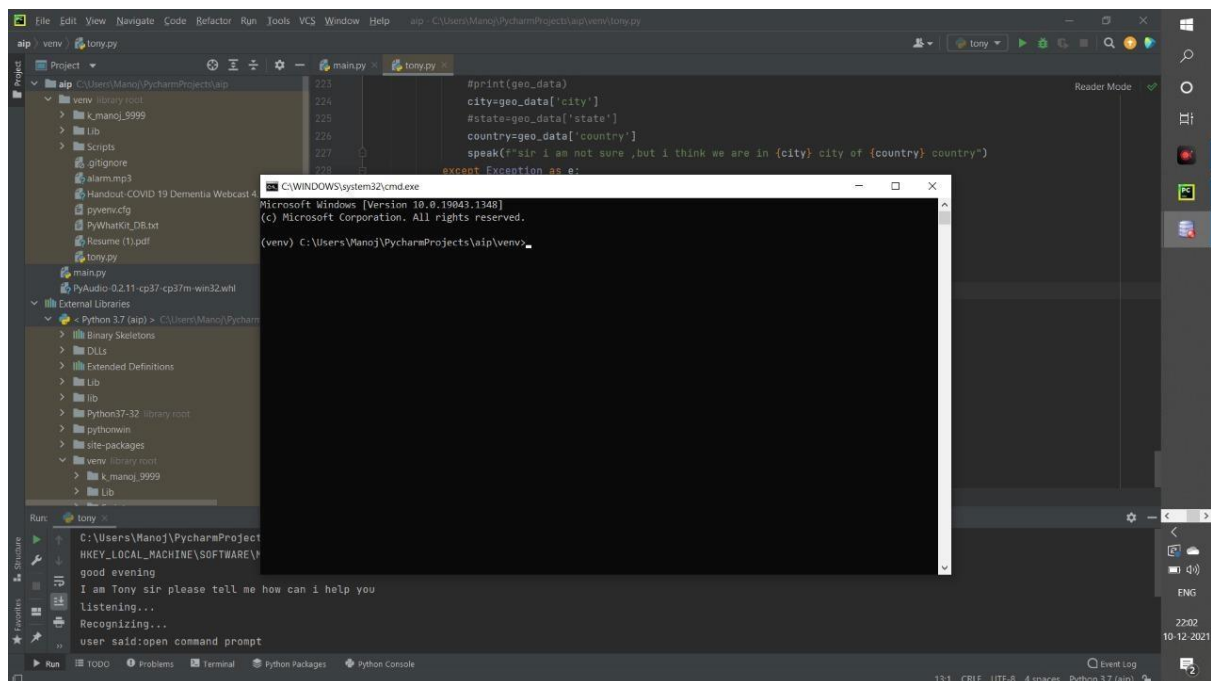
The screenshot shows the PyCharm IDE with the 'tony.py' file open. A Notepad window is open in the foreground, displaying the output of the Notepad application. The output is as follows:

```
File Edit Format View Help
|
Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

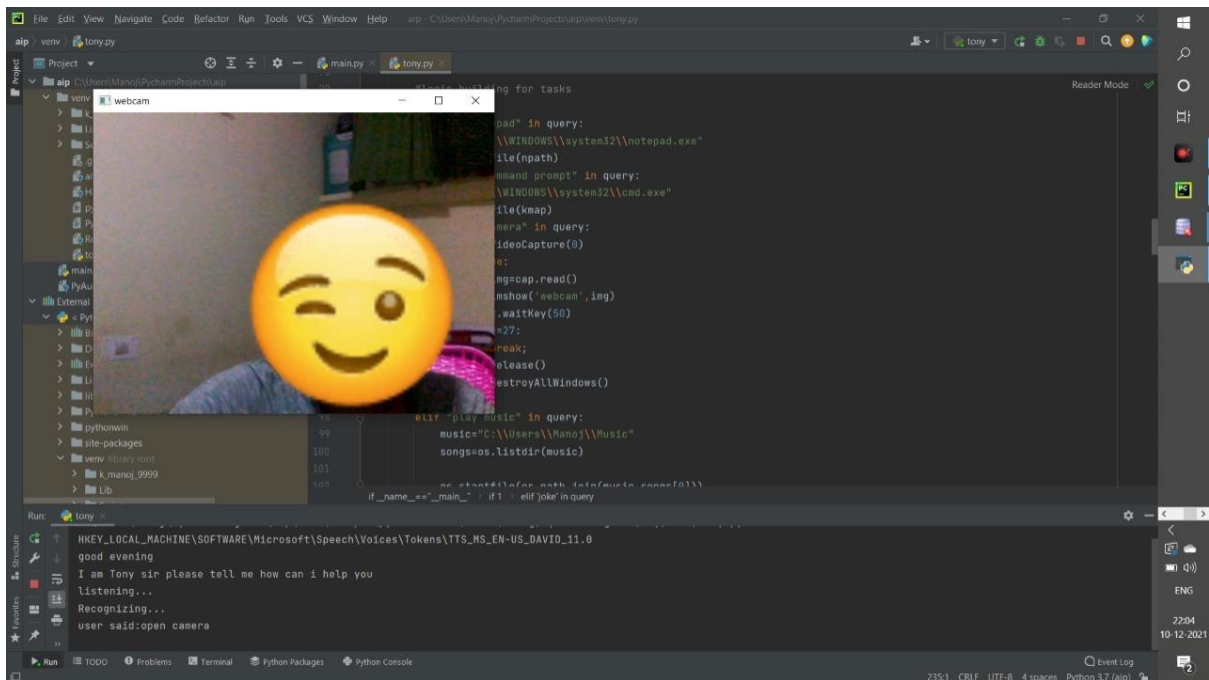
## Input for Command Prompt:



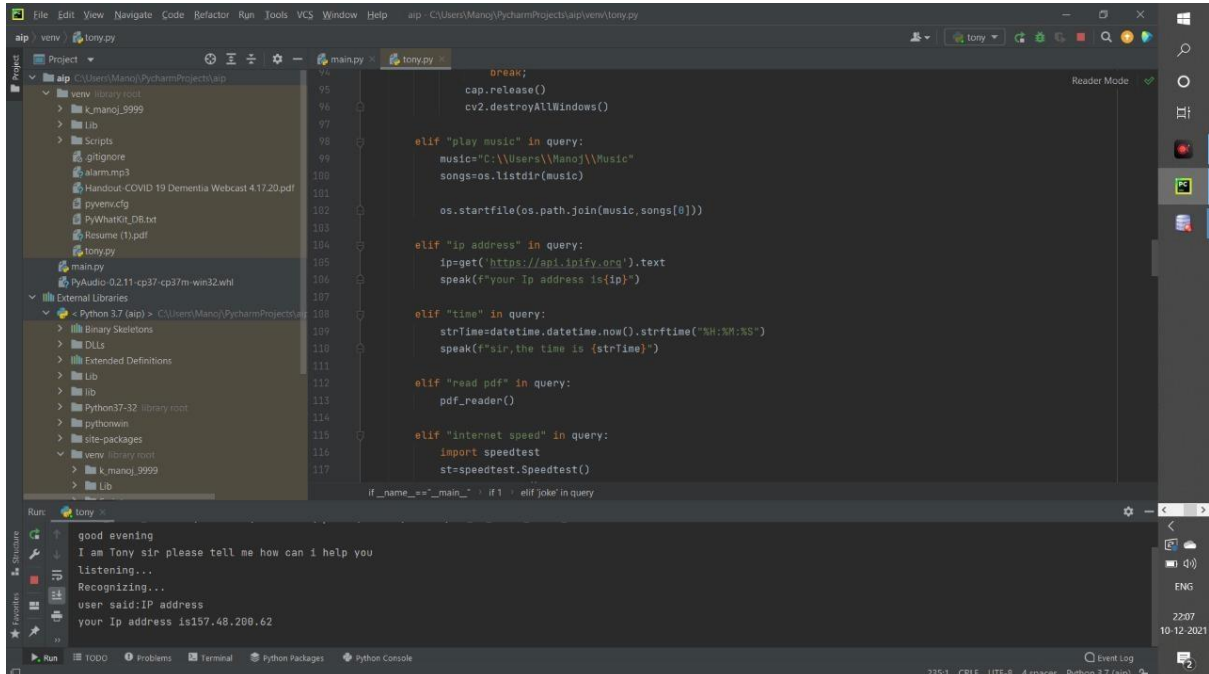
## Output of the Command Prompt:



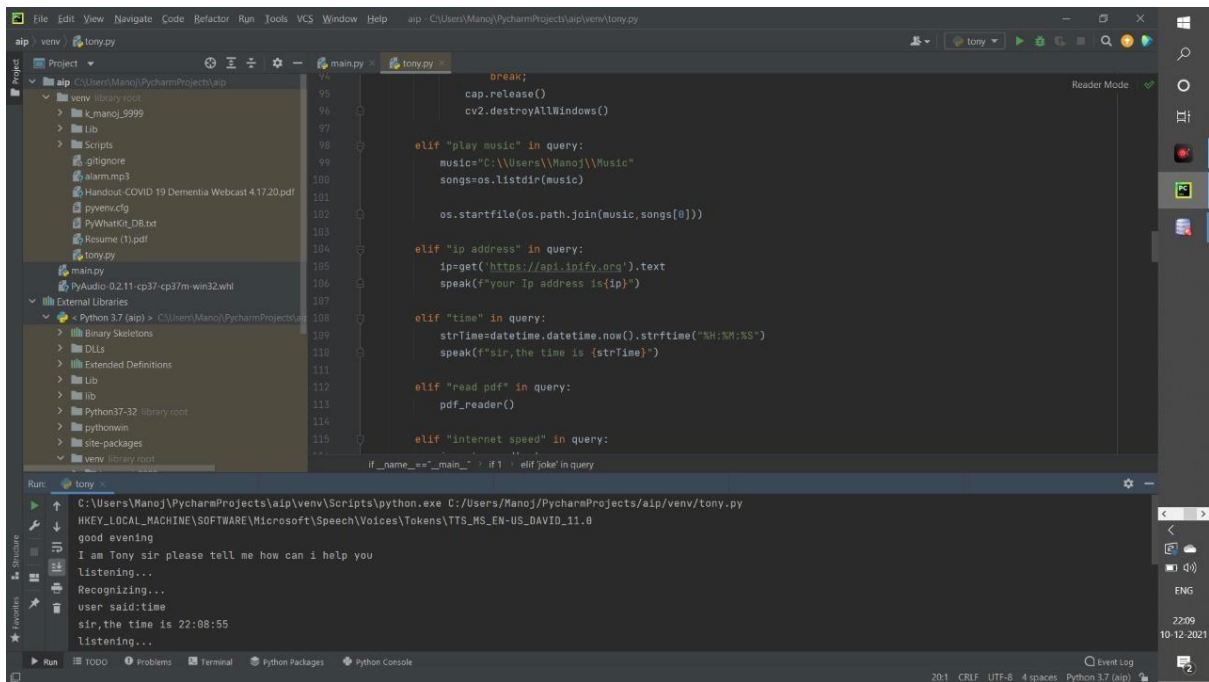
## Output of the web-cam:



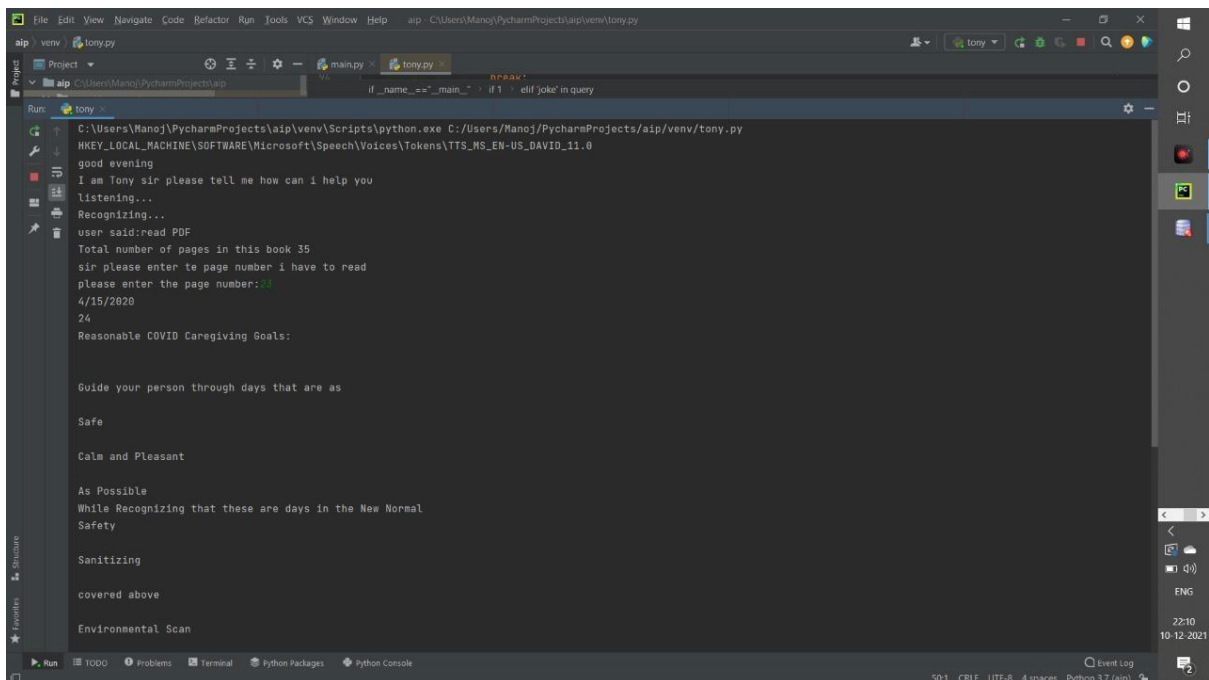
output for IP address:



To check time:

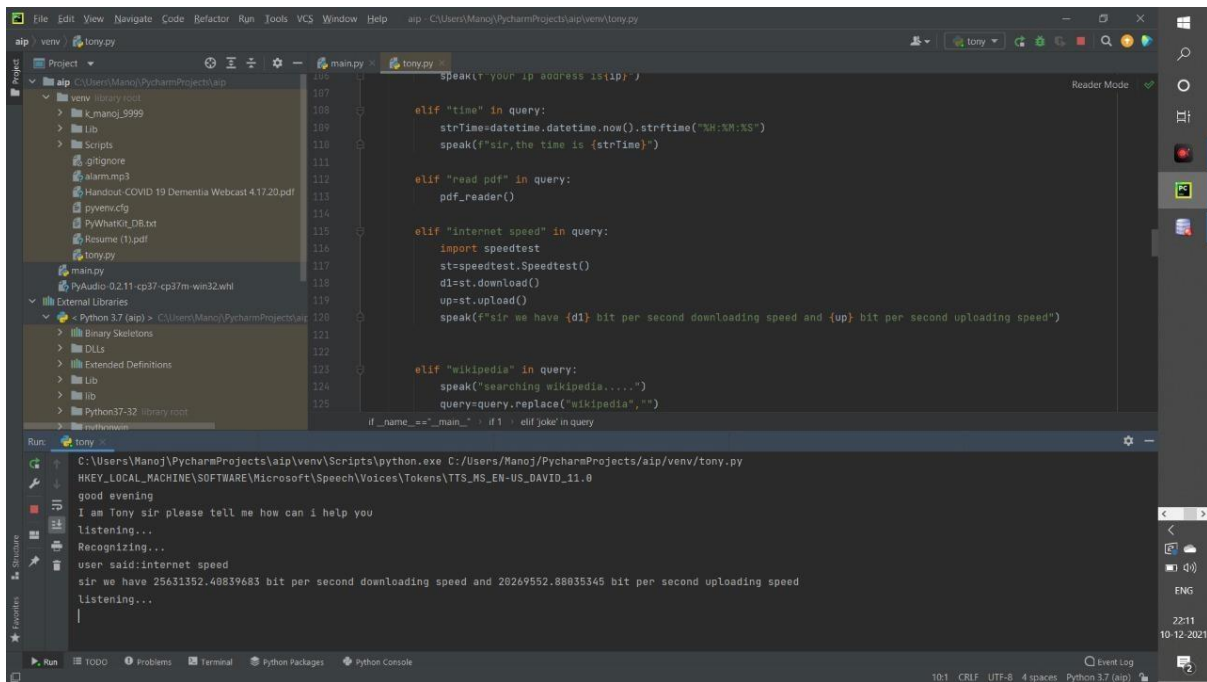


To read PDF:

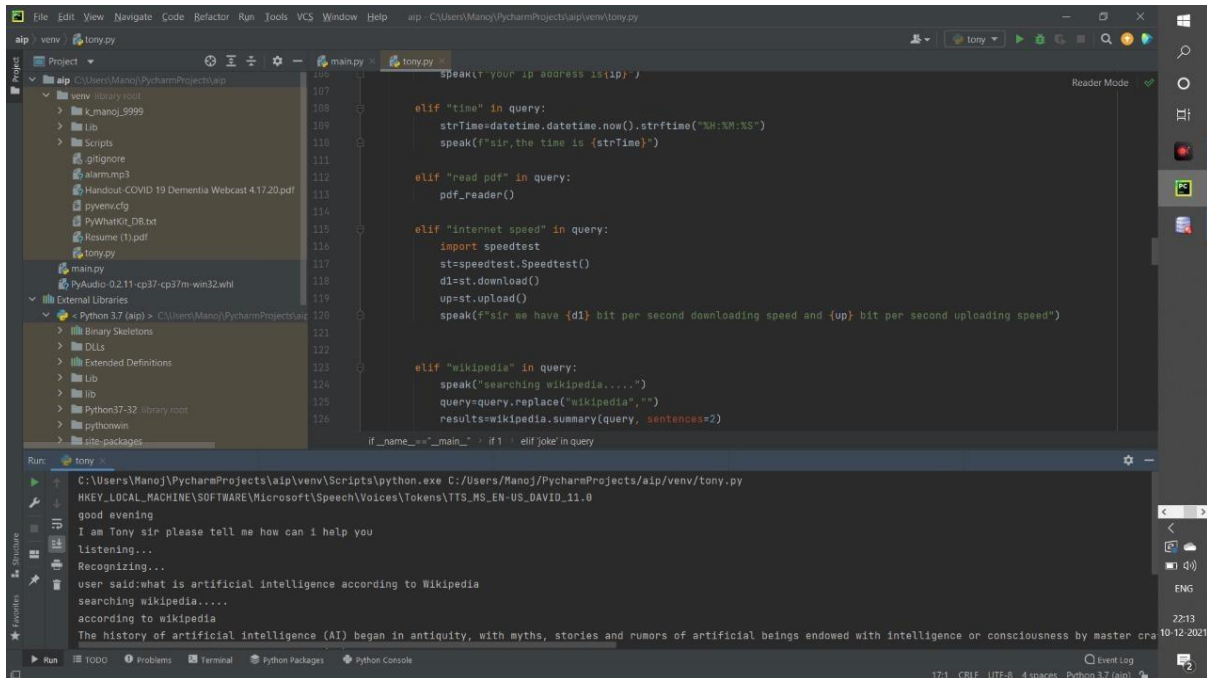


To check our internet speed:

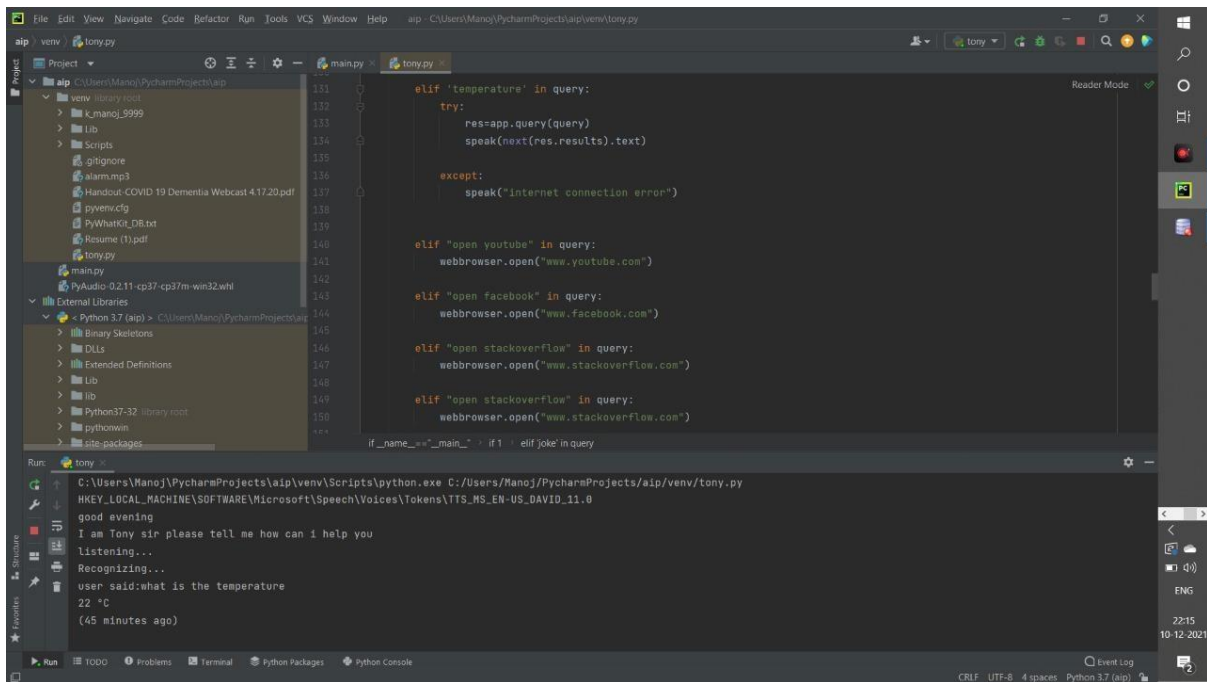




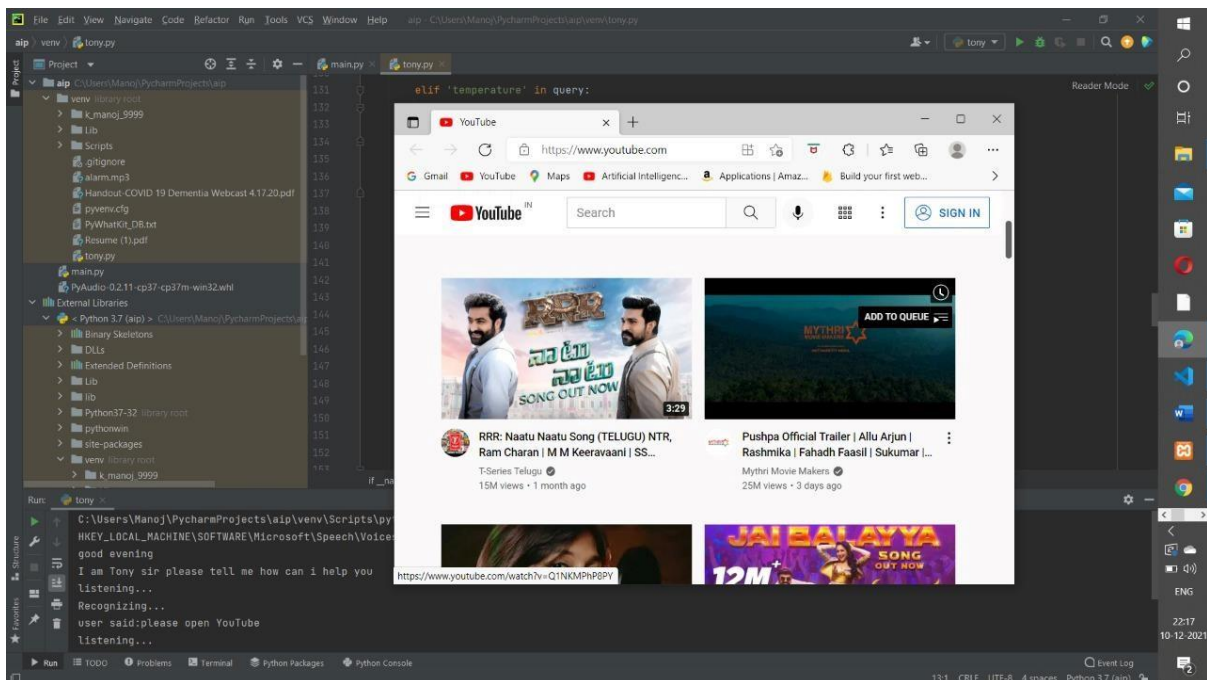
To search on Wikipedia:



To check the Temperature:

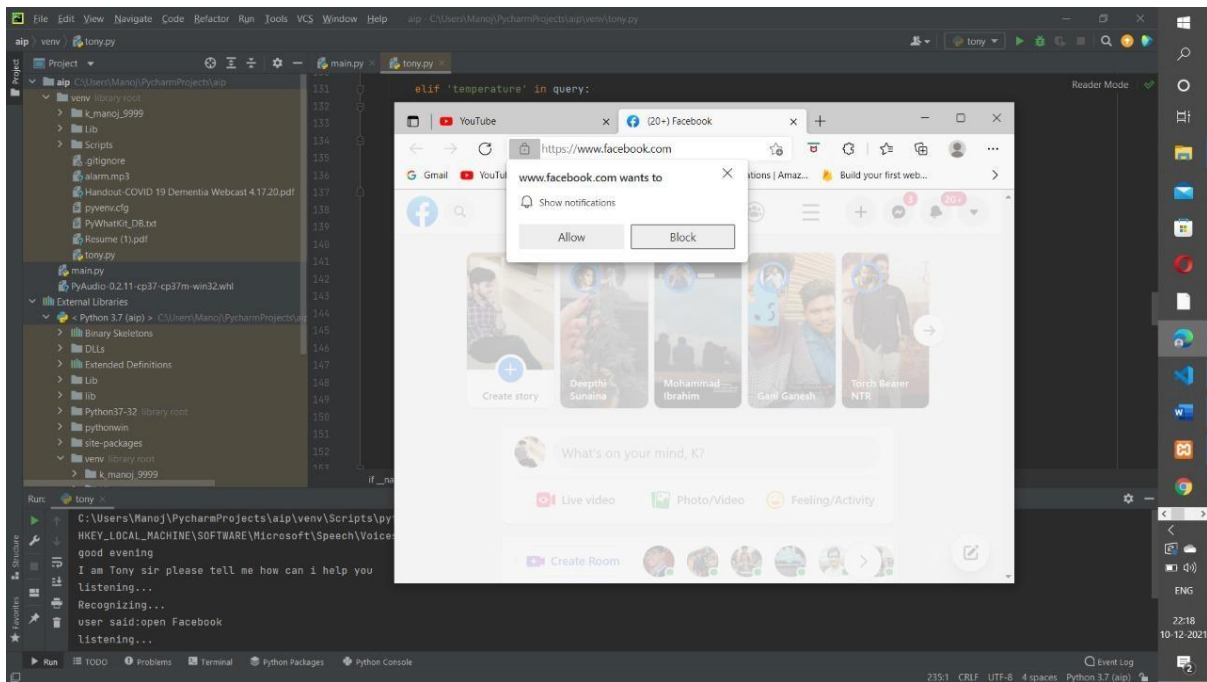


To open Entertainment website (You Tube):

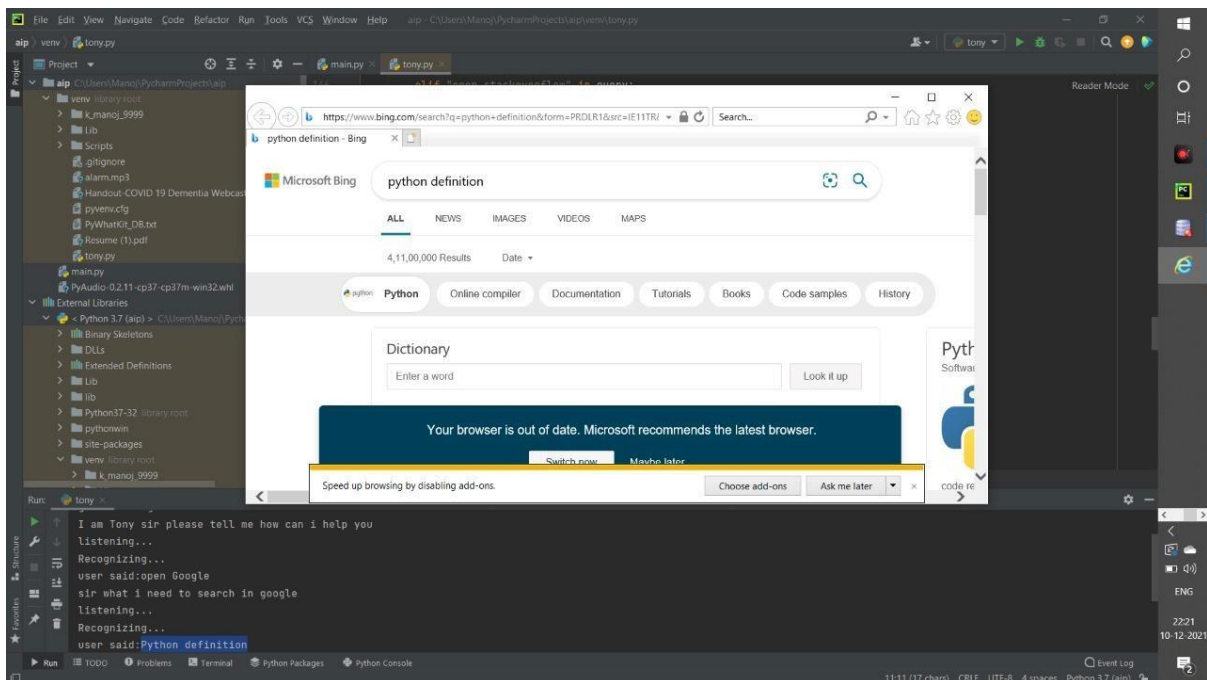


To Open Facebook:

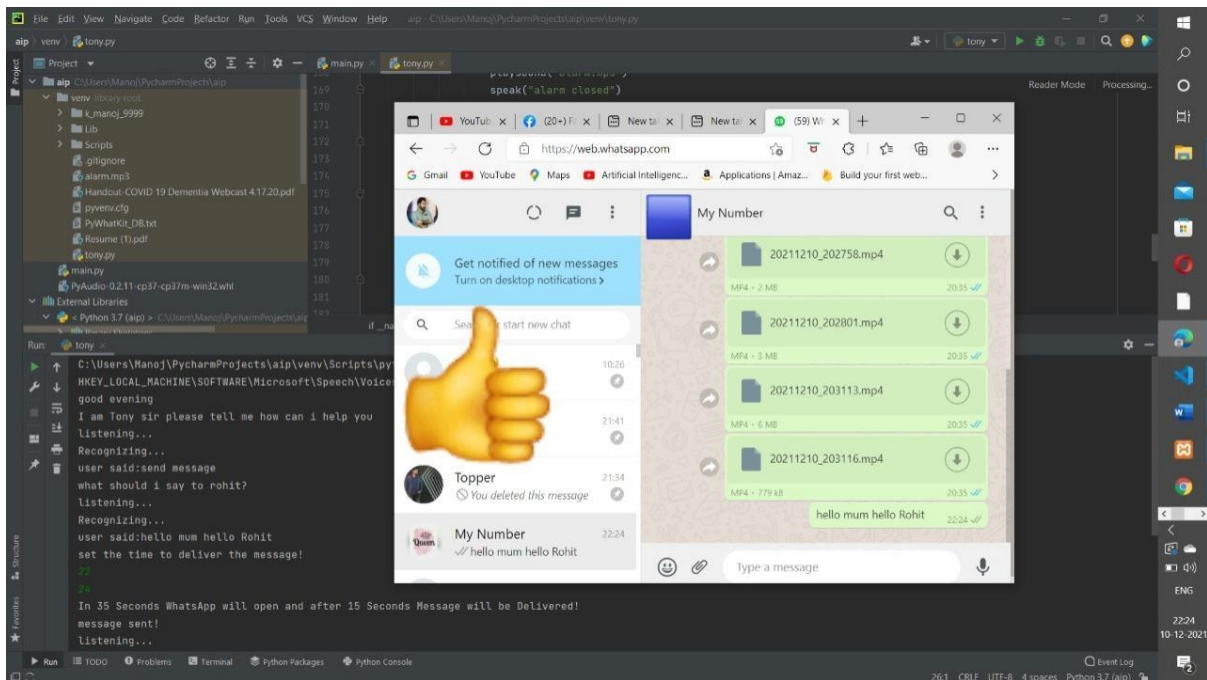




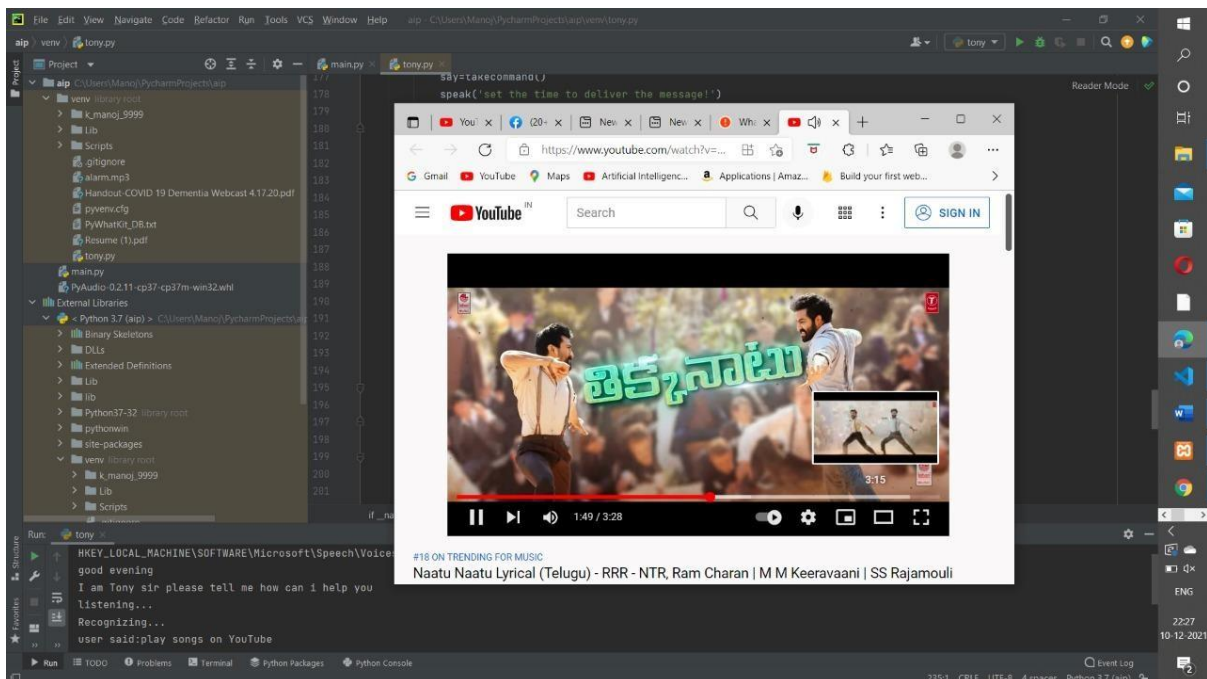
To open google and search any command:



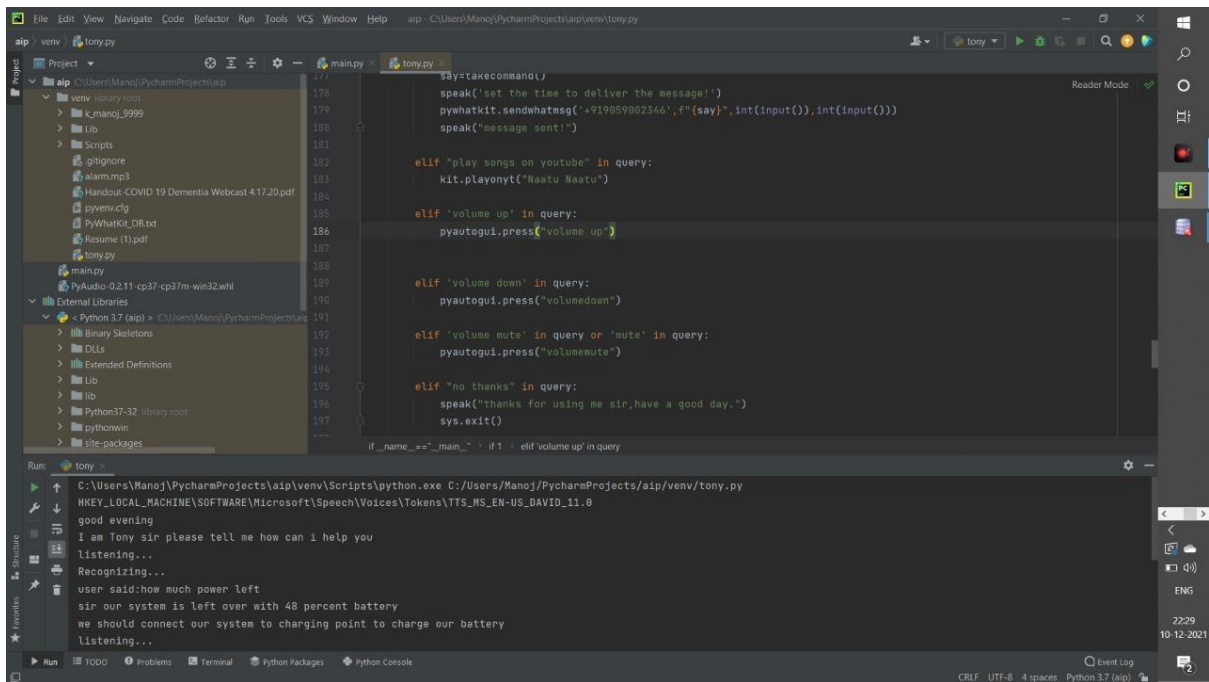
Dynamically taking the input and we can schedule the time for sending the message in WhatsApp to our beloved person:



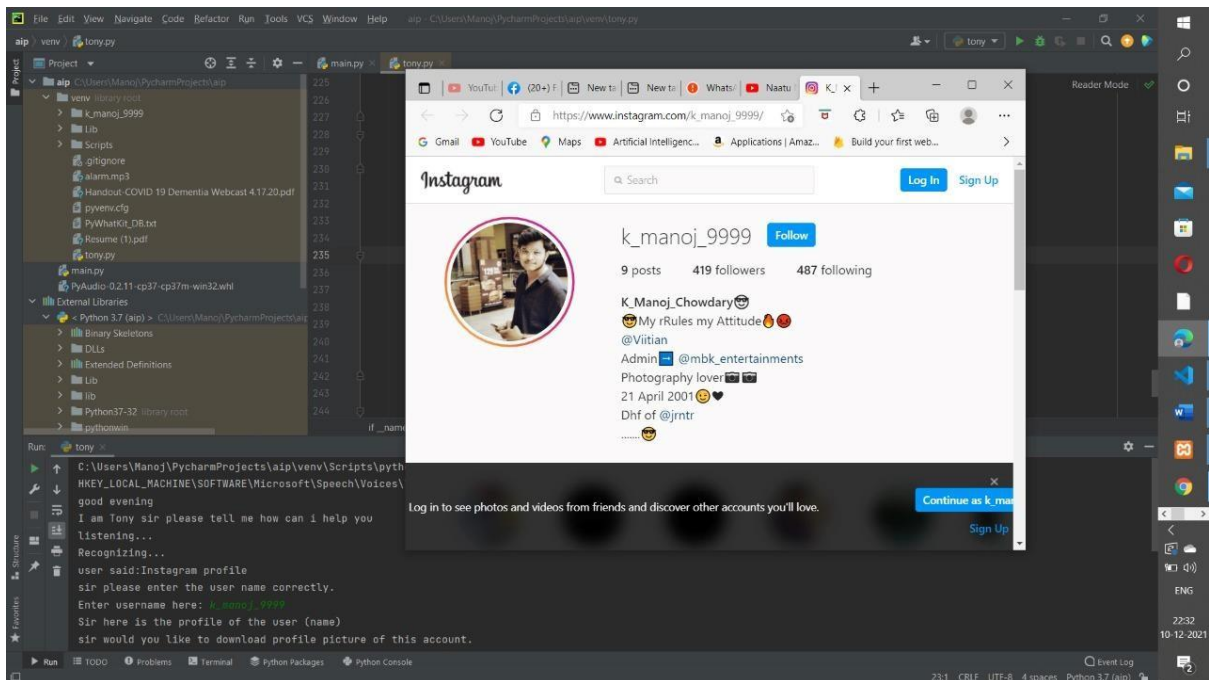
Playing the selected song on You Tube:



To check our System Battery:



Output for opening Instagram and to download the profile picture into a folder if needed:



## Conclusion: -

TONY is a very helpful voice assistant without any doubt as it saves time of the user by conversational interactions, its effectiveness and efficiency. But while working on this project, there were some limitations encountered and also realized some scope of enhancement in the future which are mentioned below:

### LIMITATIONS

- Security is somewhere an issue, there is no voice command encryption in this project.
- Background voice can interfere
- Misinterpretation because of accents and may cause inaccurate results.
- TONY cannot be called externally anytime like other traditional assistants like Google Assistant can be called just by saying, “Ok Google!”

### SCOPE FOR FUTURE WORK

- Make JARVIS to learn more on its own and develop a new skill in it.
- TONY android app can also be developed.
- Make more Jarvis voice terminals.
- Voice commands can be encrypted to maintain security.

## References: -

1. <https://arxiv.org/abs/1803.00466>
2. <https://www.sciencedirect.com/science/article/pii/S0360131521000257>
3. <https://link.springer.com/article/10.1007/s12525-021-00483-2>
4. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.706&rep=rep1&type=pdf>
5. <https://ojs.aaai.org/index.php/aimagazine/article/view/2039>

## Appendix: -

```
import sys
import pyttsx3
import pywhatkit
import speech_recognition as sr
import datetime
from playsound import playsound
import os
import PyPDF2
import pyaudio
import cv2
import requests
import get
import wikipedia
```

```

import webbrowser
import pywhatkit as
kit import smtplib
import sys import
pyjokes import
wolframalpha import
time import
instaloader
import pyautogui
app=wolframalpha.Client("EUU4UU-48HQQJHRTWY")
engine = pyttsx3.init('sapi5') voices
= engine.getProperty('voices')
print(voices[0].id)
engine.setProperty('voice', voices[0].id)
#text to speech def speak(audio):
print(audio) engine.say(audio)
engine.runAndWait() #to convert
voice into text def
takecommand():
r=sr.Recognizer() with
sr.Microphone() as source:
    print("listening...")
    r.pause_threshold=5
audio=r.listen(source,timeout=1,phrase_time_limit=5)

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

    except Exception as e: speak("say
that again please manoj...")
        return "none"
    return query

#to wish def wish():
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")
    speak("I am Tony sir please tell me how can i help you")

```

```

def pdf_reader():  book=open('Handout-COVID 19 Dementia
Webcast 4.17.20.pdf','rb')  pdfReader=PyPDF2.PdfFileReader(book)
pages=pdfReader.numPages  speak(f"Total number of pages in this
book {pages}")  speak("sir please enter te page number i have to
read")  pg=int(input("please enter the page number:"))
page=pdfReader.getPage(pg)  text=page.extractText()
    speak(text)

if __name__=="main_":
    wish()
    if
1:

        query=takecommand().lower()

        #logic building for tasks

        if "open notepad" in query:
            npath="C:\\WINDOWS\\system32\\notepad.exe"
            os.startfile(npath)      elif "open
command prompt" in query:
            kmap="C:\\WINDOWS\\system32\\cmd.exe"
            os.startfile(kmap)
        elif "open camera" in query:
            cap=cv2.VideoCapture(0)
            while True:
                ret,img=cap.read()
            cv2.imshow('webcam',img)
            k=cv2.waitKey(50)      if
            k==27:      break;
            cap.release()
            cv2.destroyAllWindows()

            elif "play music" in query:
            music="C:\\Users\\Manoj\\Music"
            songs=os.listdir(music)

                os.startfile(os.path.join(music,songs[0])) elif

                "ip address" in query:

                    ip=get('https://api.ipify.org').text

                    speak(f"your Ip address is {ip}")

```

```

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

        elif "read pdf" in query:
            pdf_reader()

        elif "internet speed" in query:
            import speedtest
            st=speedtest.Speedtest()
            d1=st.download()
            up=st.upload()
            speak(f"sir we have {d1} bit per second downloading speed and {up} bit per
            second uploading speed")

        elif "wikipedia" in query:
            speak("searching
            wikipedia.....")
            query=query.replace("wikipedia","")
            results=wikipedia.summary(query, sentences=2)
            speak("according to wikipedia")
            speak(results)
            print(results)

        elif 'temperature' in query:
            try:
                res=app.query(query)
                speak(next(res.results).text)

            except:
                speak("internet connection error")

        elif "open youtube" in query:
            webbrowser.open("www.youtube.com")

        elif "open facebook" in query:
            webbrowser.open("www.facebook.com")

        elif "open stackoverflow" in query:
            webbrowser.open("www.stackoverflow.com") elif
            "open stackoverflow" in query:
            webbrowser.open("www.stackoverflow.com")

```

```

elif "open google" in query:          speak("sir
what i need to search in google")
    cm=takecommand().lower()
    webbrowser.open(f"{cm}")

elif 'alarm' in query:
speak("Enter the Time!")
    time=input(":Enter the Time:")

while True:
    Time_Ac=datetime.datetime.now()
    now=Time_Ac.strftime("%H:%M:%s")

    if now==time:
        speak("Time to wakeup sir!")
        playsound('alarm.mp3')
        speak("alarm closed")

    elif now>time:
        break

elif "send message" in query:
speak('what should i say to rohit?')
    say=takecommand()
    speak('set the time to deliver the message!')
    pywhatkit.sendwhatmsg('+919059002346',f"{say}",int(input()),int(input()))
speak("message sent!")

elif "play songs on youtube" in query:
    kit.playonyt("Naatu Naatu")

elif 'volume up' in query:
    pyautogui.press("volume up")

elif 'volume down' in query:
    pyautogui.press("volumedown")

elif 'volume mute' in query or 'mute' in query:
    pyautogui.press("volumemute") elif

"no thanks" in query: speak("thanks

```



```

for using me sir,have a good day.")

sys.exit()

elif "how much power left" in query or "how much power we have" in query
or "battery" in query:      import psutil      battery=psutil.sensors_battery()
percentage=battery.percent      speak(f"sir our system is left over with
{percentage} percent battery")      if percentage>=75:
    speak("we have enough power to continue our work")      elif
percentage>=40 and percentage<=75:      speak("we should connect our
system to charging point to charge our battery")      elif percentage<=15
and percentage<=30:      speak("we don't have enough power to
work,please keep charging")      elif percentage<=15:
    speak("we have very less power ,please connect to charging the system will
shutdown very soon")

elif "where i am" in query or "where we are" in
query:      speak("wait sir,let me check")      try:
    ipAdd=requests.get("https://api.ipfy.org").text
print(ipAdd)
    url='https://get.geojs.io/v1/ip/geo/'+ipAdd+'.json'
geo_requests=requests.get(url)
geo_data=geo_requests.json()
    #print(geo_data)
city=geo_data['city']
#state=geo_data['state']
country=geo_data['country']
    speak(f"sir i am not sure ,but i think we are in {city} city of {country}
country")      except Exception as e:      speak("sorry sir.Due to network
issue i am not able to find where we are.")      pass

elif 'joke' in query:
    speak(pyjokes.get_joke())

elif "instagram profile" in query or "profile on instagram" in query:
speak("sir please enter the user name correctly.")      name =
input("Enter username here: ")
webbrowser.open(f"www.instagram.com/{name}")
    speak(F"Sir here is the profile of the user (name)")
time.sleep(5)
    speak("sir would you like to download profile picture of this account.")
condition = takecommand().lower()

```

```
        if "yes" in condition:            mod
= instaloader.Instaloader()
        mod.download_profile(name, profile_pic_only=True)
        speak("i am done sir, profile picture is saved in our main folder. now i
am ready for next command")    else:        pass

        speak("sir,do you have any other work")
takecommand()
```