

A

Project Report On

**“Just An Artificial Intelligence Voice
Assistant”**

Submitted for the Degree of

Bachelor of Engineering

In

DEPARTMENT OF INFORMATION TECHNOLOGY

M.S.BIDVE ENGINEERING COLLEGE, LATUR

Affiliated to

Dr. Babasaheb Ambedkar Technological University, Lonere

Submitted By

Mr. Pawar Akshay M

Mr. Bhosale Atul R.

Mrs Kamble Mayawati V.

Under The Guidance Of

Prof. D.V.Biradar



Department of Information Technology

M.S. BIDVE ENGINEERING COLLEGE, LATUR

Academic Year :2022-2023

CERTIFICATE

This is to certify that the project titled “Just An Artificial Intelligence Voice Assistant” is a record of the bonafide work done by **Mr.Pawar Akshay , Mr.Bhosale Atul , Mrs.Kamble Mayawati, .** submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering (B.E) in **INFORMATION TECHNOLOGY** of M.S. Bidve Engineering College Latur ,during the academic year 2022-23.

Prof. Biradar D.V
Project Guide



Prof. Biradar D.V.
Head Of Dept

Prof. Dharne B.V.
Principal

Name & Signature of External Examiners with Date:

1. _____

2. _____

3. _____

ACKNOWLEDGEMENTS

I feel great pleasure in submitting this Project Report “Just An Artificial Intelligence Voice Assistant”. First of all I would like to sincerely thank my **Head of Department Prof. D.V. Biradar** for his invaluable suggestions from time to time, continuous encouragement, and moral support throughout the dissertation work. It was a great moment to work with him and really no words can convey regards to him.

I thank my honorable Principal **Dr. B. V. Dharne** of M.S. Bidve Engineering College, Latur for sponsoring me to undergo this B.E. course.

Most likely I would like to express my sincere gratitude towards my parents, my family and friends, for always being there with me. With all respect and gratitude, I would like to thank all the people, who have helped me directly or indirectly. Without their silent support and encouragement for this work it could not have been possible.

ABSTRACT

Title:Just An Artificial Intelligence Voice Assistant

Objective:

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 Alexei, Sri, 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 are 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.

Functionalities of this project include:

1. It can send emails.
2. It can read PDF.
3. It can send text on Whats App.
4. It can open command prompt, your favorite IDE, notepad etc.
5. It can play music.
6. It can do Wikipedia searches for you.
7. It can open websites like Google, YouTube, etc., in a web browser.
8. It can give weather forecast.
9. It can give desktop reminders of your choice.
10. It can have some basic conversation.

Now the basic question arises in mind that how it is an AI? The virtual assistant that I have created is like if it is not an A.I, but it is the output of a bundle of the statement. But fundamentally, the main purpose of A.I machines is that it can perform human tasks with the same efficiency or even more efficiently than humans. It is a fact that my virtual assistant is not a very good example of A.I., but it is an A.I.

DECLARATION

I hereby declare that the project work reported in this thesis has been carried out by me in the U. G. Department, M. S. Bidve Engineering College, Latur. I also declare that this work has not formed the basis for the award of any degree, diploma, fellowship, or similar title of any University or Institute to the best of my knowledge.

Mr.Pawar Akshay M.

Mr.Bhosale Atul R.

Mrs.Kamble Mayawati V.

Department (Information Technology)

M.S. Bidve Engineering College, Latur.

June 2023

TABLE OF CONTENTS

DECLARATION	I
ACKNOWLEDGEMENTS	II
ABSTRACT	III
TABLE OF CONTENT	IV
Chapter 1 INTRODUCTION	1
1.1 PROSENT SYSTEM	1
1.2 PROPOSED SYSTEM	1
 Chapter 2 SYSTEM DESIGN	 3
2.1 DATA FLOW	
Chapter 3 SOFTWARE DETAILS	4
3.1 VISUAL STUDIO CODE	
3.2 PYTHON LIBRARIES	
3.2.1 Pyttsx3	
3.2.2 Speech Recognition	
3.2.3 pywhatkit	
3.2.4 datetime	
3.2.5 wikipedia	
3.2.6 Smtplib	
3.2.7 pypdf2	
3.2.8. Pyjokes	
3.2.9. webbrowser	
3.2.10. pyautogui	
3.2.11. os	
3.2.12.sys	
3.3 IMPORTED MODULES	
 Chapter 4 IMPLEMENTATION WORK DETAILS	 7
4.1 REAL LIFE APPLICATION	

- 4.1.1 Saves time:**
- 4.1.2 Conversational interaction**
- 4.1.3 Reactive nature:**
- 4.1.4 Multitasking:**
- 4.1.5 NO Trigger phase**

4.2 DATA IMPLEMENTATION AND PROGRAM EXECUTION

4.2.1 LIBRARIES AND PACKAGES

- 4.2.1 Pyttsx3
- 4.2.2 Speech Recognition
- 4.2.3 pywhatkit
- 4.2.4 datetime
- 4.2.5 wikipedia
- 4.2.6 Smtplib
- 4.2.7 pypdf2
- 4.2.8. Pyjokes
- 4.2.9. webbrowser
- 4.2.10. pyautogui
- 4.2.11. os
- 4.2.12.sys

4.3 FUNCTIONS

- 4.3.1 takeCommand()**
- 4.3.2 wishMe**
- 4.3.3 task execution**

Chapter 5 SOURCE CODE AND CAMMANDS

9

MAIN FILE

Features file

Text files

Music files

GUI

Samples

Screenshots

Commands for Assistant:

1.Wikipedia:-To search about anything on wiki

Command:wikipedia 'person Name'/place '/Thing'

2.Time:

Command:Tell me the current time

3.Date:

Command:Today's Date?

4.Weather:

Command:Hows the weather Today?

choose city...?

5.News/Headline:Tell me news/Today's Headlines....

choode Topic.

6.Location:

location based on ip address

Command;what is our location finding

7.Where is :finding any place

command:where is anycity/place

8.Change password:

Command:it ask for password just type it done"

9.Schedule myday:

command: say yes/no

enter the number of tasks :1,2,3

eg type your schedule.....

done

10.Show/see schedules:

Show my schedules:

it will pop up schedule/do i have plans/notification with tell me today's plans/schedules/report my schedule/am i busy

11.Opening and Closing apps:

just say open app name:

close note app name:

12.Switch windows: command

13.play music/Stop music:

14.Google search:

Command:search on google.....anything

15.Youtube:

Play on youtube:say name of any video

16.Facebook/Instagram-gmail:

check instagram -command

17.Click photo:

capture/click/take/picture

18.battery percentage:

19.CPU:tell cpu to statics

20:Internet speed: tell me the internet speed

21.IPAddress:what is ip address

22.Rember that :

Command:i have class at 7pm

23.Reminder:is there any

reminder-command

24.Screenshot:

Command:take Screenshot

save name:

25.Show me the screenshot:

Command-tell the name

26.CloseScreenshot:screenshot terminate screenshot

27.Read Book:command

select page no: here you go

28.Set alarm:command

give time and wait for it

29.Send email: command

give receivers email-done

30.What is how to:ask anything

31.Searching mode :

command:Turn on searching mode search anything

32.who are you:

tell me about yourself/what can you do

33.Thank you: To stop program exit /quit/stop/exit/offline

Chapter 6 Input/output screenshot

28

6.1 Live GUI of JARVIS

6.2 Figure 6.2 Input for Google search

6.3 output google search

6.4 Input to send Email

6.5 Output to send Email

6.6 Input for You Tube

6.7 Output for YouTube search

6.8 Input to play music

6.9 Output to play music

6.10 Input to open cmd

6.11 output to open cmd

6.12 Input and output for Wikipedia search

6.13 Input to open Microsoft Office

6.14 Output to open Microsoft Office

Chapter 7 SYSTEM TESTING

35

7.1FUNCTIONALITY

7.1 Input through voice commands

7.2 Output

7.2USABILITY

7.3SECURITY

7.4STABILITY

Chapter 8 Individual Contribution

37

Chapter 9 CONCLUTION

9.1. LIMITATIONS

9.1.1. Security is somewhere an issue, there is no voice command encryption in thisproject.

9.1.2. Background voice can interfere

9.1.3. Misinterpretation because of accents and may cause inaccurate results.

9.1.4. JARVIS cannot be called externally anytime like other traditional assistantslike Google Assistant can be called just by saying, "Ok Google!"

9.2. SCOPE FOR FUTURE WORK

9.2.1. Make JARVIS to learn more on its own and develop a new skill in it.

9.2.2. JARVIS android app can also be developed.

9.2.3. Make more Jarvis voice terminals.

9.2.4. Voice commands can be encrypted to maintain security.

Chapter 1: Introduction

1. 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 are 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 Visual Studio Code IDE for making this project, and I created all py files in Visual Studio Code. Along with this I used following modules and libraries in my project. pyttsx3, SpeechRecognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, PyQt etc. I have created a live GUI for interacting with the JAIVA as it gives a design and interesting look while having the conversation.

1.1 PRESENT SYSTEM

We are familiar with many existing voice assistants like Alexa, Siri, Google Assistant, Cortana which uses concept of language processing, and voice recognition. They listens 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.

1.2 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. JAIVA is different from other traditional voiceassistants 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 Visual Studio Code. All the python files were created in Visual Studio Code and all the necessary packages were easily installable in this IDE. For this project following modules and libraries were used i.e. pyttsx3, SpeechRecognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, PyQt etc. I have created a live GUI for interacting with the JAIVA as it gives a design and interesting look while having the conversation.

With the advancement JAIVA 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. Functionalities of this project 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

Chapter 2: System Design

2.1. DATA FLOW

The data flow for JAIVA is as follow:

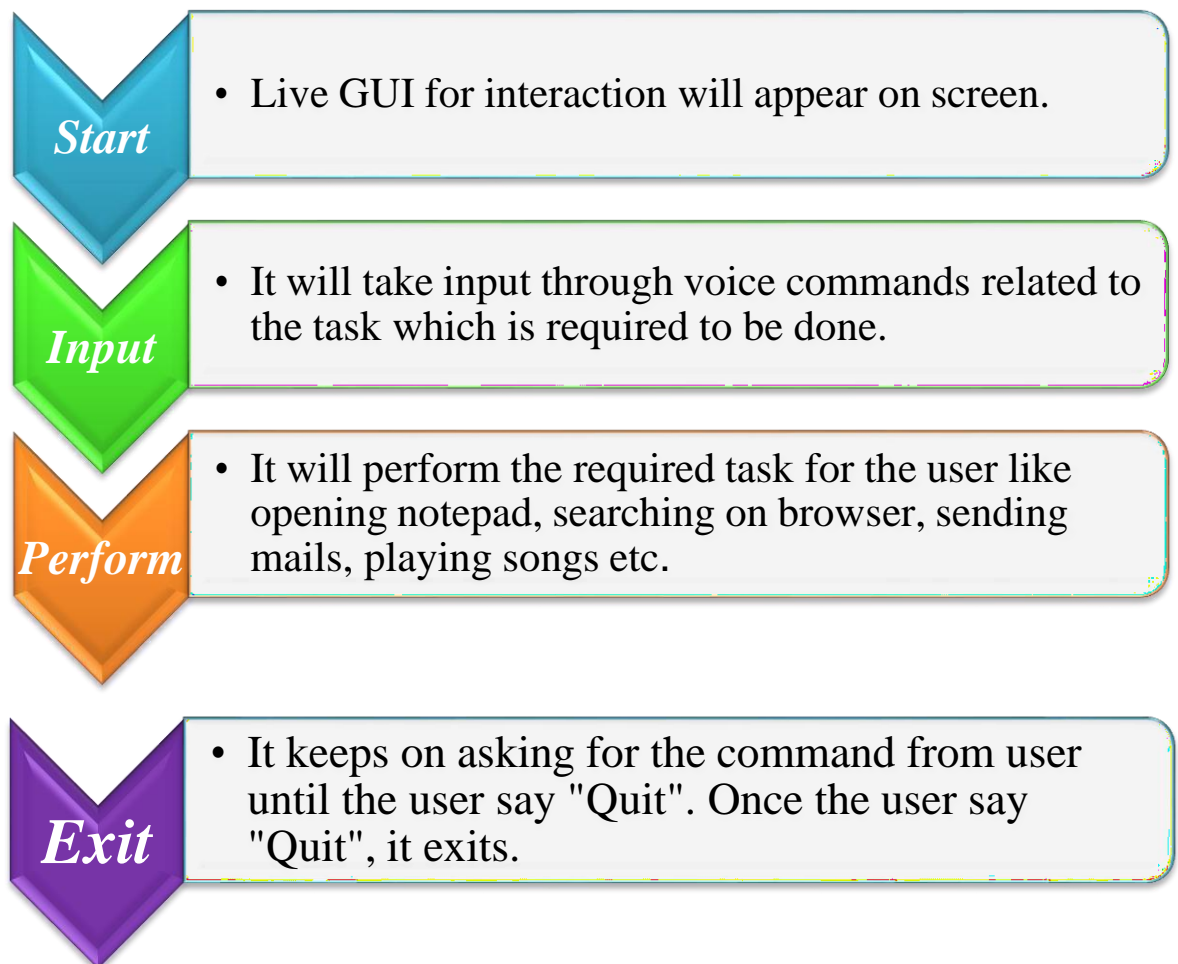


Figure 2.1 Data flow for JAIVA

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 pyPDF2 can be used to read PDF. The details of these packages are mentioned in Chapter 3 of this report.

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.

Chapter 3: Software Details

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

3.1. Visual Studio Code

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 pythondebugger, code completion, code and project navigation etc. It also provides Data Science when used with Anaconda.

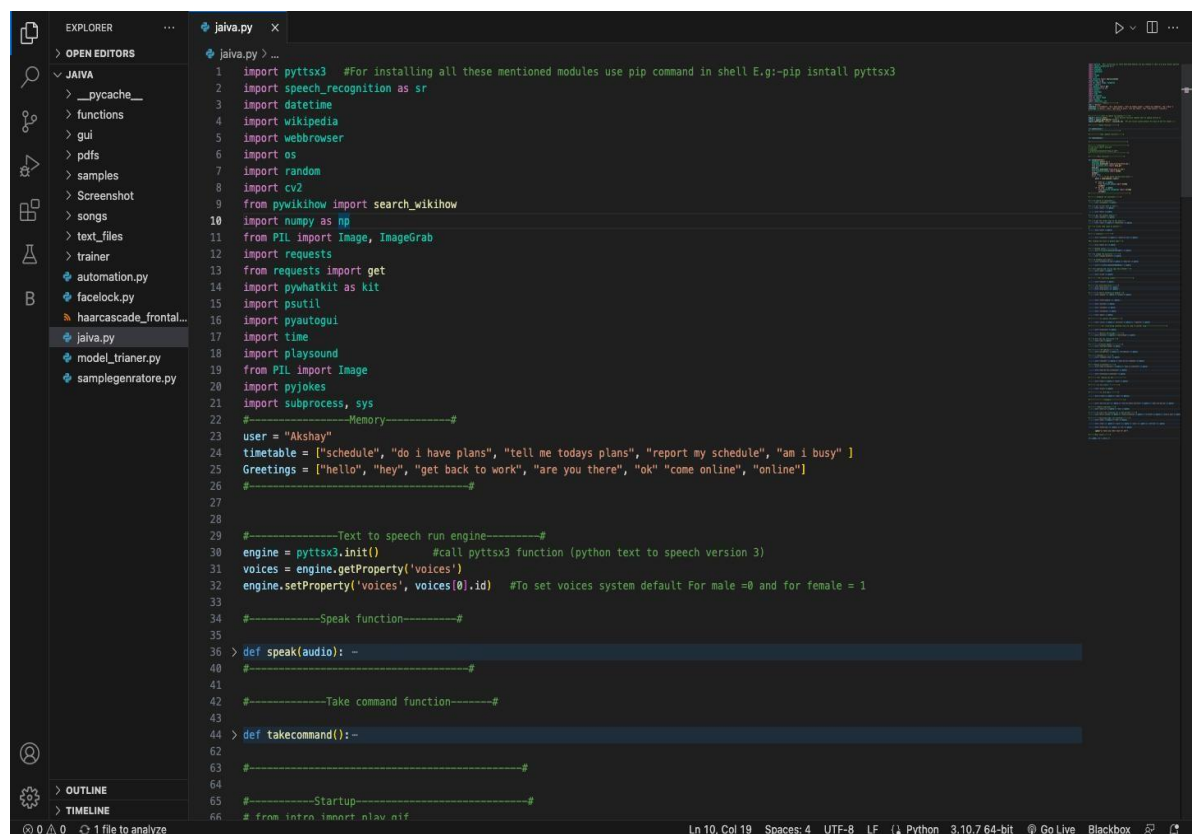


Figure 3.1 Visual Studio Code IDE

3.2. PYTHON LIBRARIES

In JAIVA following python libraries were used:

3.2.1. pyttsx3: It is a python library which converts text to speech.

3.2.2. SpeechRecognition: It is a python module which converts speech to text.

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

3.2.4. Datetime: This library provides us the actual date and time.

3.2.5. Wikipedia: It is a python module for searching anything on Wikipedia.

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

3.2.7. pyPDF2: It is a python module which can read, split, merge any PDF.

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

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

3.2.10. Pyautogui: It is a python libraries for graphical user interface.

3.2.11. os: It represents Operating System related functionality.

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

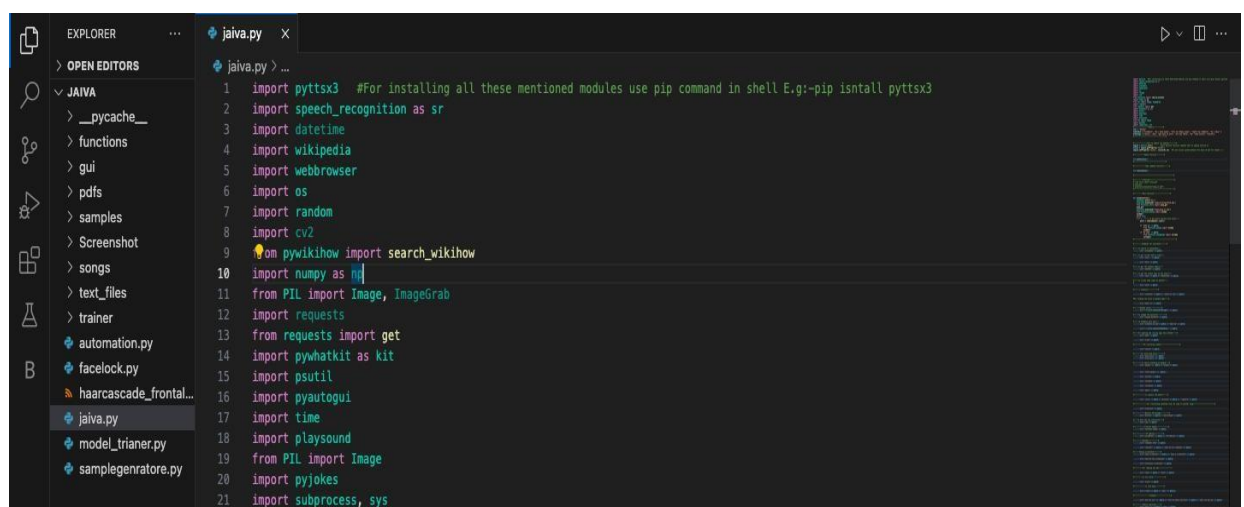


Figure 3.3 Imported Modules

Chapter 4: Implementation Work Details

JAIVA, 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. JAIVA 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.

4.1. REAL LIFE APPLICATION

4.1.1. Saves time: JAIVA 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.

4.1.2. 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.

4.1.3. 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.

4.1.4. 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.

4.1.5. 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.

4.2. 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:

4.2.1. LIBRARIES AND PACKAGES

4.2.2.1. pyttsx3: It is a python library which converts text to speech.

4.2.2.2. SpeechRecognition: It is a python module which converts speech to text.

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

4.2.2.4. Datetime: This library provides us the actual date and time.

4.2.2.5. Wikipedia: It is a python module for searching anything on Wikipedia.

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

4.2.2.7. pyPDF2: It is a python module which can read, split, merge any PDF.

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

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

4.2. 2.10. Pyautogui: It is a python librariy for graphical user interface.

4.2.2.11. os: It represents Operating System related functionality.

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

4.2.2. FUNCTIONS

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

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

4.2.2.3. taskExecution(): This is the function which contains all the necessary task execution definition like sendEmail(), pdf_reader(), news() and many conditions in if condition like “open google”, “open notepad”, “search on Wikipedia” ,”play music” and “open command prompt” etc.

Chapter 5: Source Code and Commands

5.1.1. Source Code:-

```
import pyttsx3 #For installing all these mentioned modules use pip command
in shell E.g:-pip install pyttsx3
import speech_recognition as sr
import datetime
import wikipedia
import webbrowser
import os
import random
import cv2
from pywikihow import search_wikihow
import numpy as np
from PIL import Image, ImageGrab
import requests
from requests import get
import pywhatkit as kit
import psutil
import pyautogui
import time
import playsound
from PIL import Image
import pyjokes
import subprocess, sys
#.....Memory.....#
user = "Akshay"
timetable = ["schedule", "do i have plans", "tell me todays plans", "report
my schedule", "am i busy" ]
Greetings = ["hello", "hey", "get back to work", "are you there", "ok"
"come online", "online"]
#.....#

#-----Text to speech run engine-----#
engine = pyttsx3.init() #call pyttsx3 function (python text to speech
version 3)
voices = engine.getProperty('voices')
```

```
engine.setProperty('voices', voices[0].id) #To set voices system default
For male =0 and for female = 1
```

```
#-----Speak function-----#
```

```
def speak(audio):
engine.say(audio)
print(audio)
engine.runAndWait()
#-----#
```

```
#-----Take command function-----#
```

```
def takecommand():
r = sr.Recognizer()
with sr.Microphone() as source:
print("Listening...")
r.pause_threshold = 2 #for pause in listening
r.energy_threshold = 300
audio = r.listen(source,0,4)
```

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

```
except Exception as e:
speak("Say that again please...")
return takecommand()
return query
```

```
#-----#
```

```
#-----Startup-----#
# from intro import play_gif
# play_gif
# playsound.playsound("plug_in.mp3")
#-----#
```

```
#-----Main Function-----#
```

```
def taskexecution():
```

```

pyautogui.press('Esc')
playsound.playsound('songs/accessgranted.mp3')
from functions.intro import play_gif
play_gif
playsound.playsound("songs/plug_in.mp3")
from functions.wakeup import wishme
wishme()
while True:
# if 1 : ----> to get query execute only once<----
query = takecommand().lower()

```

```

if 'wake up' in query:
from functions.wakeup import wishme
wishme()
if 'my name' in query:
from functions.changename import username
username()
#-----#

```

```

#-----Commands for assistant-----#

```

```

#--> To search on wikipedia<--
elif 'wikipedia' in query:
try:
speak('Searching on wikipedia.....')
query = query.replace('wikipedia', '')
results = wikipedia.summary(query, sentences=2)
speak(f"according to wikipedia:- {results}")
except Exception as e:
speak("Say that again please.....")

```

```

#--> to get current date & time<--
elif 'time' in query:
try:
from datetime import datetime
now = datetime.now()
current_time = now.strftime("%I:%M:%S %p")
speak("Sir Current Time is: " + current_time)
except Exception as e:
speak("I cant get you!")

```

```

elif 'date' in query:
try:
from datetime import datetime
speak("Todays date is " + str(datetime.now().day)
+ " " + str(datetime.now().month)
+ " " + str(datetime.now().year))
except Exception as e:
speak("I cant get you!")
#---> to get the weather Report<---
elif 'weather' in query:
try:
from functions.weather import weather
weather()
except Exception as e:
speak("Say that again please...")

```

```

#---> to get the latest news on any topic<---
elif "news" in query or "headlines" in query:
try:
from functions.news import news
news()
except Exception as e:
speak("Try after some time..!")
#---> to listen some jokes by pyhton<---
#
elif "joke" in query:
speak(pyjokes.get_joke())

```

```

#.....Location.....#

```

```

elif "location" in query or 'where we are' in query:
try:
from functions.loc import MY_loc
MY_loc()
except Exception as e:
speak("Say that again please...")

```

```

#For finding the place on google maps-----#

```

```

elif "where is" in query:
try:

```

```

query = query.replace("where is", "")
location = query
speak(f"Locating {location}, Please wait sir")
webbrowser.open("https://www.google.com/maps/place/" + location + "")
except Exception as e:
speak("sorry sir. I could not locate where it is please say it again...")

```

```

#.....Random words.....#
elif f'{random.choice(Greetings)}' in query:
try:
from functions.fun import funtime
funtime()
except Exception as e:
speak("Say that again please...")

```

```

#-----to change the password-----#
elif "change password" in query:
try:
speak("Whats the new password?")
newPass= input("Enter the new password:")
newPassword = open("text_files/pass.txt","w")
newPassword.write(newPass)
newPassword.close()
speak("Done Sir! Your password has been changed successfully!")
except Exception as e:
speak("Password not changeable please try again")
#---> to Schedule your day<---
elif "schedule my day" in query or "make my" in query:
try:

```

```

tasks = []
speak("Do you want to clear all old schedules? Say Yes / No")
query = takecommand().lower()
if "yes" in query:
file = open("text_files/schedules.txt", "w")
file.write(f"")
file.close()
no_tasks = int(input("Enter the Number of tasks: "))
i = 1
for i in range(no_tasks):

```



```

tasks.append(input("Enter the task: "))
file = open("text_files/schedules.txt", "a")
file.write(f"{i}. {tasks[i]}\n")
file.close()
elif "no" in query:
i = 1
no_tasks = int(input("Enter the Number of tasks: "))
for i in range(no_tasks):
tasks.append(input("Enter the task: "))
file = open("text_files/schedules.txt", "a")
file.write(f"{i}. {tasks[i]}\n")
file.close()
speak("Your task has been saved successfully!")
except Exception as e:
speak("please say it again...")

```

```

elif f'{random.choice(timetable)}' in query:
file = open("text_files/schedules.txt")
content = file.read()
file.close()

```

```

title = "Schedules"
command = f'''
osascript -e 'display notification "{content}" with title "{title}"'
'''
os.system(command)
playsound.playsound("songs/noti.mp3")

```

```

#-----for opening and closing apps Easy method----- #
elif 'open' in query:
try:
query = query.replace("open", "")
pyautogui.hotkey("command", "space")
pyautogui.typewrite(query)
pyautogui.sleep(2)
pyautogui.press("enter")
speak(f"{query} opened!")
except Exception as e:
speak("Say it again please.....")

```

```

elif 'close' in query:
try:
query = query.replace("close","")
speak(f"Ok sir! Closing {query}...")
pyautogui.hotkey("Option","command","w")
pyautogui.sleep(2)
except Exception as e:
speak("Say that again please...")

```

```

#-----For switching window-----#

```

```

elif "switch" in query:
try:
speak("Switching window")
pyautogui.keyDown("command")
pyautogui.press("tab")
time.sleep(1)
pyautogui.keyUp("command")
except Exception as e:
speak("try again...")
#-----For play/stop music-----#
elif 'play music' in query:
try:
speak("Enjoy the music..")
subprocess.call(["/usr/bin/open", "-W", "-n", "-a",
"/Applications/Resso.app"])
except Exception as e:
#print(e)
speak('Sorry sir, I am not able to play music')
elif 'stop music' in query:
os.system("pkill Resso")

```

```

#-----to search anything on google-----#
elif 'google' in query or "Google" in query:
try:
speak("sir, what should i search on google")
webbrowser.open("https://www.google.com/search?q=" +takecommand())
speak("Here is your search")
except Exception as e:
speak("Say that again please.....")

```

```
elif 'close google' in query:
os.system("pkill Chrome")
```

```
elif 'youtube' in query:
try:
speak("What you want to search on youtube, sir!")
kit.playonyt(takecommand())
speak("Enjoy Your Time, sir!")
except Exception as e:
speak("Say that again please...")
```

```
elif 'facebook' in query:
try:
controller = webbrowser.get()
controller.open("https://www.facebook.com/")
except Exception as e:
speak("Say that again please...")
```

```
elif 'instagram' in query:
try:
controller = webbrowser.get()
controller.open("https://www.instagram.com/")
except Exception as e:
speak("Say that again please...")
elif 'gmail' in query:
try:
controller = webbrowser.get()
controller.open("https://accounts.google.com/AccountChooser/identifier?flow
Name=GlifWebSignIn&flowEntry=AccountChooser")
except Exception as e:
speak("Say that again please...")
```

```
#-----to capture the photo-----#
```

```
elif 'click' in query or "picture" in query or "capture" in query:
try:
pyautogui.hotkey("command", "space")
pyautogui.typewrite("photo Booth")
pyautogui.sleep(2)
```

```

pyautogui.press("enter")
pyautogui.sleep(2)
speak("Smile please!")
pyautogui.hotkey("command", "shift", "t")
speak("Looking nice!!")
pyautogui.sleep(8)
os.system("pkill Photo Booth")
except Exception as e:
speak("Try it again please...")

```

```

#-----for translating anything from one lang to another lang-----
#

```

```

elif "translate" in query:
try:
from functions.trans import translategl
query = query.replace("translate","")
translategl(query)
except Exception as e:
speak("Say it again please!...")

```

```

#.....Battery Percentage.....#
elif "battery" in query or "percentage" in query:
try:
battery_per = psutil.sensors_battery().percent
speak(f"Power is {battery_per}% remaining..")
if int(battery_per)==30 :
speak('Your system is low of power.. please charge')
elif int(battery_per)<10 :
speak('Your system is very low of power.. please charge otherwise system
willl shutdown very soon..')
else:
speak('your System has enough Power!')
except Exception as e:
speak("Sorry.. say it again.")

```

```

#--- to know the cpu statistics-----#
elif "cpu" in query:
try:
speak(f"Cpu is running at {str(psutil.cpu_percent())} %")

```

```
except Exception as e:
speak("try again...")
```

```
#.....Internet Speed.....#
elif 'internet speed' in query:
try:
i_s = ((psutil.net_io_counters().bytes_recv +
psutil.net_io_counters().bytes_sent )/psutil.net_io_counters().bytes_sent)
st_up = (psutil.net_io_counters().bytes_recv/psutil.net_io_counters().bytes_sent)
st_dw = (psutil.net_io_counters().bytes_sent/psutil.net_io_counters().bytes_recv)
speed = (" {:.3f}".format(i_s))
st_dw_speed= (" {:.2f}".format(st_dw))
st_up_speed= (" {:.2f}".format(st_up))
speak(f"Your internet speed is: {speed} MB per second, and\n upload speed
is: {st_up_speed} MB per second\n download speed is: {st_dw_speed} MB per
second")
except Exception as e:
speak("Cant access the internet speed right now please say it again..")
#.....IP address.....#
elif 'ip address' in query or "IP address" in query:
try:
ip = get("https://api.ipify.org").text
speak(f"Your IP address is: {ip}")
except Exception as e:
speak("Say that again please...")
```

```
#.....reminder.....#
elif 'remember that' in query:
try:
rememberMsg = query.replace('remember that','')
speak("i successfully remembered that"+rememberMsg)
rmbr=open("text_files/reminder.txt","w")
rmbr.write(rememberMsg)
rmbr.close()
except Exception as e:
speak("Say that again please...")
elif "reminder" in query or "what do you remember" in query:
try:
```

```

rmbr = open("text_files/reminder.txt","r")
speak("yes sir..You told me to remind that:"+rmbr.read())
except Exception as e:
speak("Say that again please...")

```

```

#-----Taking screenshots-----#
elif "take screenshot" in query or "take a screenshot" in query:
try:
speak("By what name do you want to save the screenshot?")
name = takecommand().lower()
speak("Alright sir, taking the screenshot")
img = pyautogui.screenshot()
name = f"Screenshot/{name}.png"
img.save(name)
speak("The screenshot has been succesfully captured")
except Exception as e:
speak("Say that again please.....")

```

```

elif "show me the screenshot" in query:
try:
speak("Which screen shot do you want to see?")
name = takecommand()
img = Image.open(f'Screenshot/{name}.png')
img.show(img)
speak("Here it is sir")
time.sleep(2)
except IOError:
speak("Sorry sir, I am unable to display the screenshot")

```

```

elif "terminate screenshot" in query:
speak("screenshot terminated..")
os.system("pkill Preview")

```

```

#.....For reading any pdf.....

```

```

elif "read" in query or "book" in query:
try:
from functions.pdfreader import pdfread
pdfread()
except Exception as e:
speak("Say that again please...")

```

```
#.....To set alarm.....#
elif 'alarm' in query:
try:
speak("Please Enter the time to set the alarm")
print("Time format: HR:MM AM/PM")
almr = input("Please Enter the time: ")
from functions import alarm
alarm.alarm(almr)
except Exception as e:
speak("Say that again please...")
#.....to send email.....#
elif ('email'in query or 'mail' in query):
try:
from functions.sendmail import sendEmail
speak('Please enter the email of receiver:')
to = input("Enter the email id of reciever: ")
speak(f'what should I say?')
content = takecommand().lower()
sendEmail(to, content)
speak(f"Email has been sent!!")
except Exception as e:
speak("sorry sir. I am not able to send this email.. try once again.")
```

```
#.....Timepass..... #
```

```
elif 'who are you' in query or 'tell me about yourself' in query or 'what
can you do' in query:
try:
speak('My Name is JAIVA, A Virtual Assistant..Mr. Akshay Created me in
python for performing some tasks by voice commands.I can do tasks like
Opening Google, Play music, Open Youtube, Can tell current time. Oepening
notes and texteditor for you,and also Open applications like whatsapp,
Music Applications, tell weather reports tell news and many more.. i can
search wikipidea for any information u ask for and can google
anything..locate any place on maps etc just give me commands and i will
give you the responce accodingly..')
except Exception as e:
speak("sorry sir. pardon please..")
```

```
#-----Search anything-----#
elif 'what is' in query or 'how' in query:
try:
url          =          "https://www.google.co.in/search?q="          +(str(query))+
"&oq="+ (str(query))+"&gs_l=serp.12..0i7118.0.0.0.6391.0.0.0.0.0.0.0.0.0..0.0..
..0...1c..64.serp..0.0.0.UiQhpfaBsuU"
webbrowser.open_new(url)
time.sleep(2)
speak('Here is your answer')
time.sleep(5)
except Exception as e:
speak('Sorry, I am unable to answer it. say it again')
```

```
#-----to stop from listening for a time period-----#
elif "don't listen" in query or "stop listening" in query or "a break" in
query or "take a nap" in query or "wait" in query :
try:
speak("for how much time you want to stop me from listening commands")
print('please give the time only in integer form for eg: 10, 20, 30,
40.....100')
a = int(takecommand())
speak(f"Ok sir., i am taking a nap for {a} seconds")
time.sleep(a)
speak("it was a nice sleep, The time is over sir, i am back to work")
except Exception as e:
speak("Plese say it once again!")
```

```
#-----searching mode- ask anything-----#
elif 'mode' in query or 'mod' in query:
try:
speak('searching mod turned on.. Please tell what can i do for you?')
com = takecommand().lower()
com1 = com.replace("tell me","")
com1 = com.replace("the","")
mr = 1
how_to = search_wikihow(com1,mr)
assert len(how_to)== 1
how_to[0].print()
speak(how_to[0].summary)
except Exception as e:
```



```
speak('could not switch to Searching mode...please say that again..')
```

```
elif 'stop' in query or 'quit' in query or 'exit' in query or 'offline' in query:
    try:
        from datetime import datetime
        hour = datetime.now().hour
        if (hour >= 18) and (hour < 24):
            speak(f"Okay {user} Sir! Good Night, Have a nice sleep!")
        else:
            speak(f"Okay {user} sir! Have a nice day! ")
        quit()
        sys.exit()
    except Exception as e:
        speak("say that again.")
elif 'thank you' in query or 'no' in query:
    try:
        speak('Welcome sir! , if you need help, Call me anytime')
        break
    except Exception as e:
        speak("sorry sir. say it again.")
```

```
speak("is there any other task for me?")
```

```
#.....Main class.....#
```

```
if __name__ == "__main__":
```

```
# ----->Password protection< -----#
speak("Please Enter Your password to Run the program..!")
try:
    for i in range (3):
        print("You have only 3 attempts please be carefull..")
        a= input("Enter Your Password Here: ")
        pw_file = open("text_files/Pass.txt","r")
        pw = pw_file.read()
        pw_file.close()
        if(a==pw):
            # print("Welcome sir! Say Wake up to start..")
            break
    elif (i==2 and a!=pw):
```

```

exit()
elif(a!=pw):
speak("Password is incorrect! Try Once Again..")
except Exception as e:
speak('Try again..')

```

```

# recognizer =cv2.face.LBPHFaceRecognizer_create() #LLocal Binary Pattern
Histogram
# recognizer.read("trainer/trainer.yml") #load trained model
# cascadePath = "haarcascade_frontalface_default.xml" #cascade file path
# faceCascade = cv2.CascadeClassifier(cascadePath) #initializing haar
casdc for object detection

```

```

# font = cv2.FONT_HERSHEY_SIMPLEX #denotes the font type

```

```

# id = 6 #no of perosons you want to recognize

```

```

# names = ['', 'akki'] #names, leave first empty because counter starts from
0

```

```

# cam = cv2.VideoCapture(0) #to remove warning

```

```

# cam.set(3, 640) #set video frame width
# cam.set(4, 480) #set video frame height

```

```

# #define main window size to be recognized as a face

```

```

# minW = 0.1*cam.get(3)
# minH = 0.1*cam.get(4)

```

```

# while True:
# ret, img = cam.read() # read the frame using the above created object
# if not ret:
# print("Can't receive frame. (stream end?). Exiting ...")
# break
# converted_image =cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #converts input
into grayscale image

```

```

# faces = faceCascade.detectMultiScale(

```

```
# converted_image,
# scaleFactor = 1.2,
# minNeighbors = 5,
# minSize = (int(minW), int(minH)),
# )
```

```
# for(x,y,w,h) in faces:
# cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0),2) #used to draw a
rectangle on any image
```

```
# id, accuracy = recognizer.predict(converted_image[y:y+h,x:x+w]) #to
predict on every single image
```

```
# #check if accuracy is less than 100 ==> 0 is perfect match
```

```
# if (accuracy < 100):
# # id = names[id]
# accuracy = " {0}%".format(round(100 - accuracy))
# taskexecution()
# else:
# id = "unknown"
# accuracy = " {0}".format(round(100-accuracy))
```

```
# cv2.putText(img, str(id), (x+5,y-5), font, 1, (255,255,255),2)
# cv2.putText(img, str(accuracy), (x+5,y+h-5), font, 1, (255,255,0),1)
```

```
# cv2.imshow('camera', img)
```

```
# k= cv2.waitKey(10) & 0xff #press 'ESC' for exiting video
# if k == 27:
# break
# print("Thanks for Using this program, Have a good day...")
# cam.release()
# cv2.destroyAllWindows()
```

```
taskexecution()
```

5.1.2 Commands :- Main file

Features file :- Text files:- Music files:- GUI:-Samples:- Screenshots

Commands for Assistant:

1. **Wikipedia:-** To search about anything on wiki

Command: wikipedia 'person Name'/place '/Thing'

2. **Time:** To find the current time

Command: Tell me the current time

3. **Date:** To find the current date

Command: Today's Date?

4. **Weather:**

Command: Hows the weather Today?

choose city...?

5. **News/Headline:** Tell me news/Today's Headlines....

choose Topic.

6. **Location:**

location based on ip address

Command: what is our location finding

7. **Where is:** finding any place

command: where is anycity/place

8. **Change password:**

Command: it ask for password just type it done"

9. **Schedule myday:**

command: say yes/no

enter the number of tasks :1,2,3

eg:- type your schedule.....

done

10. Show/see schedules:

Show my schedules:

it will pop up schedule/do i have plans/notification with tell me today's plans/schedules/report my schedule/am i busy

11. Opening and Closing apps:

just say open app name:

close notes app name:

12. Switch windows:

command: Switch the window

13. play music/Stop music: play music/stop music.

14. Google search:

Command: search on google anything

15. Youtube:

command: Play on youtube: say name of any video

16. Facebook/Instagram-gmail:

check instagram -command

17. Click photo:

capture/click/take/picture

18. battery percentage:

19. CPU: tell cpu to statics

20: Internet speed: tell me the internet speed

21. IP Address : what is ip address

22. Remember that :

Command: i have class at 7pm

23. Reminder: is there any reminder-command

24. Screenshot:

Command:take Screenshot

save name:

25. Show me the screenshot:

Command:t ell the name

26.CloseScreenshot:terminate screenshot

27.Read Book:command

select page no: here you go

28.Set alaram:command

give time and wait for it

29.Send email: command

give receivers email-done

30. What is how to:ask anything

31. Searching mode :

command:Turm on searching mode search anything

32.who are you:

tell me about yourself/what can you do

33.Thank you: To stop program exit /quit/stop/exit/offline

Chapter 6: Input/Output Screenshot



Figure 6.1 Live GUI of JAIVA

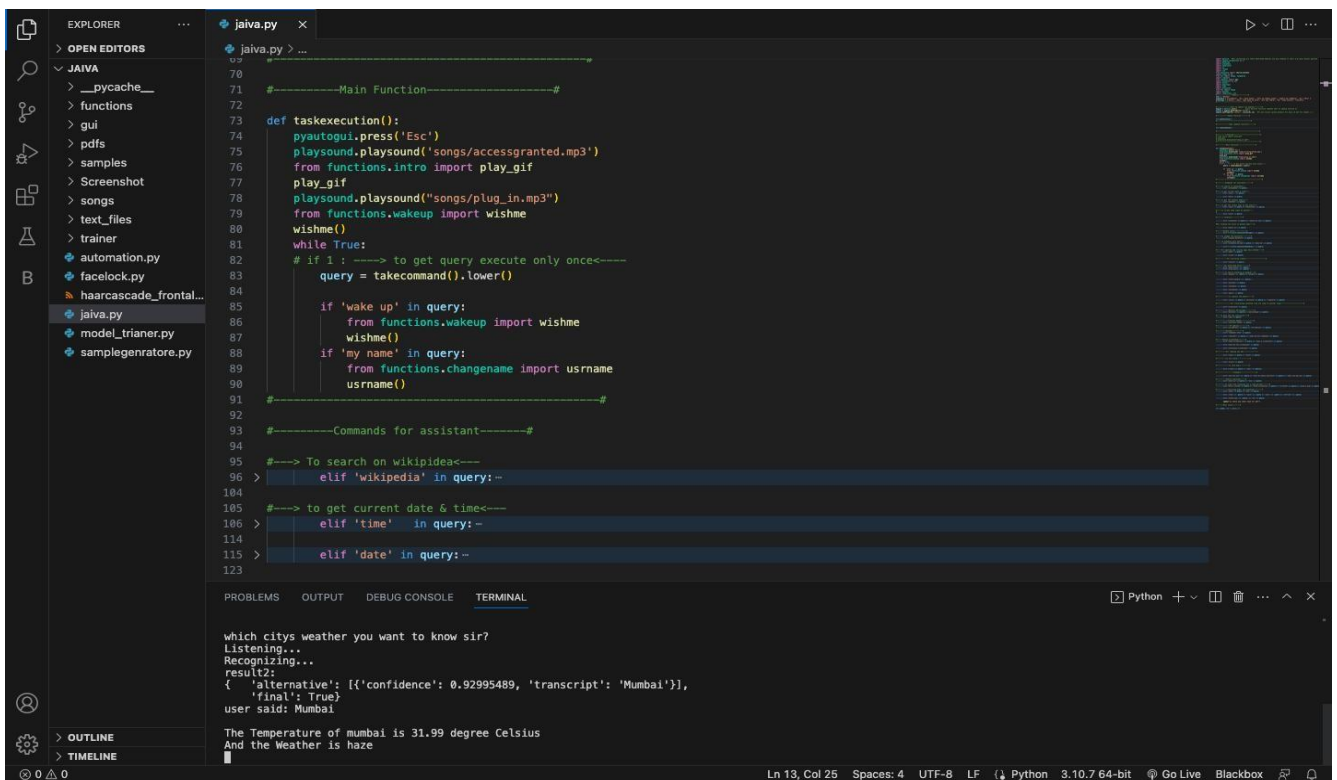


Figure 6.2 Input for Google search

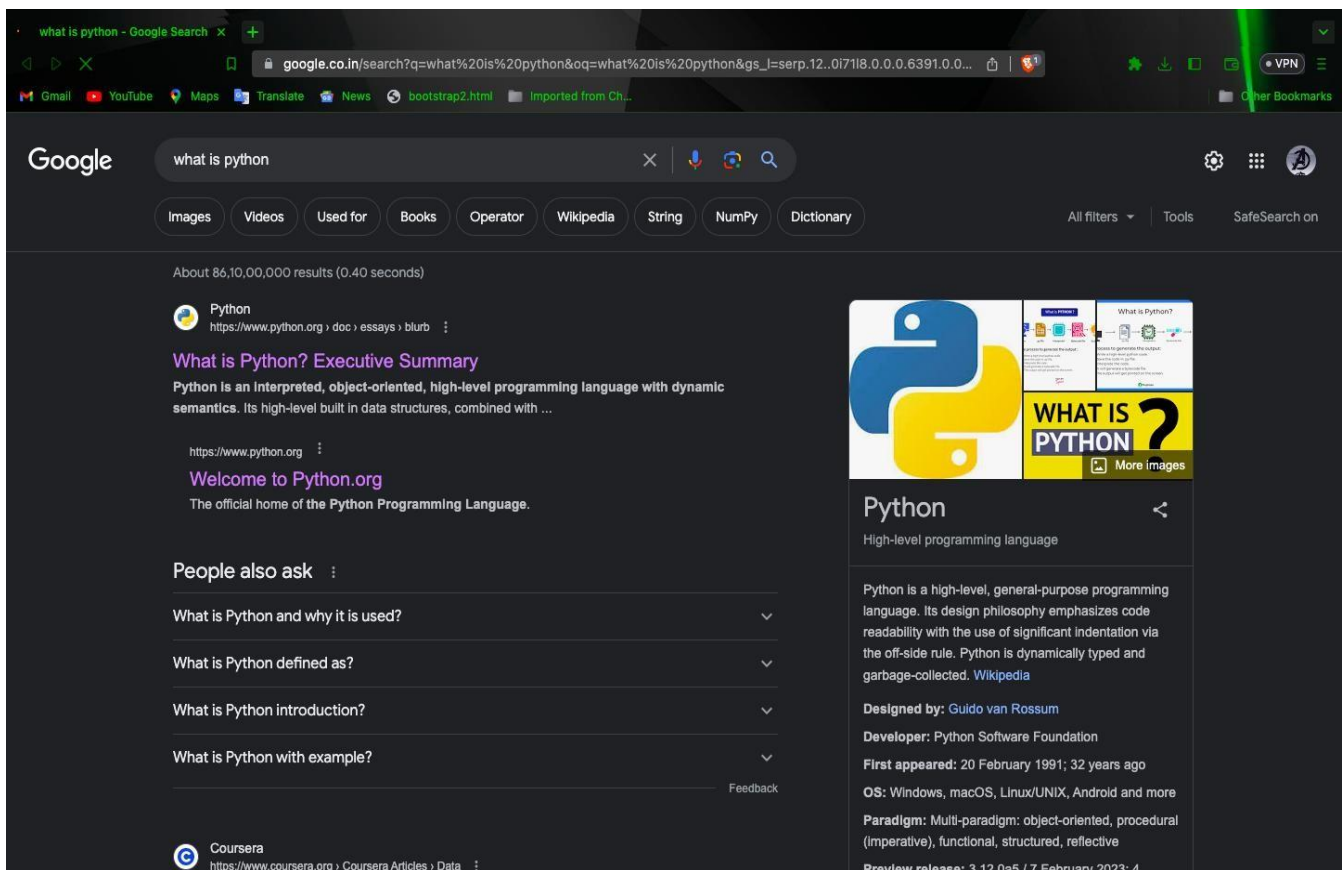


Figure 6.3 Output for Google search

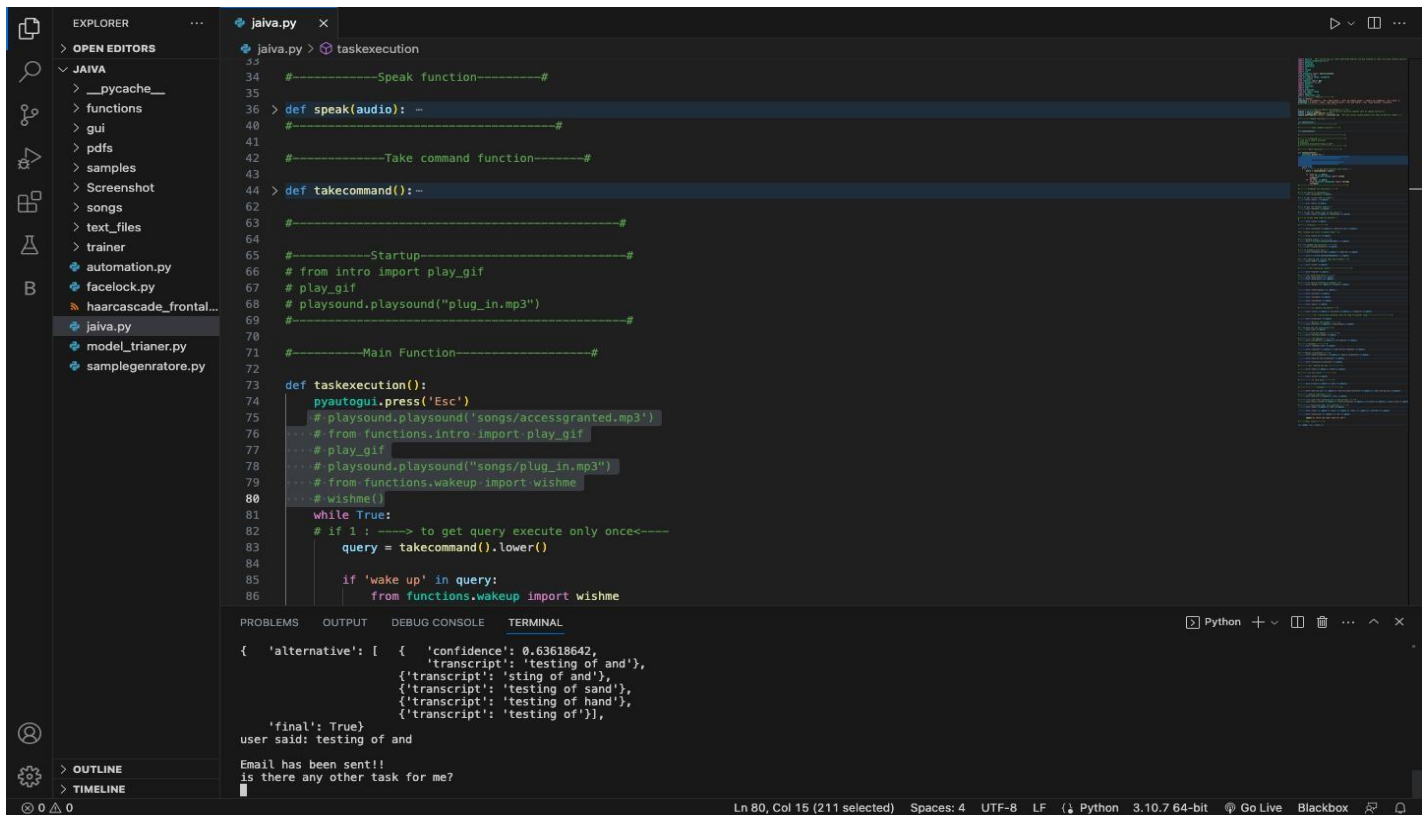


Figure 6.4 Input to send Email

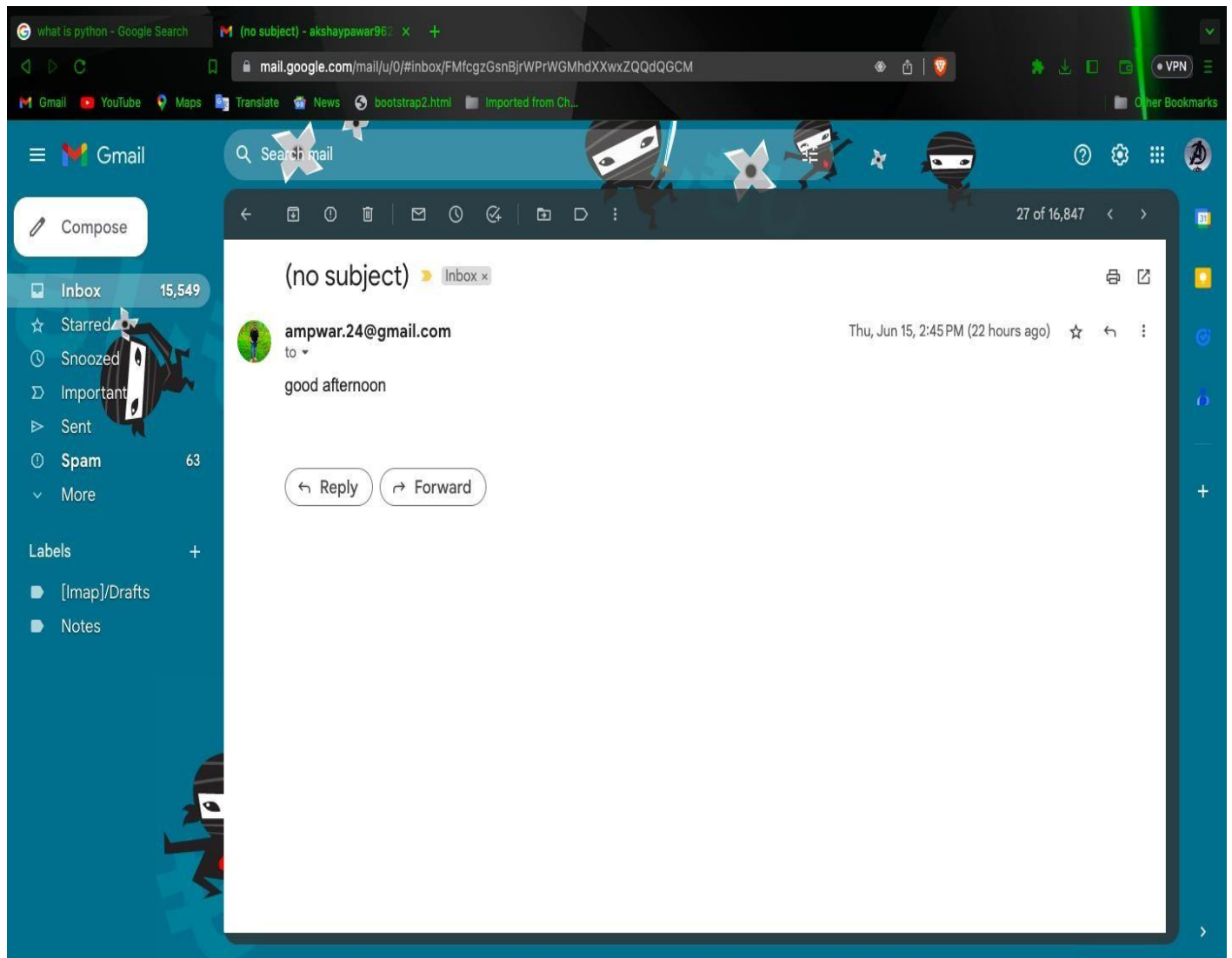


Figure 6.5 Output to send Email

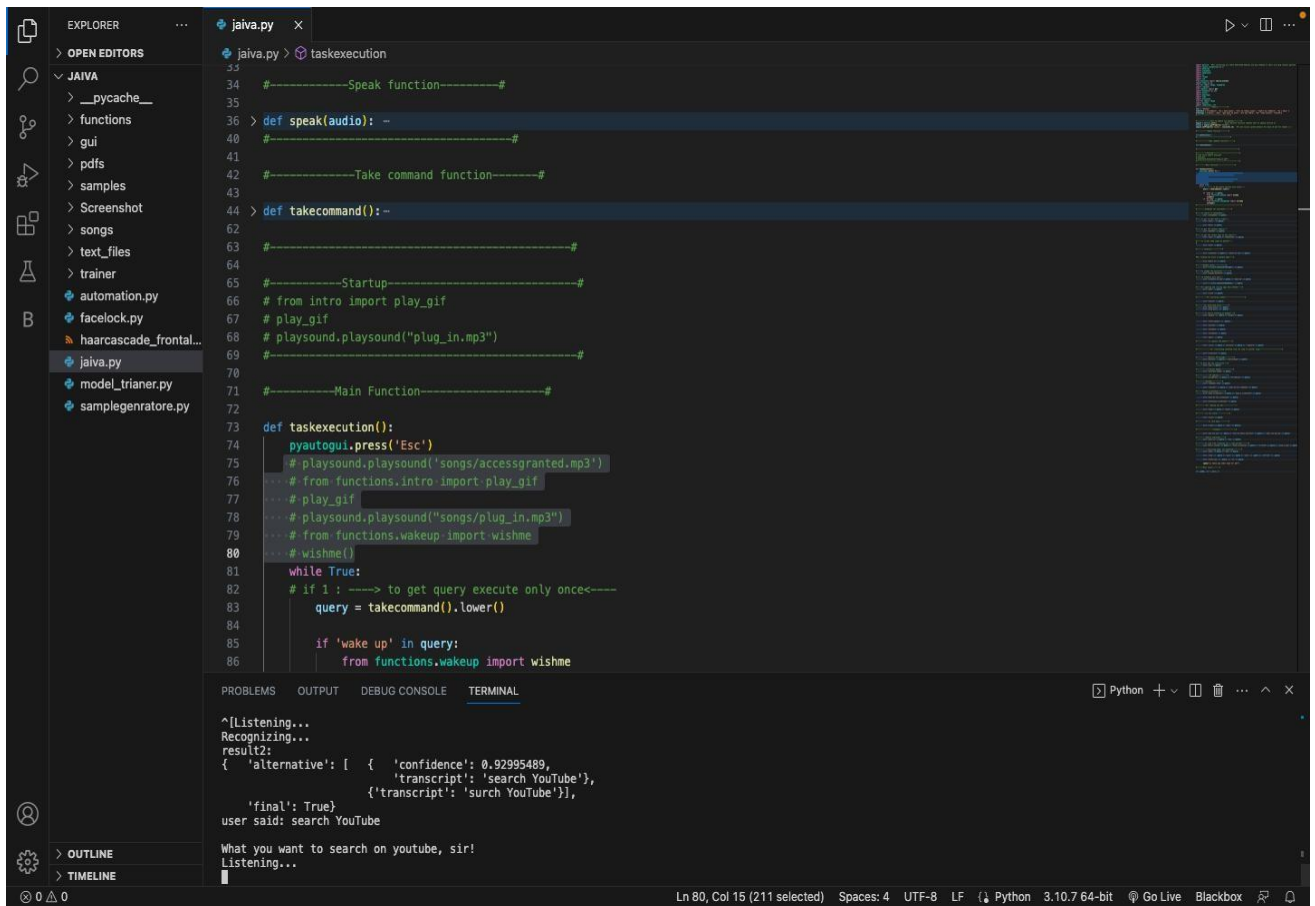


Figure 6.6 Input for YouTube search

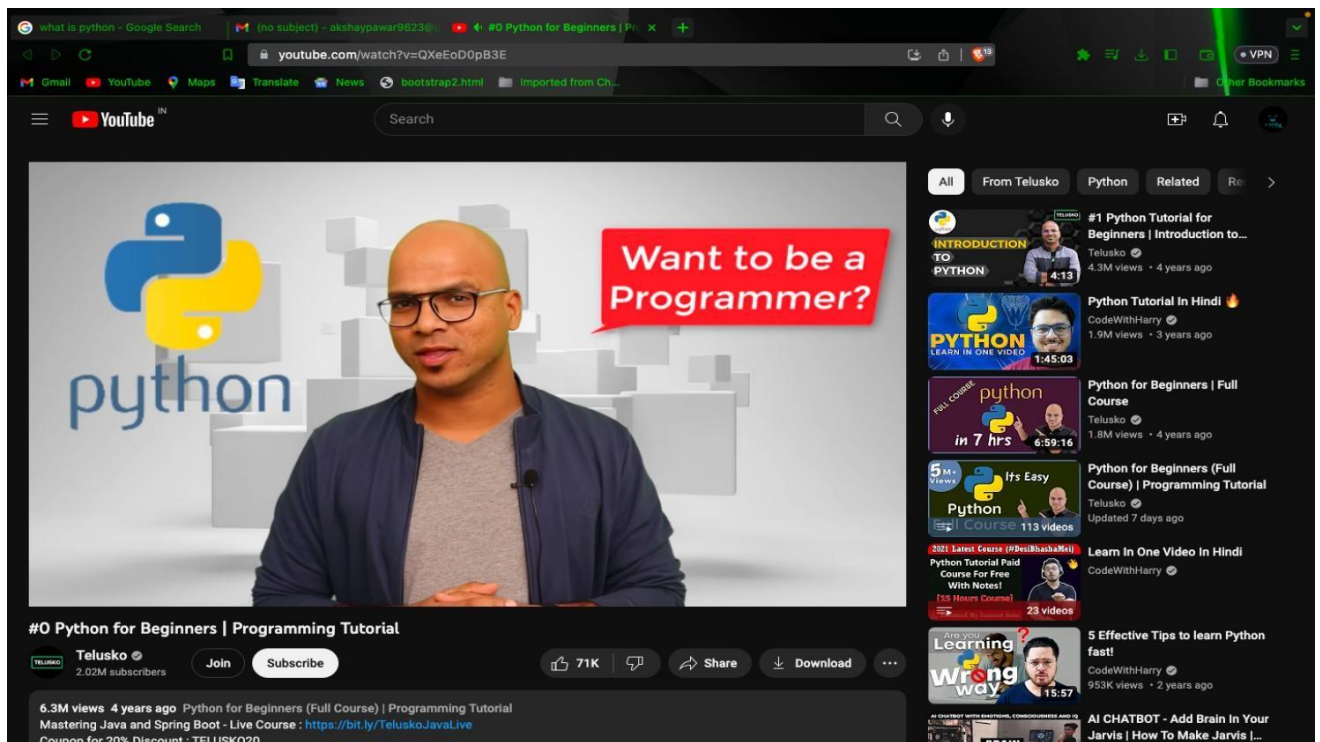


Figure 6.7 Output for YouTube search

```

474 #-----Search anything-----#
475 > elif 'what is' in query or 'how' in query:-
485
486 #-----to stop from listening for a time period-----#
487 > elif 'don't listen' in query or 'stop listening' in query or 'a break' in query or 'take a nap' in query or 'wait' in query :-
497
498 #-----searching mode- ask anything-----#
499 > elif 'mode' in query or 'mod' in query:-
512
513 > elif 'stop' in query or 'quit' in query or 'exit' in query or 'offline' in query:-
525
526 > elif 'thank you' in query or 'no' in query:-
532
533     speak("is there any other task for me?")
534
535 #-----Main class-----#
536
537 > if __name__ == "__main__":-
628
629
630
631
632
633
634
635
636

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Recognizing...
result2:
{ 'alternative': [{'confidence': 0.88687539, 'transcript': 'play music'},
{'transcript': 'play musik'},
{'transcript': 'play muzik'},
{'transcript': 'play muzik'},
{'transcript': 'play muzic'}],

'final': True}
user said: play music
Enjoy the music..

Figure 6.8 Input to play music

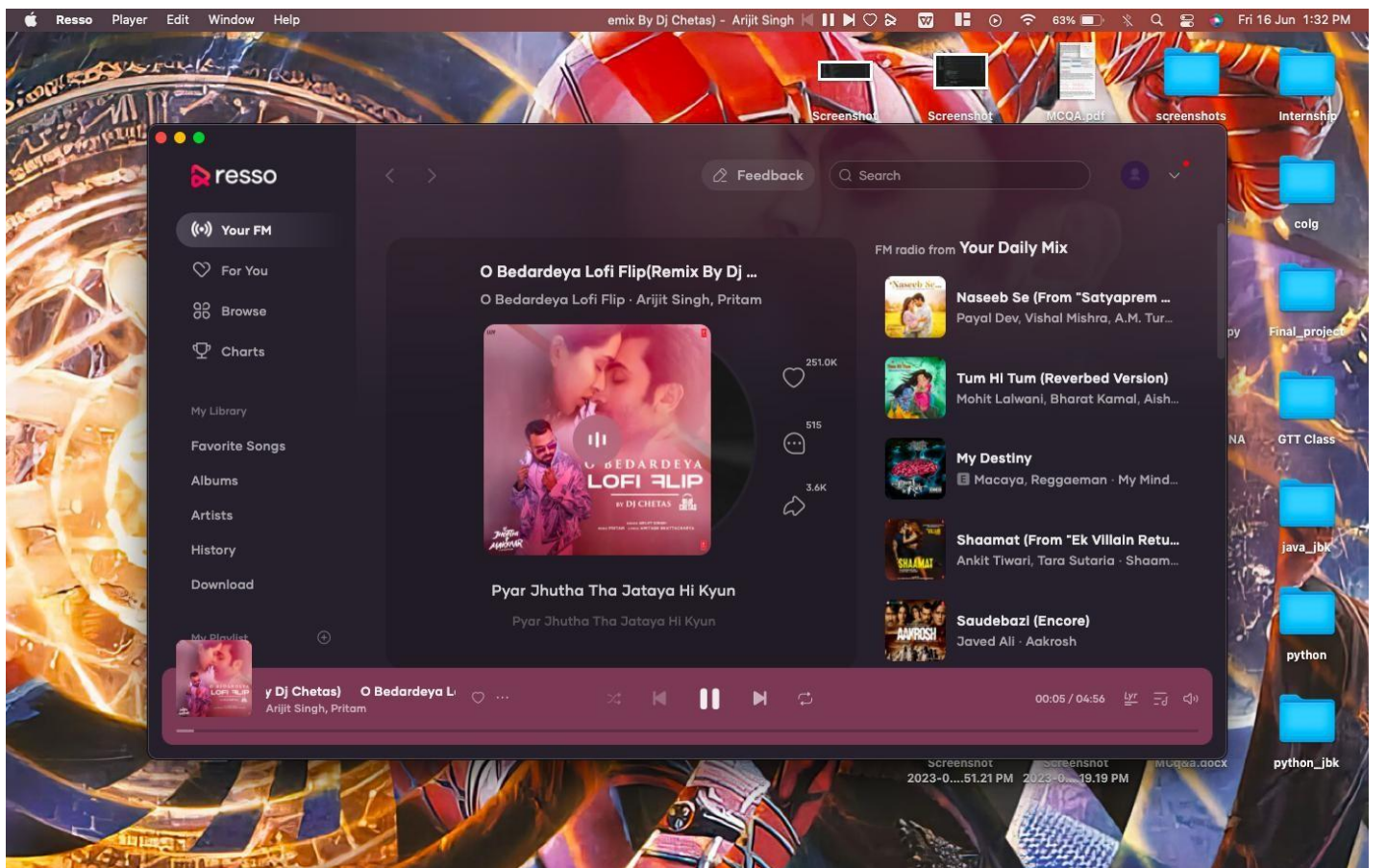


Figure 6.9 Output to play music

```

Listening...
Recognizing...
Say that again please...
Listening...
Recognizing...
User said: open command prompt

```

Figure 6.10 Input to open cmd

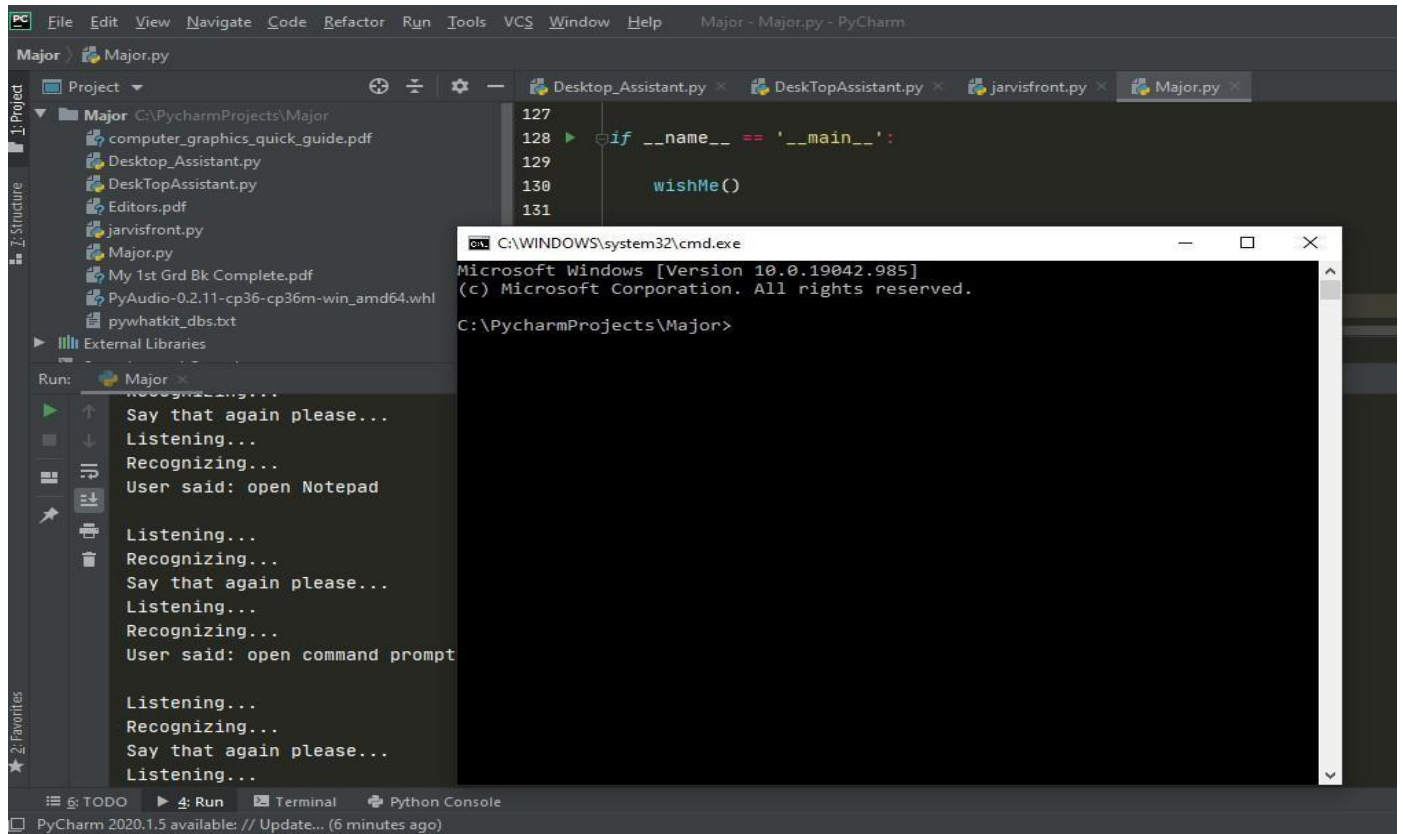


Figure 6.11 Output to open cmd


```

Run: Major x
C:\Python36\python.exe C:/PycharmProjects/Major/Major.py
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_EN-US_DAVID_11.0
Good Evening!
Hello!, I am Zira. Please tell me how may I help you
Listening...
Recognizing...
User said: search Python on Wikipedia

Searching Wikipedia...
Yamini, According to Wikipedia.....
Python is an interpreted high-level general-purpose programming language. Python's design philosophy

```

Figure 6.12 Input and output for Wikipedia search

```

Run: Major x
C:\Python36\python.exe C:/PycharmProjects/Major/Major.py
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_
Good Evening!
Hello!, I am Zira. Please tell me how may I help you
Listening...
Recognizing...
User said: open Microsoft Office

```

Figure 6.13 Input to open Microsoft Office

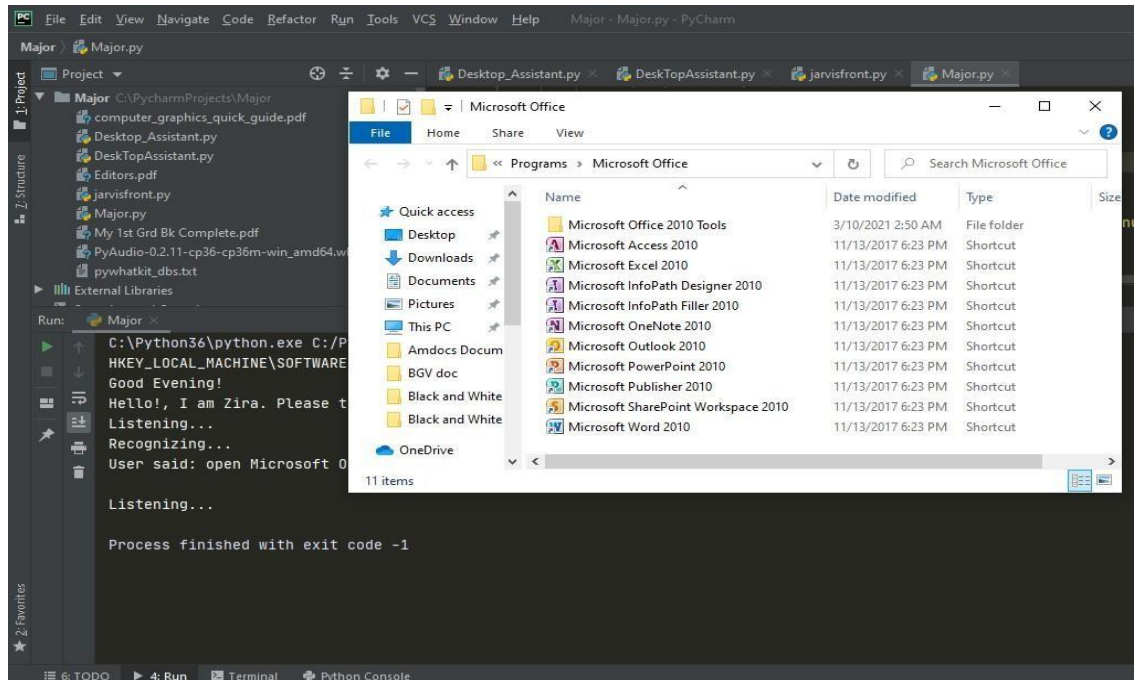


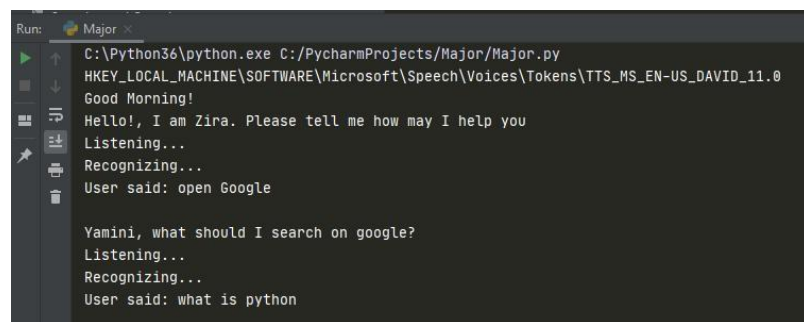
Figure 6.14 Output to open Microsoft Office

Chapter 7: System testing

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

7.1. 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 JAIVA can search on Google or not, as we can see in the figure 7.1, user said “Open Google”, then JAIVA asked, ”What should I search on Google?” then user said, “What is Python”, JAIVA open Google and searched for the required input.



```

Run: Major x
C:\Python36\python.exe C:/PycharmProjects/Major/Major.py
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_EN-US_DAVID_11.0
Good Morning!
Hello!, I am Zira. Please tell me how may I help you
Listening...
Recognizing...
User said: open Google

Yamini, what should I search on google?
Listening...
Recognizing...
User said: what is python
  
```

Figure 7.1 Input through voice commands

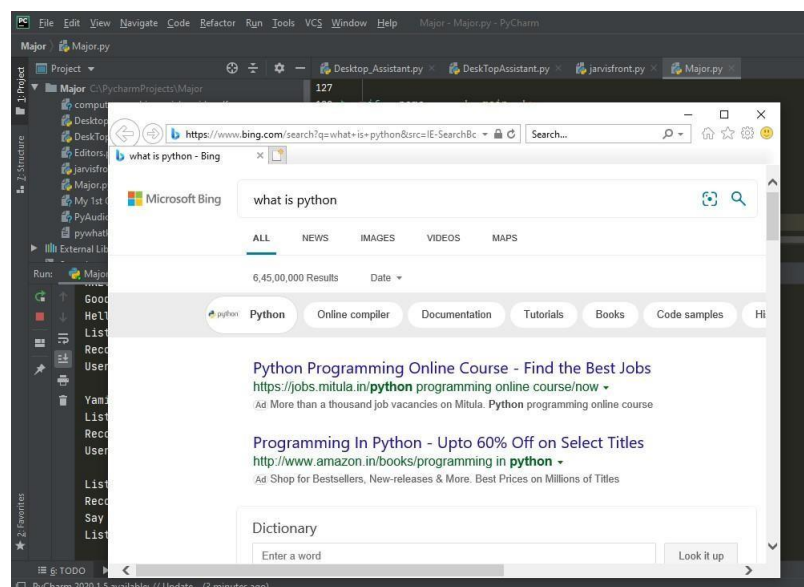


Figure 7.2 Output

7.2. 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 responds 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.

7.3. SECURITY

The security testing mainly focuses on vulnerabilities and risks. As JAIVA 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.

7.4. 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.

Chapter 8: Group Contribution

The project titled “**A.I. DESKTOP VOICE ASSISTANT: JAIVA**” was designed by me and my group.. From installing of all the packages, importing, creating all the necessary functions, designing GUI in PyQt and connecting that live GUI with the backend, was all done by me individually.

I, myself have done all the research before making this project, designed the requirement documents for the requirements and functionalities, wrote synopsis and all the documentation, code and made the project in such a way that it is deliverable at each stage. I have created the front end (.ui file) of the project using PyQt designer, the front end comprises of a live GUI and is connected with the .py file which contains all the classes and packages of the .ui file. The live GUI consists of moving GIFs which makes the front end attractive and user friendly.

I have written the complete code in Python language and in Visual Studio Code IDE from where it was very easy to install the packages and libraries, I have created the functions like `takeCommand()`, `wishMe()` and `taskExecution()` which has the following functionalities, like `takeCommand()` which is used to take the command as input through microphone of user and returns the output as string, `wishMe()` that greets the user according to the time like Good Morning, Good Afternoon and Good Evening and `taskExecution()` which contains all the necessary task execution definition like `sendEmail()`, `pdf_reader()`, `news()` and many conditions in if condition like “open Google”, “open notepad”, “search on Wikipedia”, “play music” and “open command prompt” etc.

While making this project I realized that with the advancement JAIVA 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. Functionalities of this project 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.

At last, I have updated my report and completed it by attaching all the necessary screen captures of inputs and outputs, mentioning the limitations and scope in future of this project.

Chapter 9: Conclusion

JAIWA 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:

9.1. LIMITATIONS

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

9.2. SCOPE FOR FUTURE WORK

- 9.2.1. Make JAIWA to learn more on its own and develop a new skill in it.
- 9.2.2. JAIWA android app can also be developed.
- 9.2.3. Make more JAIWA voice terminals.
- 9.2.4. Voice commands can be encrypted to maintain security.

M.B.E. Society's
M.S. Bidve Engineering College, Latur
(DEPARTMENT OF UNDER GRADUATION)
Approval Sheet for Chapters of Project Report

Title of Project: "Just An Artificial Intelligence Voice
Assistant"

Name of the Candidate: Mr. Pawar Akshay M.

Mr. Bhosale Atul R.

Mrs. Kamble Mayawati V.

Name of the Guide: Prof. D.V.Biradar

Chapter No.	Chapter Name	Guide's Signature with Date	Remarks
CHAPTER-1	INTRODUCTION		
CHAPTER-2	SYSTEM DESIGN		
CHAPTER-3	SOFTWARE DETAILS		
CHAPTER-4	IMPLEMENTATION WORK DETAILS		
CHAPTER-5	SOURCE CODE AND COMMANDS		
CHAPTER -6	INPUT/ OUTPUT SCREENSHOT		
<u>CHAPTER-7</u>	SYSTEM TESTING		
<u>CHAPTER-8</u>	INDIVIDUAL CONTRIBUTION		
<u>CHAPTER-9</u>	CONCLUSION		