International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 12 | December - 2023 SJIF Rating: 8.176 ISSN: 2582-3930

AI DESKTOP ASSISTANT

1.Dr.Preetha J Professor /Head Department Of Artificial Intelligence and Data Science , Muthayammal Engineering College, Rasipuram. hod.ads@mec.edu.in	2.UMA K Assistant Professor, Department Of Master of Computer Application, Muthayammal Engineering College, Rasipuram. umakkmoorthy@gmail.com.	3.DEGA. UDAY Student Department Of Artificial Intelligence and Data Science, Muthayammal Engineering College, Rasipuram. udaydega10@gmail.com.

4.LOKESH .M Student Department Of Artificial Intelligence and Data Science, Muthayammal Engineering College, Rasipuram. lokilokesh7032@gmail.com	5.SATEESH .B Student Department Of Artificial Intelligence and Data Science, Muthayammal Engineering College, Rasipuram. sateeshsatyam@gmail.com

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 **SpeechRecognition** 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 WhatsApp.
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 mail 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.

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 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, SpeechRecognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, pyQt etc. I have created a live GUI for interacting with the JARVIS 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. 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, SpeechRecognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, pyQt etc. I have created a live GUI for interacting with the JARVIS as it gives a design and interesting look while having the conversation.
With the advancement JARVIS 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.

2: System Design:

2.1. DATA FLOW

The data flow for JARVIS is as follow:

2.1 Data flow for JARVIS system is designed using the concept of Artificial Intelligence and with

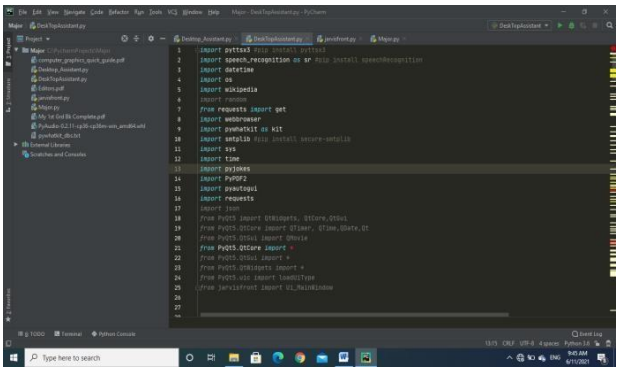
The help of necessary packages of Python. Python provides many libraries and packages to perform the tasks, for example pyPDF2 can be used to read PDF. The details of these packages are mentioned in Chapter 3 of this report.
The data in this project is nothing but user input, whatever the user says, 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.

3: Software Details

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, SpeechRecognition, Datetime, Wikipedia, Smtplib, pywhatkit, pyjokes, pyPDF2, pyautogui, pyQt etc. I have created a live GUI for interacting with the JARVIS as it gives a design and interesting look while having the conversation

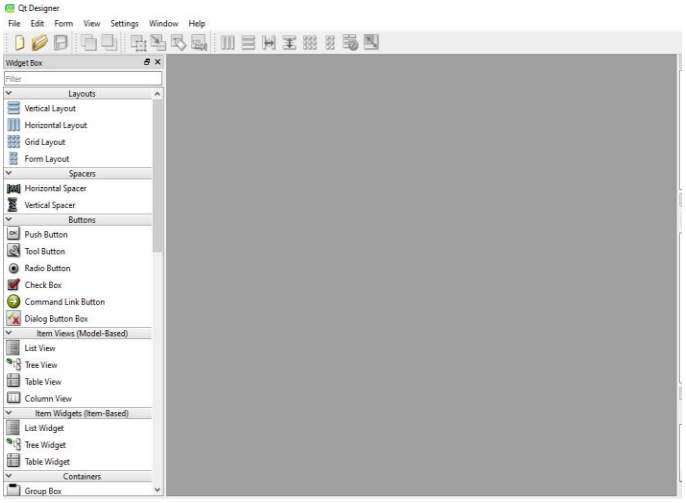
3.1. PYCHARM :

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.



3.2. PYQT5 FOR LIVE GUI :‘Q

PyQt5 is the most important python binding. It contains set of GUI widgets. PyQt5 has some important python modules like QTWidgets, QtCore, QtGui, and QtDesigner etc.



3.3. PYTHON LIBRARIES :

In JARVIS following python libraries were used:

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

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

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

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

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

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

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

3.3.8. Pyjokes: It is a python libararies which contains lots of interesting jokes in it.

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

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

3.3.11. os: It represents Operating System related functionality.

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

```
import pyttsx3 #pip install pyttsx3
import speech_recognition as sr #pip install speechRecognition
import datetime
import os
import wikipedia
import random
from requests import get
import webbrowser
import pywhatkit as kit
import smtplib #pip install secure-smtplib
import sys
import time
import pyjokes
import PyPDF2
import pyautogui
import requests
import json
from PyQt5 import QtWidgets, QtCore,QtGui
from PyQt5.QtCore import QTimer, QTime,QDate,Qt
from PyQt5.QtGui import QMovie
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5.QtWidgets import *
from PyQt5.uic import loadUiType
from jarvisfront import Ui_MainWindow
```

Figure 3.3 Imported Modules

4: Implementation Work Details

JARVIS, 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.

4.1. REAL LIFE APPLICATION :

4.1.1. Saves time: JARVIS 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.9

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 promp.

Chapter 5: Source Code and Commands

```
import pyttsx3
import speech_recognition as sr
import datetime
import os
import cv2
import random
from requests import get
import wikipedia
import webbrowser
import pywhatkit as kit
import smtplib
import sys
import time
import pyjokes
from bs4 import BeautifulSoup



engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
# print(voices[1].id)
engine.setProperty('voice', voices[len(voices) - 1].id)


# text to speech
def speak(audio):
    engine.say(audio)
    print(audio)
    engine.runAndWait()



# to convert voice into text
def takecommand():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("listening...")
        r.pause_threshold = 1
        audio = r.listen(source ,timeout=5,phrase_time_limit=10)

    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...")
        return "none"
    return query

# to wish
def wish():
    hour = int(datetime.datetime.now().hour)
    tt = time.strftime("%I:%M %p")

    if hour >= 0 and hour <= 12:
```

```
        speak(f"good morning, its {tt}")
    elif hour >= 12 and hour <= 18:
        speak(f"good afternoon, its {tt}")
    else:
        speak(f"good evening, its {tt}")
    speak("i am panda sir. please tell me how may i help you")

#to send email
def sendEmail(to,content):
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.ehlo()
    server.starttls()

server.login('lokilokesh9912@gmail.com', 'Lokesh@7032')
    server.sendmail('your email id', to, content)
    server.close()


if _name_ == "_main_":
    wish()
    while True:
    #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 adobe reader" in query:
            apath = "c:\\Program Files (x86)\\Adobe\\Reader 11.0\\Reader\\AcroRd32.exe"
            os.startfile(apath)

        elif "open command prompt" in query:
            os.system("start cmd")

        elif "open camera" in query:
            cap = cv2.VideoCapture(0)
            while True:
                ret, img = cap.read()
                cv2.imshow('cam', img)
                k = cv2.waitKey(50)
                if k==27:
                    break;
                cap.release()
                cv2.destroyAllWindows()

        elif "play music" in query:
            music_dir = "C:\\ music.m"
            songs = os.listdir(music_dir)
            # rd = random.choice(songs)
            for song in songs:
                if song.endswith('.mp3'):

os.startfile(os.path.join(music_dir, song))
```

```
elif "ip address" in query:
        ip = get('https://api.ipify.org').text
        speak(f"your IP address is {ip}")

    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 "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 google" in query:
        speak("sir, what should i search on google")
        cm = takecommand().lower()
        webbrowser.open(f"{cm}")

    elif "send whatsapp meessage" in query:
        kit.sendwhatmsg("+919121603721", "this is testing protocol",4,51)
        time.sleep(120)
        speak("message has been sent")

    elif " play songs on youtube" in query:
        kit.playonyt("see you again")

    elif "email to loki" in query:
        try:
            speak("what should i say?")
            content = takecommand().lower()
            to = "EMAIL TO OTHER PERSON"
            sendEmail(to,content)
            speak("Email has been sent to loki")

        except Exception as e:
            print(e)
            speak("sorry sir i am not able to sent this mail to loki")

    elif "you can sleep" in query:
        speak("thanks for using me sir, have a good day.")
        sys.exit()
```

```
#to close any application
    elif "close notepad" in query:
        speak("okay sir,closing notepad")
        os.system("taskkill /f /im notepad.exe")

#to set an alaem
    elif "set alarm" in query:
        nn = int(datetime.datetime.now().hour)
        if nn==22:
            music_dir = 'C:\\ music.m'
            songs = os.listdir(music_dir)

os.startfile(os.path.join(music_dir,songs[0]))
#to find a joke
    elif "tell me a joke" in query:
        joke = pyjokes.get_joke()
        speak(joke)

    elif "shut down the system" in query:
        os.system("shutdown /s /t 5")

    elif "restart the system" in query:
        os.system("shutdown /r /t 5")

    elif "sleep the system" in query:
        os.system("rund1132.exe powrprof.d11,Setsuspendedstate 0,1,0")

    # speak("sir, do you have any other work"

    elif "temparature" in query:
        search = "temperature in salem"
        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
        speak(f"current(search) is {temp}")

    #elif "activate how to do mod" in query:

        # from pywikihow import search_wikihow
        #speak("How to do mode is activated please tell me what you want to know")
        #how = take command()
        #max_results = 1
        #how_to = search_wikihow(how, max_results)
        #assert len(how_to) == 1
        #how_to[0].print()
        #speak(how to[0].summary)

    elif "activate how to do mod" in query:
```

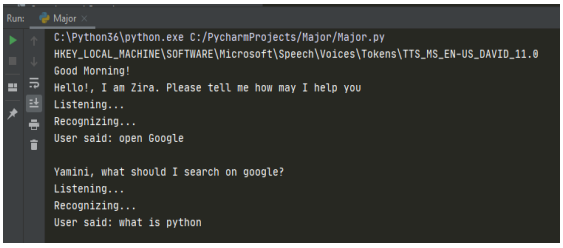
```
        speak("How to do mode is activated")
        while True:
            speak("please tell me what you want to know")
            how = takecommand()
            try:
                if "exit" in how or "close" in how:
                    speak("okay sir, how to do mode is closed")
                    break
                else:
                    max_results = 1
                    how_to = search_wikihow(how, max_results)
                    assert len(how_to) == 1
                    how_to[0].print()

speak(how_to[0].summary)
            except Exception as e:
                speak("sorry sir, i am not able to find this")


    def run(self):
        #self.TaskExecution()
        speak("please say wakeup to continue")
        while true:
            self.query = self.takecommand()
            if "wakeup" in self.query or "are you there" in self.query or "hello" in self.query:
                self.TaskExecution()
```

```
class Main(QMainWindow):
    def init(self):
        super().init()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)

self.ui.pushButton.clicked.connect(self.startTask)

self.ui.pushButton_2.clicked.connect(self.close)


    def del(self):
        sys.stdout = sys.stdout


    # def run(self):
    #     self.TaskExection
    def startTask(self):
        self.ui.movie = QtGui.QMovie("Jarvis/utils/images/live_wallpaper.gif")
        self.ui.label.setMovie(self.ui.movie)
        self.ui.movie.start()
        self.ui.movie = QtGui.QMovie("Jarvis/utils/images/initiating.gif")

self.ui.label_2.setMovie(self.ui.movie)
        self.ui.movie.start()
        timer = QTimer(self)
```

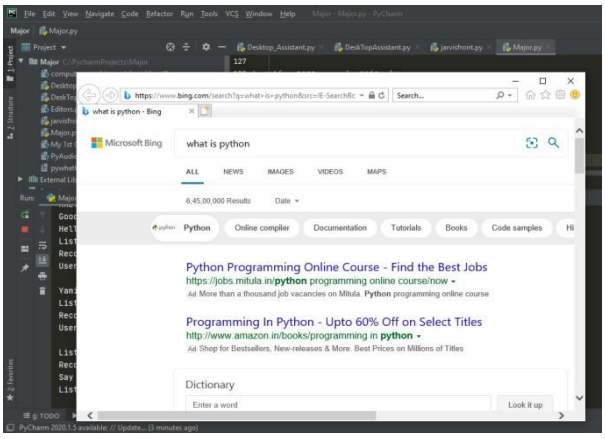
```
timer.timeout.connect(self.showTime)
        timer.start(1000)
        startExecution.start()


    def showTime(self):
        current_time = QTime.currentTime()
        current_date = QDate.currentDate()
        label_time = current_time.toString('hh:mm:ss')
        label_date = current_date.toString(Qt.ISODate)

self.ui.textBrowser.setText(label_date)

self.ui.textBrowser_2.setText(label_time)


import cv2

def open_camera():
    # Initialize the camera
    cap = cv2.VideoCapture(0)

    while True:
        # Capture frame-by-frame
        ret, frame = cap.read()

        # Display the resulting frame
        cv2.imshow('Camera Feed', frame)

        # Break the loop if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # Release the camera and close the window
    cap.release()
    cv2.destroyAllWindows()


if _name_ == "_main_":
    open_camera()
#-------------------- To take screenshot ----------
-------------

    elif"take a screenshot" in query or "take a screenshot" in query:
        speak("sir,please tell me the name for this screenshot file")
        name = takecommand().lower()
        speak("please sir hold the screen for few seconds, i am taking screenshot")
        time.sleep(3)
        img = pyautogui.screenshot()
        img.save(f"{name}.png")
        speak("i am done sir, the screenshot is saved in our main folder. now i am ready for another work")


    elif 'alarm' in query:
        speak("sir please tell me the time to set alarm.for example set alarm to 4:30 a.m.")
        tt = takecommand()  #set alarm to 4:30 a.m.
```

```
tt = tt.replace("set alarm to ", "") #4:30 a.m.

tt = tt.replace(".","") #4:30 am

tt = tt.upper() #4:30 AM

import MyAlarm

MyAlarm.alarm(tt)
```

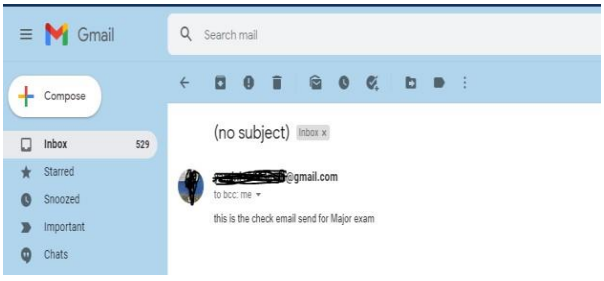
Chapter 6: Input/Output Screenshot



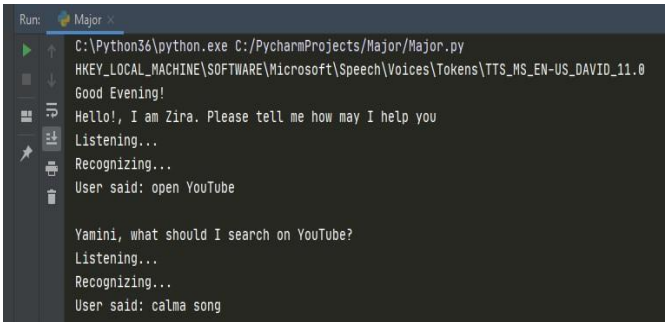
6.1 Input for Google search



6.2 Output for Google search



6.3 Output to send Email



6.4 Input for YouTube search

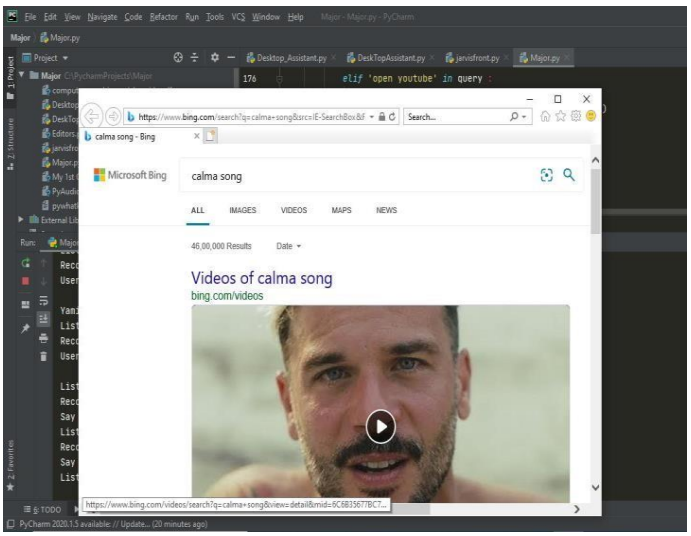
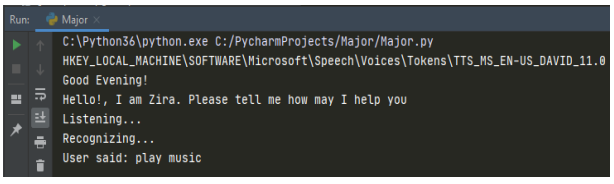
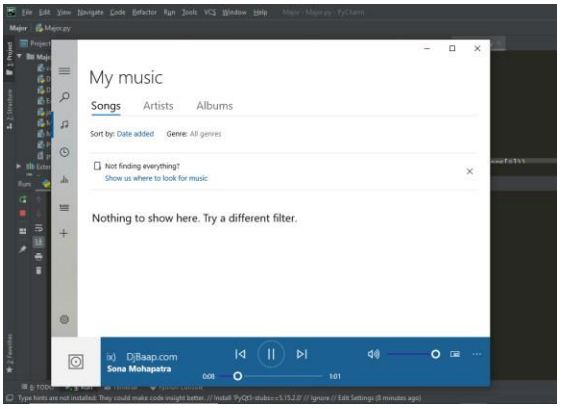


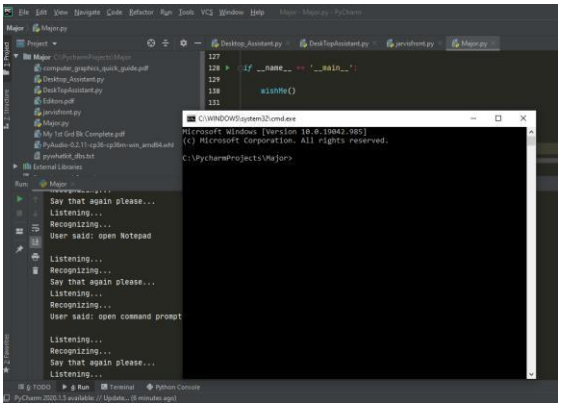
Figure 6.5 Input for YouTube search



6.6 Input to play music



6.7 Output to play music



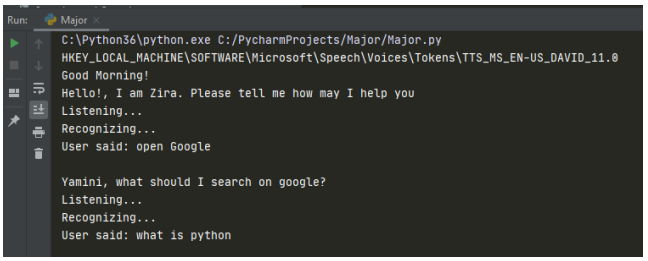
6.7 Output to open cmd.

Chapter 7: System testing:

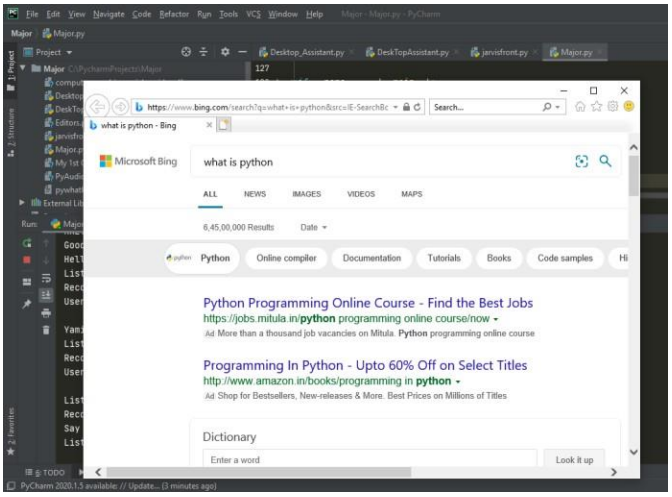
The system testing is done on fully integrated system to check whether the requirements are matching or not. The system testing for JARVIS desktop assistant focuses on the following four parameters:

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



7.1 Input through voice commands



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 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.

7.3. SECURITY :

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.

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.

Chapter8: Individual Contribution:

The project titled "**A.I. DESKTOP VOICE ASSISTANT: JARVIS**" was designed
by me individually. 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 PyCharm 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 JARVIS 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 :

JARVIS 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. JARVIS 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 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