

Major Project Report on **“AI BASED DESKTOP VIRTUAL ASSISTANT ”**

In partial fulfillment of requirements for the degree of
Bachelor of Technology (B. Tech.)
in
Computer Science and Engineering



Submitted by
Ms. Akshita Kanther (180161)

Under the Guidance of
Mr. Hitesh Jangir

Department of Computer Science and Engineering
SCHOOL OF ENGINEERING AND TECHNOLOGY
Mody University of Science and Technology
Lakshmangarh, Distt. Sikar-332311

April, 2022

A C K N O W L E D G E M E N T

I sincerely express my gratitude to my guide for his benevolent guidance in completing the report on AI based desktop virtual assistant. His kindness and help have been the source of encouragement for me.

It gives me a great sense of pleasure to present the report of the major project undertaken during B.Tech fourth year. I owe special debt of gratitude to Mr. Hitesh Jangir for his constant support and guidance throughout the course of my work, his sincerity, thoroughness and perseverance have been a constant of inspiration for me. It is only his cognizant efforts that my endeavors have seen light of the day.

I would relish expressing my gratitude and appreciation towards the Dean SET Dr. Prateek Bhanti , Mody University Lakshmangarh, for giving this opportunity to showcase my aptitude and build the substructure that i would require in the professional life.

Akshita Kanther (180161)

CERTIFICATE

This is to certify that the major project report entitled “AI Based Desktop Virtual Assistant ” submitted by Ms. Akshita Kanther, as a partial fulfillment for the requirement of B. Tech. VIII Semester examination of the School of Engineering and Technology, Mody University of Science and Technology, Lakshmangarh for the academic session 2021-2022 is an original project work carried out under the supervision and guidance of Mr. Hitesh Jangir has undergone the requisite duration as prescribed by the institution for the project work.

PROJECT GUIDE:

Signature:

Name: Mr. Hitesh Jangir

Date:

HEAD OF DEPARTMENT

Signature:

Name:

Date:

EXAMINER-I:

Signature:

Name:

Date:

EXAMINER-II

Signature:

Name:

Date:

ABSTRACT

The main task of a voice assistant is to minimize the use of input devices like keyboard, mouse, touch pens, etc. This will reduce both the hardware cost and space taken by it.

The voice assistant we have developed is a desktop-based built using python modules and libraries. This assistant is just a basic version that could perform all the basic tasks which have been mentioned. The functionalities or basic tasks 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 desktop reminders of your choice. It can have some basic conversation.

As we know Python is an emerging language so it becomes easy to write a script for Voice Assistant in Python. The instructions for the assistant can be handled as per the requirement of user. Speech recognition is the process of converting speech into text. This is commonly used in voice assistants like Alexa, Siri, etc. In Python there is an API called **Speech-Recognition** which allows us to convert speech into text. It was an interesting task to make my own assistant. It became easier to send emails without typing any word, Searching on Google without opening the browser, and performing many other daily tasks like playing music, opening your favorite IDE with the help of a single voice command. In the current scenario, advancement in technologies 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.

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.

LIST OF FIGURES

Figures	Page no.
Figure 2.1: Data flow for JARVIS	3
Figure 3.1: PyCharm IDE.....	4
Figure 3.2: Qt Designer.....	5
Figure 3.3: Imported Modules.....	6
Figure 6.1: Live GUI of JARVIS.....	20
Figure 6.2: Input for Google search.....	20
Figure 6.3: Output for Google search.....	21
Figure 6.4: Input to send Email.....	21
Figure 6.5: Output to send Email.....	21
Figure 6.6: Input for YouTube search.....	22
Figure 6.7: Output for YouTube search.....	22
Figure 6.8: Input to play music.....	22
Figure 6.9: Output to play music.....	23
Figure 6.10: Input to open cmd.....	23
Figure 6.11: Output to open cmd.....	23
Figure 6.12: Input and output for Wikipedia search.....	24
Figure 7.1: Input through voice commands.....	25
Figure 7.2: Output.....	25

Table of Contents

Sr.no.	Topics	Page no.
	Acknowledgement	i
	Certificate	ii
	Abstract	iii
	List of Figures	iv
1.	Introduction	1-2
1.1	<i>-Present System</i>	2
1.2	<i>-Proposed System</i>	2
2.	System Design	3
2.1	<i>-System flowchart</i>	3
3.	Hardware and Software details	4-6
3.1	<i>-Software Details</i>	4-6
3.2	<i>-Hardware Details</i>	6
4.	Implementation Work Details	7-8
4.1	<i>-Real life applications</i>	7
4.2	<i>-Data implementation and program execution</i>	7-8
5.	Source Code/Simulation Code/Commands etc.	9-19
6.	Input/output Screens/ Model's Photograph	20-24
7.	System Testing	25-26
8.	Individual Contribution	27
9.	Conclusion	28
9.1	<i>-Limitations</i>	28
9.2	<i>-Scope for future work</i>	28
10.	Bibliography	29
11.	Annexures	30
	<i>-Plagiarism Report</i>	30

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 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 listen to 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 it. 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.

The proposed system will have the following functionality:

1. The system can have both male and female voices according to user requirements.
2. Features supported in the current version include playing music, emails, texts, search on Wikipedia, or opening system installed applications, opening anything on the web browser, etc.
3. The system will keep listening for commands and the time for listening is variable which can be changed according to user requirements.
4. If the system is not able to gather information from the user input it will keep asking again to repeat till the desired no. of times.

Chapter 2: System Design

2.1. SYSTEM FLOWCHART

The data flow for JARVIS is as follow:

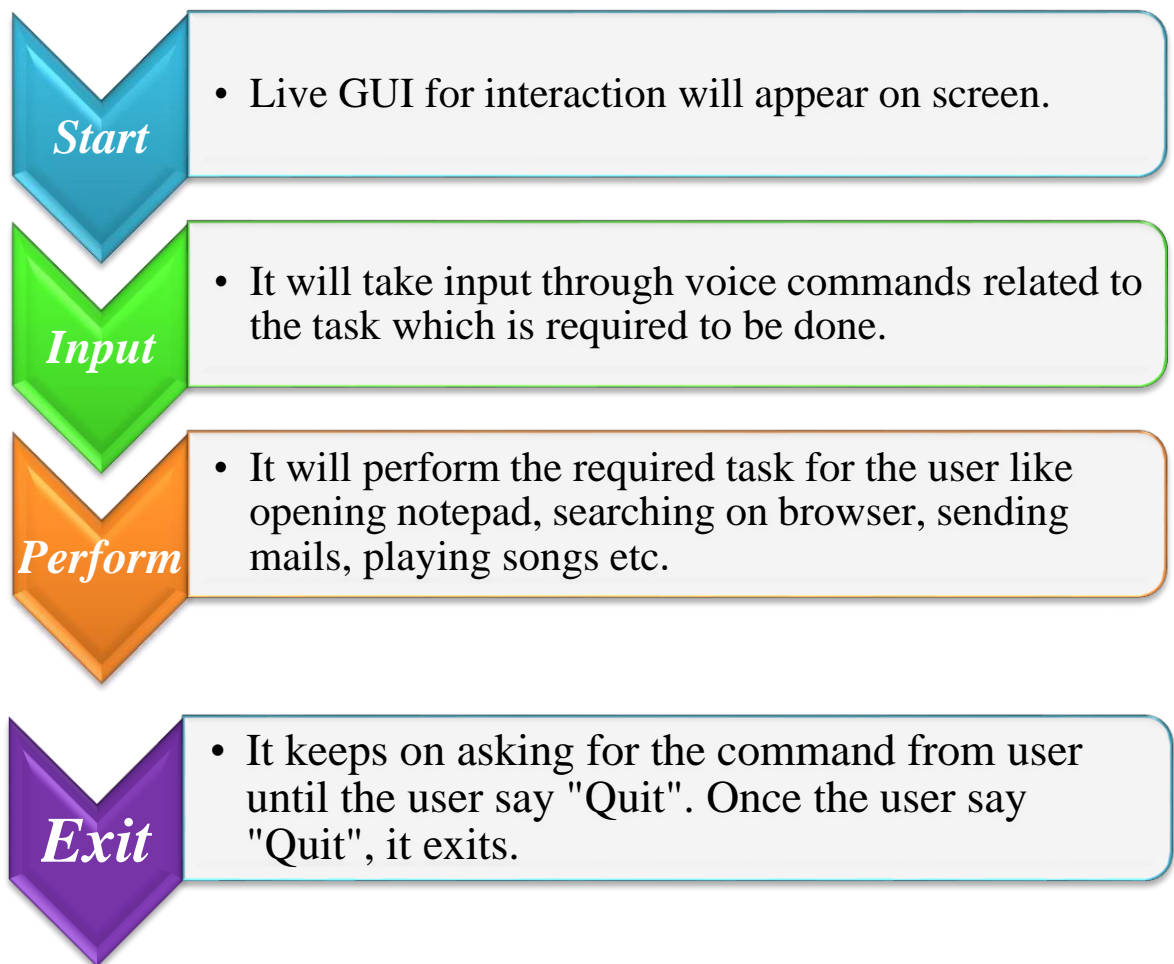


Figure 2.1 Data flow for JARVIS

The overall design of our system consists of the following phases:

1. Taking input from the user in the form of voice.
2. Converting the speech into text to be processed by the assistant.
3. The converted text is now processed to get the required results.
4. The text contains one or two keywords that determine what query is to be executed. If the keyword doesn't match any of the queries in the code then the assistant asks the user to speak again.
5. The result which is in the form of text is converted to speech again to give results to the user.

Chapter 3: Hardware and 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. SOFTWARE DETAILS

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.

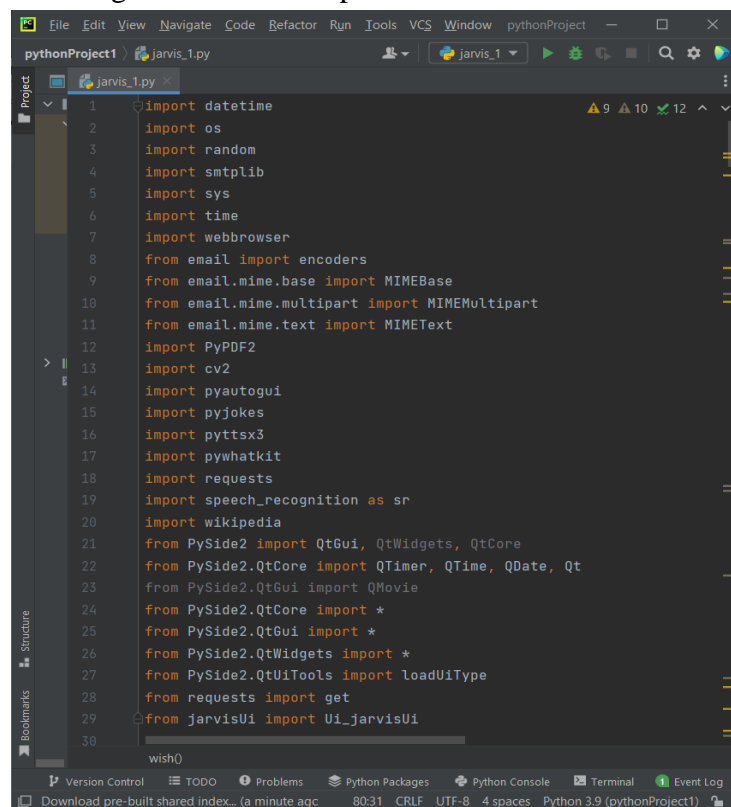


Figure 3.1 PyCharm IDE

2. PYSIDE2 FOR LIVE GUI

PySide2 is the most important python binding. It contains set of GUI widgets. PySide2 has some important python modules like QTWidgets, QtCore, QtGui, and QtDesigner etc.

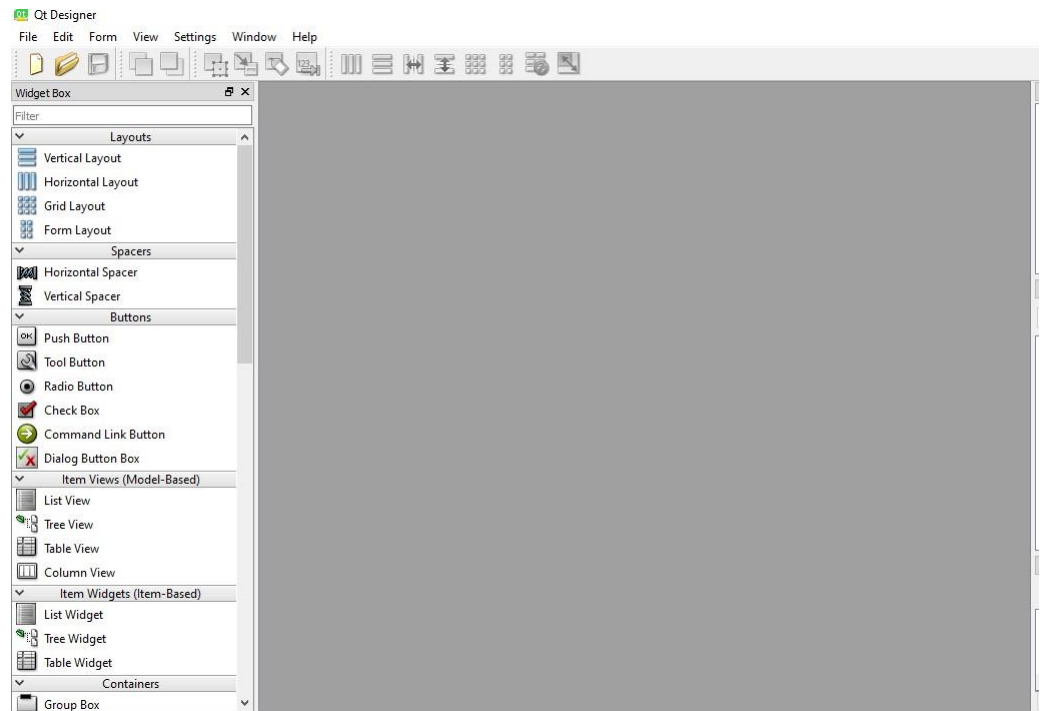


Figure 3.2 Qt Designer

3. PYTHON LIBRARIES

In JARVIS following python libraries were used:

- i. **pyttsx3**: It is a python library which converts text to speech.
- ii. **SpeechRecognition**: It is a python module which converts speech to text.
- iii. **pywhatkit**: It is python library to send WhatsApp message at a particular time with some additional features.
- iv. **Datetime**: This library provides us the actual date and time.
- v. **Wikipedia**: It is a python module for searching anything on Wikipedia.
- vi. **Smtplib**: Simple mail transfer protocol that allows us to send mails and to route mails between mail servers.
- vii. **pyPDF2**: It is a python module which can read, split, merge any PDF.
- viii. **Pyjokes**: It is a python library which contains lots of interesting jokes in it.
- ix. **Webbrowser**: It provides interface for displaying web-based documents to users.

- x. **Pyautogui:** It is a python libraries for graphical user interface.
- xi. **os:** It represents Operating System related functionality.
- xii. **sys:** It allows operating on the interpreter as it provides access to the variables and functions that usually interact strongly with the interpreter.

```
1  import datetime
2  import os
3  import random
4  import smtplib
5  import sys
6  import time
7  import webbrowser
8  from email import encoders
9  from email.mime.base import MIMEBase
10 from email.mime.multipart import MIMEMultipart
11 from email.mime.text import MIMEText
12 import PyPDF2
13 import cv2
14 import pyautogui
15 import pyjokes
16 import pyttsx3
17 import pywhatkit
18 import requests
19 import speech_recognition as sr
20 import wikipedia
21 from PySide2 import QtGui, QtWidgets, QtCore
22 from PySide2.QtCore import QTimer, QTime, QDate, Qt
23 from PySide2.QtGui import QMovie
24 from PySide2.QtCore import *
25 from PySide2.QtGui import *
26 from PySide2.QtWidgets import *
27 from PySide2.QtUiTools import loadUiType
28 from requests import get
29 from jarvisUi import Ui_jarvisUi
```

Figure 3.3 Imported Modules

3.2. HARDWARE DETAILS

1. Processor used: Intel(R) Core™ i5-7200U CPU@2.50GHz 2.71GHz
2. RAM :8.00GB

Chapter 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 it.

4.1. REAL LIFE APPLICATION

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.
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.
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. **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.
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:

1. LIBRARIES AND PACKAGES

- i. **pyttsx3:** It is a python library which converts text to speech.
- ii. **SpeechRecognition:** It is a python module which converts speech totext.

- iii. **pywhatkit:** It is python library to send WhatsApp message at a particular time with some additional features.
- iv. **Datetime:** This library provides us the actual date and time.
- v. **Wikipedia:** It is a python module for searching anything onWikipedia.
- vi. **Smtplib:** Simple mail transfer protocol that allows us to send mailsand to route mails between mail servers.
- vii. **pyPDF2:** It is a python module which can read, split, merge anyPDF.
- viii. **Pyjokes:** It is a python libraries which contains lots of interesting jokes in it.
- ix. **Webbrowser:** It provides interface for displaying web-based documents to users.
- x. **Pyautogui:** It is a python librariy for graphical user interface.
- xi. **os:** It represents Operating System related functionality.
- xii. **sys:** It allows operating on the interpreter as it provides access to the variables and functions that usually interact strongly with the interpreter.

2. FUNCTIONS

- i. **takeCommand():** The function is used to take the command as inputthrough microphone of user and returns the output as string.
- ii. **wishMe():** This function greets the user according to the time like Good Morning, Good Afternoon and Good Evening.
- iii. **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

Jarvis.py

```
import datetime
import os
import random
import smtplib
import sys
import time
import webbrowser
from email import encoders
from email.mime.base import MIMEBase
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
import PyPDF2
import cv2
import pyautogui
import pyjokes
import pyttsx3
import pywhatkit
import requests
import speech_recognition as sr
import wikipedia
from PySide2 import QtGui, QtWidgets, QtCore
from PySide2.QtCore import QTimer, QTime, QDate, Qt
from PySide2.QtGui import QMovie
from PySide2.QtCore import *
from PySide2.QtGui import *
from PySide2.QtWidgets import *
from PySide2.QtUiTools import loadUiType
from requests import get
from JarvisUi import Ui_JarvisUi

'''Defining engine for voices'''
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)

'''def news():
    main_url = http://newsapi.org/v2/top-headlines?sources=techcrunch&apiKey=YOUR\_API\_HERE

    main_page = requests.get(main_url).json()
```

```

# print(main_page)
articles = main_page["articles"]
# print(articles)
head = []
day = ["first", "second", "third", "fourth", "fifth", "sixth", "seventh", "eighth", "ninth",
"tenth"]
for ar in articles:
    head.append(ar["title"])
for i in range(len(day)):
    # print(f'today's {day[i]} news is: ', head[i])
    speak(f'today's {day[i]} news is: {head[i]}')

def pdf_reader():
    book = open('D:\\Email Validation in JavaScript.pdf', 'rb')
    pdfReader = PyPDF2.PdfFileReader(book)
    pages = pdfReader.numPages
    speak(f"Total no. of pages in this pdf {pages}")
    speak("please enter the page no. from which i have to read")
    pg = int(input("Please enter the page number: "))
    page = pdfReader.getPage(pg)
    text = page.extractText()
    speak(text)

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

def send_whatsapp_message(number, message):
    pywhatkit.sendwhatmsg_instantly(f"+91 {number}", message)

# to wish
def wish():
    hour = int(datetime.datetime.now().hour)
    if hour >= 0 and hour <= 12:
        speak("Good morning")
    elif hour > 12 and hour < 16:
        speak("Good afternoon")
    else:
        speak("Good evening")
    speak("Hello, I am Jarvis . Please tell me how can i help you")

def sendEmail(to, content):
    server = smtplib.SMTP('smtp.gmail.com', 587)

```



```

server.starttls()
server.ehlo()
server.login('youremail@gmail.com', 'Yourpassword')
server.sendmail('youremail@gmail.com', to, content)
server.close()

```

```

class MainThread(QThread):
    def __init__(self):
        super(MainThread, self).__init__()

    def run(self):
        self.start()

    """function for user input and convert voice into text"""
    def takecommand(self):
        r = sr.Recognizer()
        with sr.Microphone() as source:
            print("Listening...")
            r.pause_threshold = 1
            audio = r.listen(source, timeout=4, 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...")
            return "none"
        return query

    def start(self):
        wish()
        while True:
            # if 1:
            self.query = self.takecommand().lower()

            if "open notepad" in self.query:
                npath = "C:\\windows\\system32\\notepad.exe"
                os.startfile(npath)

            elif "close notepad" in self.query:
                speak("okay sir, closing notepad")
                os.system("taskkill /f /im notepad.exe")

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

```

```

elif "tell me a joke" in self.query:
    joke = pyjokes.get_joke()
    speak(joke)

elif "open adobe reader" in self.query:
    npath = "C:\\ProgramData\\Microsoft\\Windows\\Start Menu\\Programs\\Adobe
Reader XI"
    os.startfile(npath)

elif "open camera" in self.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 self.query:
    music_dir = "C:\\Users\\hp\\Music\\4K YouTube to MP3"
    songs = os.listdir(music_dir)
    rd = random.choice(songs)
    "for song in songs:
        if song.endswith('.mp3'):"
    os.startfile(os.path.join(music_dir, rd))

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

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

elif 'what is your name' in self.query:
    speak("My name is Jarvis.")

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

```

```

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

# elif "open stackoverflow" in self.query:
# webbrowser.open("https://stackoverflow.com")

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

elif "take screenshot" in self.query or "take a screenshot" in self.query:
    speak("Please tell me the name for this screenshot file")
    name = self.takecommand().lower()
    speak("please hold the screen for few seconds i am taking screenshot")
    time.sleep(3)
    img = pyautogui.screenshot()
    img.save(f"{name}.png")
    speak("Screenshot is taken successfully")

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

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

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

elif 'what is the time' in self.query:
    Time = datetime.datetime.now().strftime("%H:%M:%S")
    speak(f"The time is {Time}")

# elif 'send mail' in self.query or 'send email' in self.query or 'send gmail' in
self.query:
    # try:
    #     speak("What should I say?")
    #     content = self.takecommand()
    #     to = 'akshitakanther18.set@modyuniversity.ac.in'
    #     sendEmail(to, content)
    #     speak('Email has been sent!')
    # except Exception as e:
    #     print(e)
    #     speak("Sorry, I am not able to send this email")

```

```

elif 'send mail' in self.query or 'send email' in self.query or 'send gmail' in
self.query:
    speak("sir what should i say")
    self.query = self.takecommand().lower()
    if "send a file" in self.query:
        email = 'youremail@gmail.com' # Your email
        password = 'Yourpassword' # Your email account password
        send_to_email = 'akshitakanther18.set@modyuniversity.ac.in' #sending the
message to
        speak("okay, what is the subject for this email")
        self.query = self.takecommand().lower()
        subject = self.query # The Subject in the email
        speak("and , what is the message for this email")
        self.query2 = self.takecommand().lower()
        message = self.query2 # The message in the email
        speak("please enter the correct path of the file into the shell")
        file_location = input("please enter the path here") # File attachment in email
        speak("please wait,i am sending email now")
        msg = MIMEMultipart()
        msg['From'] = email
        msg['To'] = send_to_email
        msg['Subject'] = subject

        msg.attach(MIMEText(message, 'plain'))

        # Setup the attachment
        filename = os.path.basename(file_location)
        attachment = open(file_location, "rb")
        part = MIMEBase('application', 'octet-stream')
        part.set_payload(attachment.read())
        encoders.encode_base64(part)
        part.add_header('Content-Disposition', "attachment; filename= %s" %
filename)

        # Attach the attachment to the MIMEMultipart object
        msg.attach(part)

        server = smtplib.SMTP('smtp.gmail.com', 587)
        server.starttls()
        server.login(email, password)
        text = msg.as_string()
        server.sendmail(email, send_to_email, text)
        server.quit()
        speak("email has been sent successfully")

```

```

else:
    email = 'youremail@gmail.com' # Your email
    password = 'Yourpassword' # Your email account password
    send_to_email = 'akshitakanther18.set@modyuniversity.ac.in' # Whom you
are sending the message to
    message = self.query # The message in the email

    server = smtplib.SMTP('smtp.gmail.com', 587) # Connect to the server
    server.starttls() # Use TLS
    server.login(email, password) # Login to the email server
    server.sendmail(email, send_to_email, message) # Send the email
    server.quit() # Logout of the email server
    speak("email has been sent successfully")

elif 'play song on youtube' in self.query or 'song youtube' in self.query:
    speak("What should i play on youtube?")
    answer = self.takecommand()
    pywhatkit.playonyt(answer)
    speak(f'Playing {answer}')

elif 'hello' in self.query or 'hi' in self.query:
    speak('Hello! how can i help you?')
    answer = self.takecommand()

elif 'what can you do for me' in self.query:
    speak('I can play music, send Email, take screenshot etc.')

elif "send whatsapp message" in self.query:
    speak('On what number should I send the message sir? Please enter in the
console: ')
    number = input("Enter the number: ")
    speak("What is the message sir?")
    message = self.takecommand().lower()
    send_whatapp_message(number, message)
    speak("I've sent the message sir.")

elif "exit" in self.query:
    speak("thanks for using me , have a good day.")
    sys.exit()

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

elif "restart the system" in self.query:

```

```

os.system("shutdown /r /t 5")

elif "sleep the system" in self.query:
    os.system("rundll32.exe powrprof.dll,SetSuspendState 0,1,0")

elif 'switch the window' in self.query:
    pyautogui.keyDown("alt")
    pyautogui.press("tab")
    time.sleep(1)
    pyautogui.keyUp("alt")

elif 'close the window' in self.query:
    speak("closing the window")
    pyautogui.hotkey('alt', 'f4')

elif 'minimise the windows ' in self.query or 'minimise the window' in self.query:
    speak("minimizing window")
    pyautogui.hotkey('Win', 'd')

elif 'maximize the windows' in self.query or 'maximize the window' in self.query:
    speak("maximizing windows")
    pyautogui.hotkey('Win', 'd')

elif 'read pdf' in self.query:
    pdf_reader()

elif 'timer' in self.query or 'stopwatch' in self.query:
    speak("For how many minutes?")
    timing = self.takecommand()
    timing = timing.replace('minutes', '')
    timing = timing.replace('minute', '')
    timing = timing.replace('for', '')
    timing = float(timing)
    timing = timing * 60
    speak(f'I will remind you in {timing} seconds')
    time.sleep(timing)
    speak('Your time has been finished')

elif "tell me news" in self.query:
    speak("please wait sir, fetching the latest news")
    news()

startExecution = MainThread()
FROM_MAIN_ = loadUiType(os.path.join(os.path.dirname(__file__), "./jarvisUi.ui"))

```

```

class Main(QMainWindow):
    def __init__(self):
        super().__init__()
        self.ui = Ui_jarvisUi()
        self.ui.setupUi(self)
        self.ui.pushButton.clicked.connect(self.startTask)
        self.ui.pushButton_2.clicked.connect(self.close)

    def startTask(self):
        self.ui.movie = QtGui.QMovie("../Downloads/man1.gif")
        self.ui.label.setMovie(self.ui.movie)
        self.ui.movie.start()
        self.ui.movie = QtGui.QMovie("../Downloads/man2.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)

app = QApplication(sys.argv)
jarvis = Main()
jarvis.show()
exit(app.exec_())

```

JarvisUi.py

```

from PySide2 import QtCore, QtGui, QtWidgets

```

```

class Ui_jarvisUi(object):
    def setupUi(self, jarvisUi):
        jarvisUi.setObjectName("jarvisUi")
        jarvisUi.resize(1121, 695)
        self.centralwidget = QtWidgets.QWidget(jarvisUi)
        self.centralwidget.setObjectName("centralwidget")
        self.label = QtWidgets.QLabel(self.centralwidget)

```

```

self.label.setGeometry(QRect(0, -5, 1121, 701))
self.label.setText("")
self.label.setPixmap(QtGui.QPixmap("../Downloads/man1.gif"))
self.label.setScaledContents(True)
self.label.setObjectName("label")
self.pushButton = QtWidgets.QPushButton(self.centralwidget)
self.pushButton.setGeometry(QRect(810, 600, 101, 28))
font = QtGui.QFont()
font.setPointSize(12)
font.setBold(True)
font.setWeight(75)
self.pushButton.setFont(font)
self.pushButton.setStyleSheet("background-color: rgb(255, 255, 0);")
self.pushButton.setObjectName("pushButton")
self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
self.pushButton_2.setGeometry(QRect(910, 600, 101, 28))
font = QtGui.QFont()
font.setPointSize(12)
font.setBold(True)
font.setWeight(75)
self.pushButton_2.setFont(font)
self.pushButton_2.setStyleSheet("background-color: rgb(255, 0, 0);")
self.pushButton_2.setObjectName("pushButton_2")
self.label_2 = QtWidgets.QLabel(self.centralwidget)
self.label_2.setGeometry(QRect(10, 10, 371, 101))
self.label_2.setText("")
self.label_2.setPixmap(QtGui.QPixmap("../Downloads/man2.gif"))
self.label_2.setObjectName("label_2")
self.textBrowser = QtWidgets.QTextBrowser(self.centralwidget)
self.textBrowser.setGeometry(QRect(620, 20, 201, 51))
font = QtGui.QFont()
font.setPointSize(16)
font.setBold(True)
font.setWeight(75)
self.textBrowser.setFont(font)
self.textBrowser.setStyleSheet("background:transparent;\n"
"border-radius:none;\n"
"color:white;\n")
self.textBrowser.setObjectName("textBrowser")
self.textBrowser_2 = QtWidgets.QTextBrowser(self.centralwidget)
self.textBrowser_2.setGeometry(QRect(820, 20, 201, 51))
font = QtGui.QFont()
font.setPointSize(16)
font.setBold(True)
font.setWeight(75)

```



```

        self.textBrowser_2.setFont(font)
        self.textBrowser_2.setStyleSheet("background:transparent;\n"
"border-radius:none;\n"
"color:white;\n")
        self.textBrowser_2.setObjectName("textBrowser_2")
        jarvisUi.setCentralWidget(self.centralwidget)
        self.statusbar = QtWidgets.QStatusBar(jarvisUi)
        self.statusbar.setObjectName("statusbar")
        jarvisUi.setStatusBar(self.statusbar)

        self.retranslateUi(jarvisUi)
        QtCore.QMetaObject.connectSlotsByName(jarvisUi)

def retranslateUi(self, jarvisUi):
    _translate = QtCore.QCoreApplication.translate
    jarvisUi.setWindowTitle(_translate("jarvisUi", "MainWindow"))
    self.pushButton.setText(_translate("jarvisUi", "Run"))
    self.pushButton_2.setText(_translate("jarvisUi", "Exit"))

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    jarvisUi = QtWidgets.QMainWindow()
    ui = Ui_jarvisUi()
    ui.setupUi(jarvisUi)
    jarvisUi.show()
    sys.exit(app.exec_())

```

Chapter 6: Input/Output Screenshot

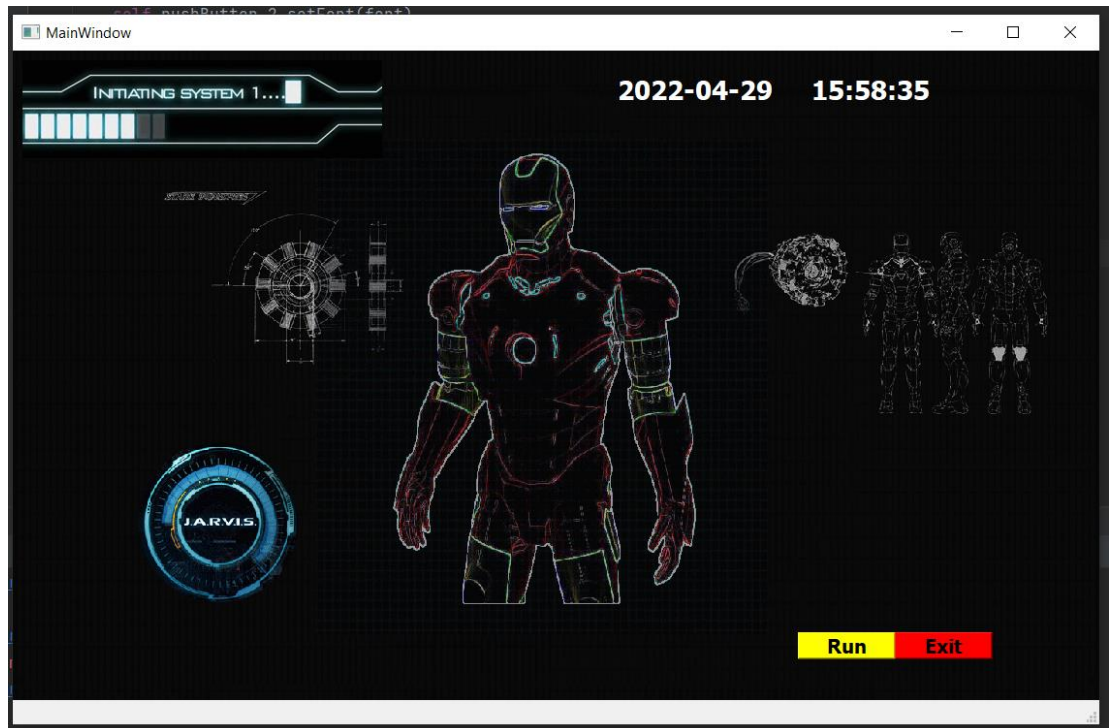


Figure 6.1 Live GUI of JARVIS

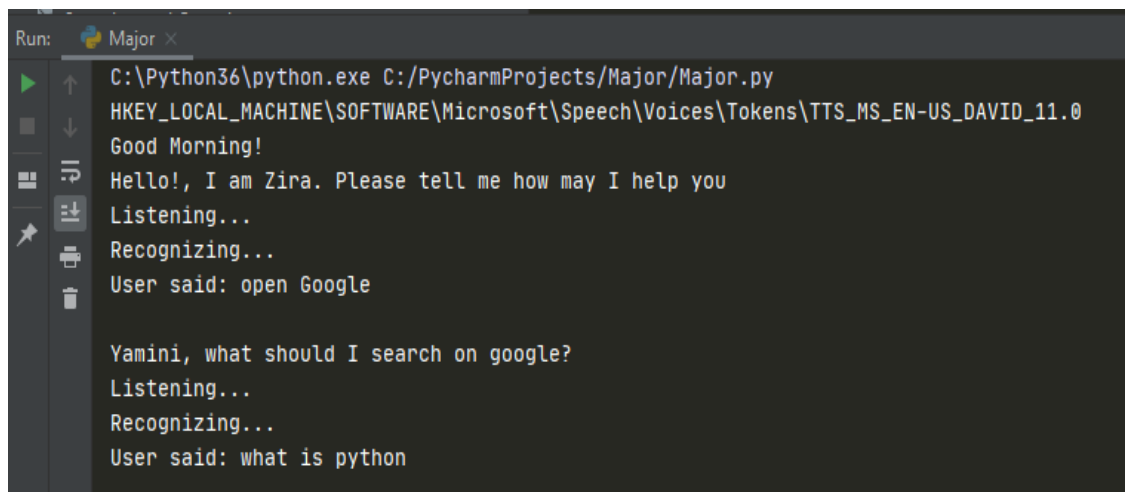


Figure 6.2 Input for Google search

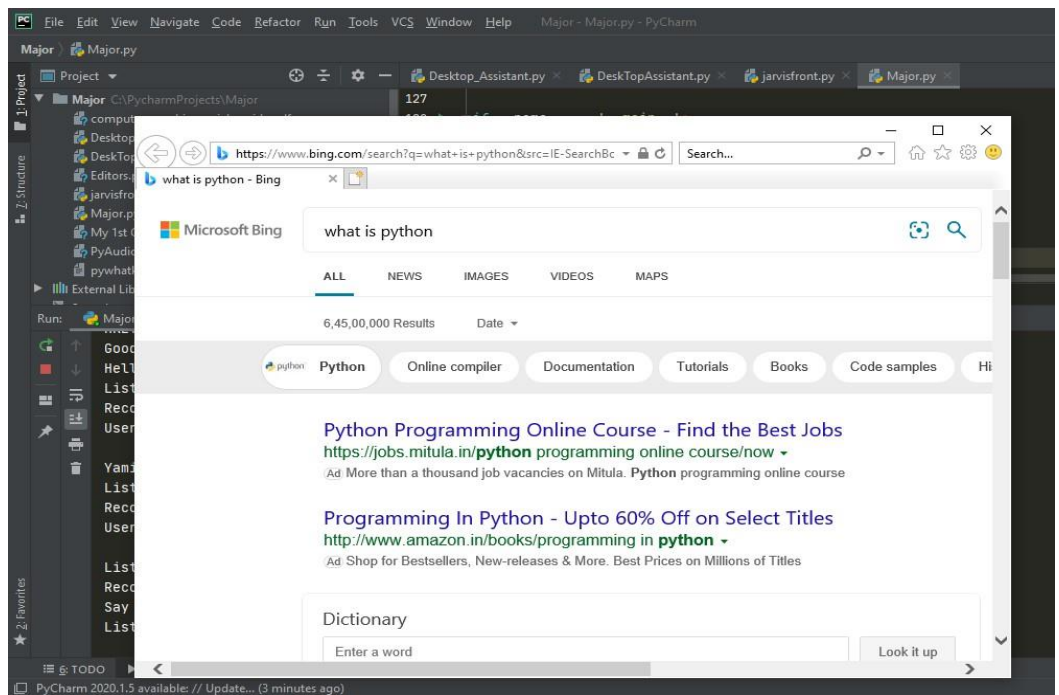


Figure 6.3 Output for Google search

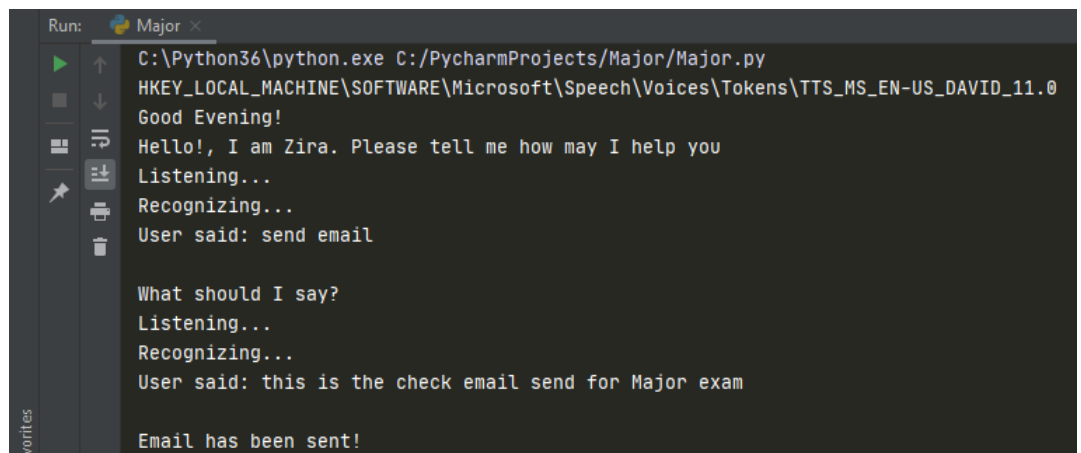


Figure 6.4 Input to send Email

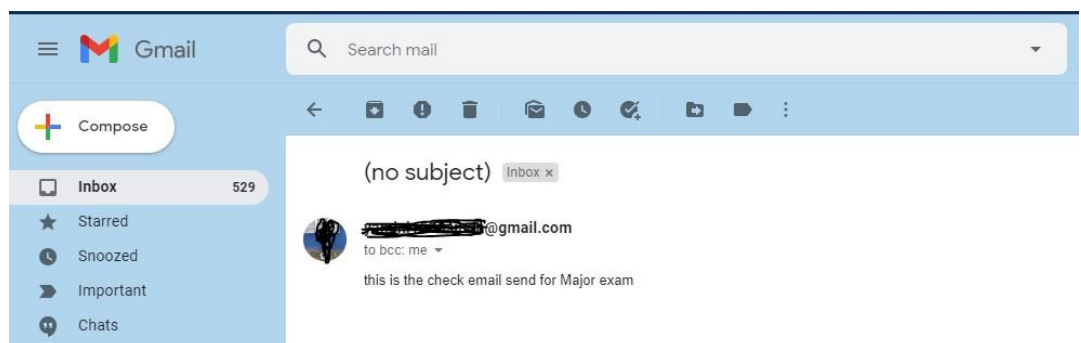


Figure 6.5 Output to send Email

```
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: open YouTube

Yamini, what should I search on YouTube?
Listening...
Recognizing...
User said: calma song
```

Figure 6.6 Input for YouTube search

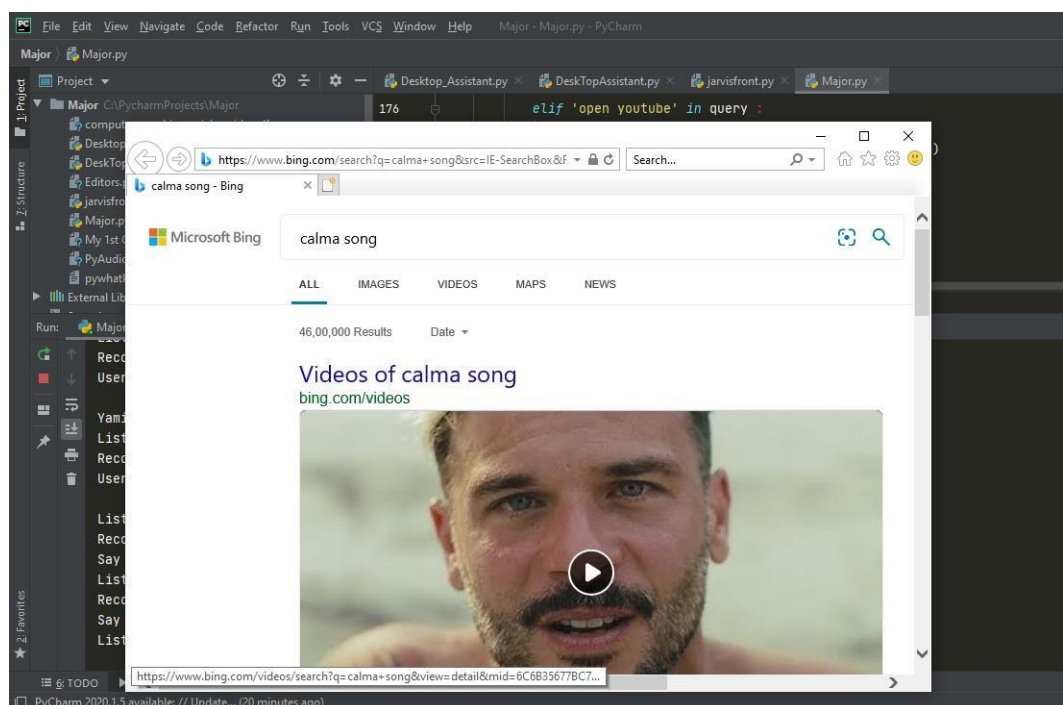


Figure 6.7 Output for YouTube search

```
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: play music
```

Figure 6.8 Input to play music

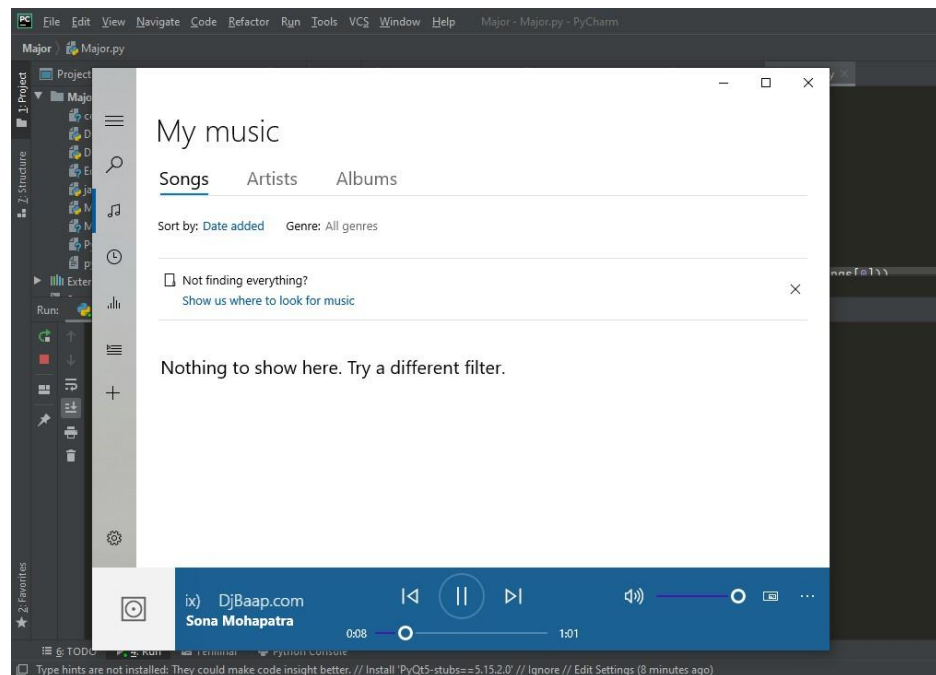


Figure 6.9 Output to play music

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

Figure 6.10 Input to open cmd

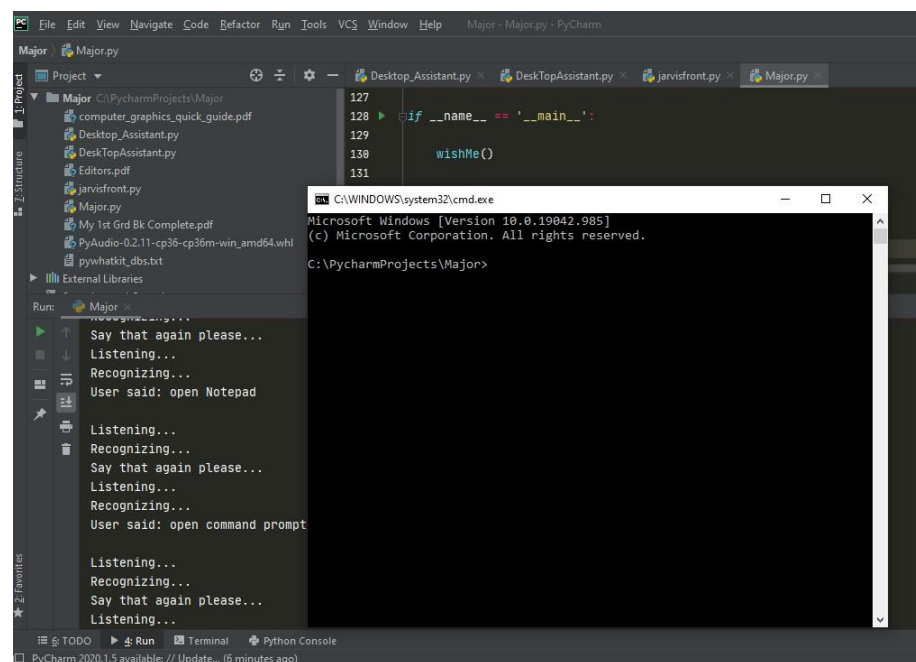
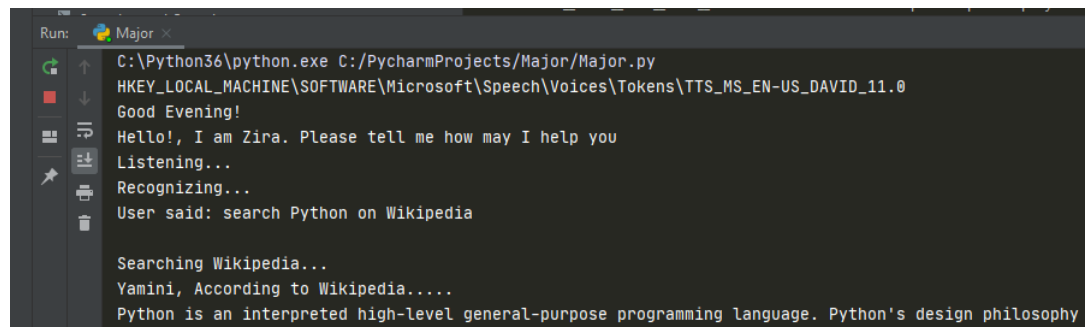


Figure 6.11 Output to open cmd



```
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
```

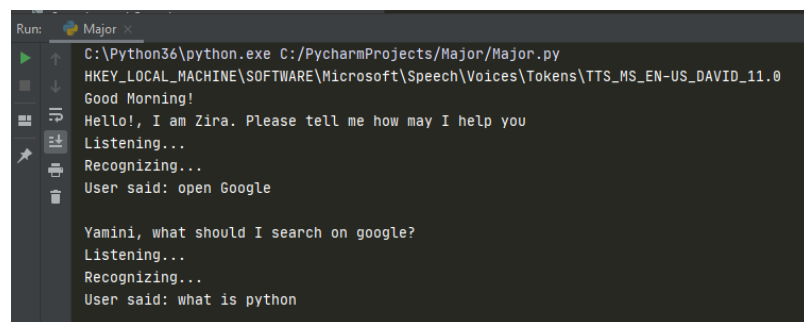
Figure 6.12 Input and output for Wikipedia search

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:

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.



```
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

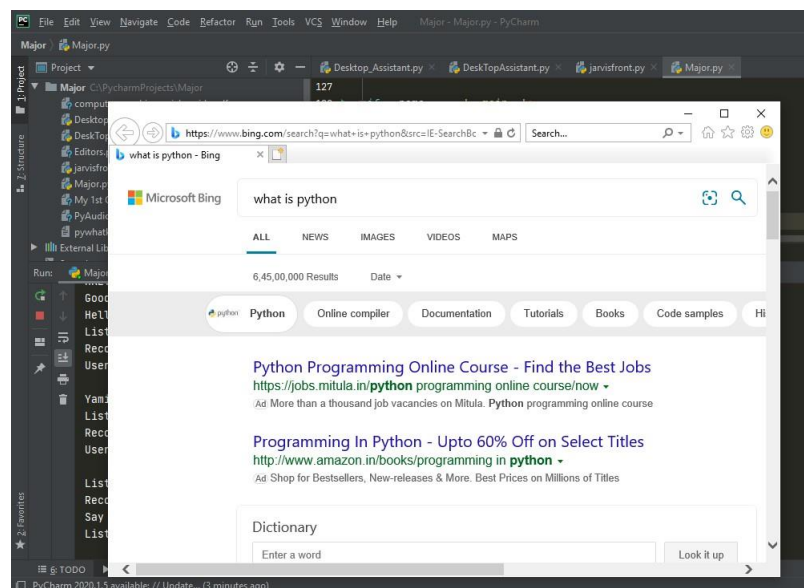


Figure 7.2 Output

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.

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.

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: Individual Contribution

The project titled “**A.I. BASED DESKTOP VIRTUAL ASSISTANT : JARVIS**” was designed by me individually. From installing of all the packages, importing, creating all the necessary functions, designing GUI in PySide2 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 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

1. Security is somewhere an issue, there is no voice command encryption in this project.
2. Background voice can interfere
3. Misinterpretation because of accents and may cause inaccurate results.
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

The virtual assistants which are currently available are fast and responsive but we still have to go a long way. The understanding and reliability of the current systems need to be improved a lot. The assistants available nowadays are still not reliable in critical scenarios. The future of these assistants will have the virtual assistants incorporated with Artificial Intelligence which includes Machine Learning, Neural Networks, etc. and IoT. With the incorporation of these technologies, we will be able to achieve new heights. What the virtual assistants can achieve is much beyond what we have achieved till now. Most of us have seen Jarvis, that is a virtual assistant developed by iron man which is although fictional but this has set new standards of what we can achieve using voice-activated virtual assistants.

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

Bibliography

- [1] D O'SHAUGHNESSY, SENIOR MEMBER, IEEE, "Interacting With Computers by Voice: Automatic Speech Recognition and Synthesis" proceedings of THE IEEE, VOL. 91, NO. 9, SEPTEMBER 2003
- [2] Kei Hashimoto, Junichi Yamagishi, William Byrne, Simon King, Keiichi Tokuda, "An analysis of machine translation and speech synthesis in speech-to-speech translation system" proceedings of 5108978-1-4577-0539- 7/11/\$26.00 ©2011 IEEE.
- [3] Nil Goksel-Canbek Mehmet Emin Mutlu, "On the track of Artificial Intelligence: Learning with Intelligent Personal Assistant" International Journal of Human Sciences.
- [4] H. Phatnani, Mr. J. Patra and Ankit Sharma' "CHATBOT ASSISTING: SIRI" Proceedings of BITCON-2015 Innovations For National Development National Conference on Research and Development in Computer Science and Applications, E-ISSN2249–8974.
- [5] Sutar Shekhar, P. Sameer, Kamad Neha, Prof. Devkate Laxman, " An Intelligent Voice Assistant Using Android Platform", March 2015, IJARCSMS, ISSN: 232 7782
- [6] VINAY SAGAR, KUSUMA SM, "Home Automation Using Internet of Things", June-2015, IRJET, e-ISSN: 2395 -0056.
- [7] "Speech recognition with flat direct models," IEEE Journal of Selected Topics in Signal Processing, 2010.
- [8] Rishabh Shah, Siddhant Lahoti, Prof. Lavanya. K, "An Intelligent Chatbot using Natural Language Processing". International Journal of Engineering Research, Vol. 6, pp. 281-286, 1 May 2017.




Annexures



Document Information

Analyzed document	AI Based virtual assistant major project report.pdf (D134859820)
Submitted	2022-04-28T08:08:00.0000000
Submitted by	hitesh Jangir
Submitter email	hiteshjangir.cet@modyuniversity.ac.in
Similarity	13%
Analysis address	hiteshjangir.cet.modyun@analysis.urkund.com

Sources included in the report

W	URL: https://ijariie.com/AdminUploadPdf/Voice_Activated_Intelligent_Assistant_Using_ML_ijariie15025.pdf Fetched: 2022-01-15T16:52:06.9930000	 5
W	URL: https://www.fiverr.com/anmoljain16/make-an-customizable-jarvis-for-you-that-can-do-very-very-cool-things Fetched: 2022-04-28T08:09:36.9230000	 1
W	URL: https://www.scribd.com/document/568173534/Final-Completed-Virtual-Assistant-Alexa-an-Artificial-Intelligence Fetched: 2022-04-28T08:09:23.6530000	 1