JARVIS - Virtual Computer Assistant

Project Report

Submitted in partial fulfilment of the requirement for the award of the Degree of Bachelor in

INFORMATION TECHNOLOGY

Session-2019-20

Submitted by:

Rishu Kumar (1611025)
Sonali Priya (1611035)
Shyam Nandan Kumar (1611034))
Sonu Kumar (1611037)
Rajan Kumar (1711007D)

Project Guide:

Dr. Pooja Sharma
(Assistant prof. dept. IT)

Department of Higher, Technical Education & Skill Development
Govt. of Jharkhand.

P.O. Sindri Institute, Dhanbad -828123 (Jharkhand)

Department of Information Technology

BIT SINDRI

Department of Higher, Technical Education & Skill Development
Govt. of Jharkhand.

P.O. Sindri Institute, Dhanbad-828123 (Jharkhand)

CERTIFICATE

This is to certify that the Project Report entitled "JARVIS – Virtual Desktop Assistant " submitted by Sonali Priya (1611035), Shyam Nandan Kumar (1611034), Sonu Kumar (1611037), Rishu Kumar(1611025), Rajan Kumar(1711007D) for the partial fulfilment of the requirement for the award of

"BACHELOROFTECHNOLOGY"
inINFORMATIONTECHNOLOGY**(Session 2019-20) has been carried out under mysupervision and guidance. It is certified that the report embodies the results of the work carried out by them within the prescribed period.**

APPROVED FOR SUBMISSION

Project Guide

Prof. (Dr.) Pooja Sharma

Assistant Prof.

Department of IT

BIT Sindri, Dhanbad

Head of the Department

Prof. F. Ansari

Department of IT BIT Sindri, Dhanbad

ACKNOWLEDGEMENT

We express our sincere gratitude to Prof. (Dr.) Pooja Sharma for her valuable guidance and timely suggestions during the entire duration of my project work, without which this work would not have been possible. We would also like to convey our deep respects to HOD and all other faculty members of *Information Technology Department* who have bestowed their great effort and help at appropriate times without which it would have been very difficult on my part to finish this work. Finally, we would also like to thank our friends for their advice and pointing out our mistakes.

Sonali Priya(1611035)
Shyam Nandan Kumar(1611034)
Sonu Kumar(1611037)
Rishu Kumar(1611025)
Rajan Kumar(1711007D)

ABSTRACT

The project is inspired by *Google's version* of Amazon's Alexa, Apple's Siri and Microsoft's Cortana forms of different *Google Assistant*. It has made incredible progress since its 2016 launch and is probably the most advanced and dynamic of the assistants out there. The project aims to present a virtual computer assistant to assist the user with easy and interactive application. This assistant faciliate us with voice commanding feature which provide a sequence of voice interaction between the user and machine. This application will control our device and access the information present in our device at one call. It will offer voice commands, voice searching and voice-activated device control, letting the device complete a number of tasks.

The application is built in *Jupyter Notebook* platform with Python as a programming language. Along with this, the application demands to import different softwares and libraries like pyttsr3, webbrowser, smtplib, speech_recognition, datetime etc. The application also requires a proper set of code in python for executing the commands and getting desired output.

It works by receiving the voice of the user as an instruction and gives the response. The application in lists features like sending e-mails, playing music, listing music, greeting user, opening various sites like we tube, google and solving user queries. With the availability of this software one can control different actions in the computer with the help of voice, providing the user totally new dimension for performing his task.

Thus, this application increases the ease and comfort of PC users by trying to present a total interactive interface between the user and machine, which in long run will make human understanding and interest towards machine more famaliar and easy to work.

Table of Contents

1. Building JARVIS

Idea is simple. An AI we can

- talk by our phone and computer,

- that can control home including lights.
- temperature, appliances, music and security, -that learns our tastes and patterns,
- that can learn new words and concepts, and
- that can even entertain us.

2. INSTALLATION REQUIREMENT

Working With Microphones

- Installing PyAudio
- The Microphone Class
- Using listen() to Capture Microphone Input
- Handling Unrecognizable Speech

2.1 Picking a Python Speech Recognition Package

A handful of packages for speech recognition exist on PyPI. A few of them include:

- apiai
- assemblyai
- google-cloud-speech
- pocketsphinx
- SpeechRecognition
- watson-developer-cloud
- wit

2.1.1 Installing SpeechRecognition

We can install SpeechRecognition from a terminal with pip:

```
pip install SpeechRecognition
```

Once installed, we should verify the installation by opening an interpreter session and typing:

```
>>>
>>>importspeech_recognitionassr
>>>sr.__version__
'3.8.1'
```

2.1.2 The Recognizer Class

All of the magic in SpeechRecognition happens with the Recognizer class.

The primary purpose of a Recognizer instance is, of course, to recognize speech. Each instance comes with a variety of settings and functionality for recognizing speech from an audio source.

Creating a Recognizer instance is easy. In our current interpreter session, just type:

```
>>>PYTHON
>>>r=sr.Recognizer()
```

- recognize_bing(): Microsoft Bing Speech
- recognize_google(): Google Web Speech API

- recognize_google_cloud(): Google Cloud Speech - requires installation of the google-cloud-speech package
- recognize_houndify(): Houndify by SoundHound

```
PYTHON
>>>r.recognize_google()
```

What happened?

We probably got something that looks like this:

```
Traceback (most recent call last):
```

```
  File "<stdin>", line 1, in <module>
TypeError: recognize_google() missing 1 required positional argument:
'audio_data'
```

2.1.3. Working With Microphones

To access our microphone with SpeechRecognizer, we have installed the PyAudio package.

Installing PyAudio

The process for installing PyAudio will vary depending on our operating system.

Debian Linux

If we're on Debian-based Linux (like Ubuntu) we can install PyAudio with apt:

```
$ sudo apt-get install python-pyaudio python3-pyaudio
```

Once installed, we may still need to run pip install pyaudio, especially if we are working in a virtual environment.

macOS

For macOS, first we will need to install PortAudio with Homebrew, and then install PyAudio with pip:

```
$ brew install portaudio
$ pip install pyaudio
```

Windows

On Windows, we can install PyAudio with pip:

```
$ pip install pyaudio
```

2.2. Testing the Installation

Once we've got PyAudio installed, we can test the installation from the console.

```
$ python -m speech_recognition
```

Make sure our default microphone is on and unmuted. If the installation worked, we should see something like this:

```
A moment of silence, please...
Set minimum energy threshold to 600.4452854381937
Say something!
```

The Microphone Class

Open up another interpreter session and create an instance of the recognizer class.

```
>>>
>>>importspeech_recognitionassr
>>>r=sr.Recognizer()
```

Now, instead of using an audio file as the source, we will use the default system microphone. We can access this by creating an instance of the Microphone class.

```
>>>
>>>mic=sr.Microphone()
```

If our system has no default microphone (such as on a Raspberry Pi), or we want to use a microphone other than the default, we will need to specify which one to use by supplying a device index. We can get a list of microphone names by calling the list_microphone_names() static method of the Microphone class.

Using listen() to Capture Microphone Input

Now that we've got a Microphone instance ready to go, it's time to capture some input.

Just like the AudioFile class, Microphone is a context manager. We can capture input from the microphone using the listen() method of the Recognizer class inside of the with block. This method takes an audio source as its first argument and records input from the source until silence is detected.

```
>>>
>>>withmicassource: ...
audio=r.listen(source)
...
```

Once we execute the with block, try speaking "hello" into our microphone. Wait a moment for the interpreter prompt to display again. Once the ">>>" prompt returns, we're ready to recognize the speech.

```
>>>
```

```
>>>r.recognize_google(audio)
'hello'
```

To handle ambient noise, we'll need to use the adjust_for_ambient_noise() method of the Recognizer class, just like we did when trying to make sense of the noisy audio file. Since input from a microphone is far less predictable than input from an audio file, it is a good idea to do this anytime we listen for microphone input.

```
>>>
>>>withmicassource:
... r.adjust_for_ambient_noise(source)
... audio=r.listen(source)
...
```

Text-to-speech in Python with pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech.

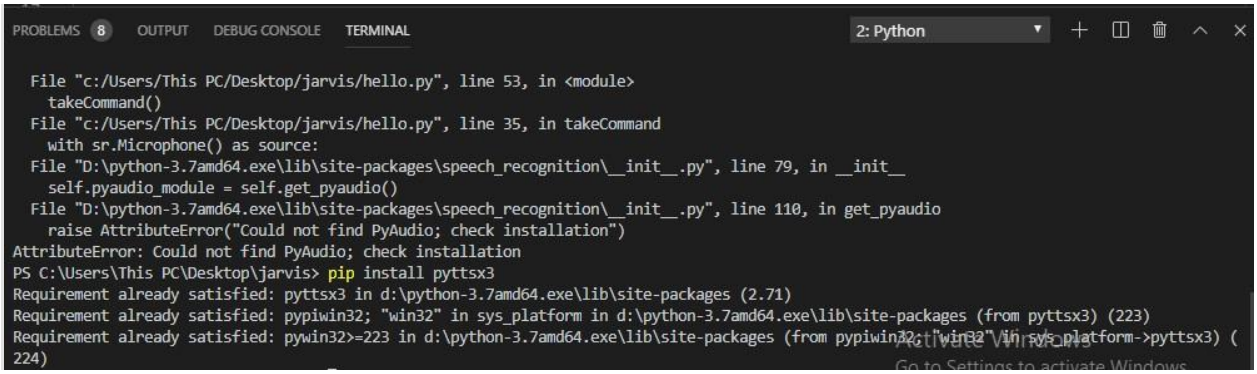
The pyttsx3 module supports two voices first is female and the second is male which is provided by "sapi5" for windows. It supports three TTS engines :

- *sapi5* – SAPI5 on Windows
- *nsss* – NSSpeechSynthesizer on Mac OS X
- *espeak* – eSpeak on every other platform

Installataion

To install the pyttsx3 module, first of all, we have to open the terminal and write

```
pip install pyttsx3
```



2.3. Code : Python program to convert text to speech

```
# Import the required module for text
# to speech conversion
import pyttsx3

#init function to get an engine instance for the speech synthesis   engine
= pyttsx3.init()

#say method on the engine that passing input text to be spoken  engine.say('Hello
sir, how may I help we, sir.')

#run and wait method, it processes the voice commands.
engine.runAndWait() Output :
The output of the above program would be a voice saying,
'Hello sir, how may I help we, sir.'
```

Change voice and language

The voices available will depend on what our system has installed. We can get a list of available voices on our machine by pulling the **voices** property from the engine. Note that the voices we have available on our computer might be different from someone else's machine. There is a default voice set so we are not required to pick a voice. This is only if we want to change it from the default.

```
# Print all available
voices import pyttsx3
engine = pyttsx3.init()
```

```
voices = engine.getProperty('voices') for
voice in voices:
    print "Voice:"
    print(" - ID: %s" % voice.id)     print(" - Name:
%s" % voice.name)     print(" - Languages: %s" %
voice.languages)     print(" - Gender: %s" %
voice.gender)     print(" - Age: %s" % voice.age)
```

Example output from my Windows 10 machine with three voices available.

Simple Mail Transfer Protocol (SMTP) is a protocol, which handles sending email and routing e-mail between mail servers.

Python provides **smtplib** module, which defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon.

Here is a simple syntax to create one SMTP object, which can later be used to send an e-mail –

Install SMTPLib

1. Open up our computer's command prompt.

   On my Windows PC, this is done by going to **Start**,

then typing in "**CMD**", clicking the "**CMD**" app to open it

   up.

2. In the command prompt, type in:

```
pip install smtplib
```

```
import smtplib

smtpObj = smtplib.SMTP([host [, port [, local_hostname]]])
```

Here is the detail of the parameters –

- **host** − This is the host running our SMTP server. We can specify IP address of the host or a domain name like tutorialspoint.com. This is optional argument.

- **port** − If we are providing *host* argument, then we need to specify a port, where SMTP server is listening. Usually this port would be 25.
- **local_hostname** − If our SMTP server is running on our local machine, then we can specify just *localhost* as of this option.

An SMTP object has an instance method called **sendmail**, which is typically used to do the work of mailing a message. It takes three parameters −

- The *sender* − A string with the address of the sender.
- The *receivers* − A list of strings, one for each recipient.
- The *message* − A message as a string formatted as specified in the various RFCs.

3. INTRODUCTION

This system is voice-based personal assistant has always seemed a little out of place in the enterprise. It's a useful tool for search, for reminders, and to write the note just by speaking it up. Window assistant is to create voice apps for the intelligent assistant. When user need to open any other application, he/she can use the command open. E.g. Open Notepad, File explorer, goggle chrome, this will help to open the applications. When user want to write the message can use command write. And to for searching purpose search command can be use. It will also restart and shutdown on the command.

Interactions between a user and our Window assistant skill are mostly free-form, so assistant must understand language naturally and also in context. Window assistant determines what a user wants to do by identifying the user intent from spoken or textual input by utterance.

3.1 MOTIVATION

Assistant initially debuted in 2016 as part of Google's messaging app *Allo,* and it's voice activated speaker *Google Home*. The Assistant have been extended to support a large variety of devices like smart phones, cars, computers etc.. The user can primarily interact with the assistant with the help of voice, though keyboard input is also supported. The Assistant will beable to identify objects and gather visual information through the device's camera, and support purchasing products and sending money, as well as identifying songs.

The features include:

- letting third-party device makers incorporate their own "Actions on Google" commands for their respective products
- incorporating text-based interactions and more languages
- allowing users to set a precise geographic location for the device to enable improved location-specific queries.

3.2 AIM AND OBJECTIVE OF THE PROJECT

The aim of this project is to build an assistant, “Jarvis” for the computer, which will perform the task as an assistant for the device. The computer assistance will be capable of doing a variety of tasks for make a good interface between user and the machine. With the help of this facility the user will be able to manage different tasks with the help of his voice, though keyboard input facility would also prevail. Jarvis will act as an assistant for the user’s laptop proving him the experience of interacting with the person. This type of application will increase the comfort zone of the users along with increase in the interest rate of the user in performing any task in minimal time. The application will include a variety of features for

making the use of the application more familiar and convenient for the user point of view.

4. LITERATURE REVIEW

Virtual desktop assistant is an awesome thing. If you want your machine to run on your command like Jarvis did for Tony. As we know Python is a suitable language for script writers and developers. So we write a script for Personal Voice Assistant using Python. The query for the assistant can be manipulated as per the user's need.

The implemented assistant can open up the application (if it's installed in the system), search Google, Wikipedia and YouTube about the query, calculate any mathematical question, etc by just giving the **voice command.**

We are using **Google speech recognition API** and google text to speech for voice input and output respectively.

Also, for calculating mathematical expression **WolframAlpha API** is used.

4.1 Modules needed

- **pyttsx3:** pyttsx is a cross-platform text to speech library which is platform independent. The major advantage of using this library for text-to-speech conversion is that it works offline. To install this module type the below command in the terminal.

```
pip install pyttsx3
```

- **Speech Recognition:** It allow us to convert audio into text for further processing. To install this module type the below command in the terminal.

```
pip install Speech Recognition
```

- **webbrowser:** It provides a high-level interface which allows displaying Web-based documents to users. To install this module type the below command in the terminal.

```
pip install webbrowser
```

- **Wikipedia:** It is used to fetch a variety of information from the Wikipedia website. To install this module type the below command in the terminal.

```
pip install Wikipedia
```

❖ 4.2 Advantages

- It converts text to speech
- It will assist you to find the applications easily.
- It can be used in windows 7.
- Easy to use
- Can work with variety of commands
- Custom commands
- Secure
- Helpful for disabled
- Artificial intelligence

❖ 4.3 Limitation

- Data need to be entered properly otherwise; outcome may won't be accurate.
- Limited language support
- Costly
- Expensive equipments
- It cannot work in noisy environment

5. DESCRIPTION

The application in built in Jupiter Notebook platform with the use of Python as programing language. The application will start with a greeting to the user say, Good Morning on the basis of the time ie. It's day or night. Then after the assistant will introduce her and then ask for the command to the user. The user will provide the command in micro phone with will be then converted in to the required language with the help of set of codes. If the assistant is unable the get understandable query then it will throw an exception. If the user demands to open any site like Google, Wetube etc . then the assistant will take him to the required site for further actions. The application will also provide the user with 'Play Music ' facilities like playing music in the PC, in listing songs etc.. The user is also provided with the facility to mail anyone with the help of his voice using the assistant. He can perform any Mathematical calculations or get to know the weather forecast at his one call with the help of the assistant, who is receive the voice of the user and provide him the result at verbal form making the whole process more interactive and smart.

The project thus aims to take a step forward in the world where we make the human interaction with machine more innovative and interesting. This application will help us to take one such step.

6. INTERFACE DESCRIPTION

6.1.1 LINE OF CODE :

```
In [4]: import pyttsx3
        import webbrowser
        import smtplib
        import random
        import speech_recognition as sr
        import wikipedia
        import datetime
        import wolframalpha
        import os
        import sys
```

6.1.2 DESCRIPTION :

The above code shows the names of the libraries and software we need to import and install to our system to run the application in Jupyter Notebook platform.

6.2.1 LINE OF CODE :

```
engine = pyttsx3.init('sapi5')

client = wolframalpha.Client('Your_App_ID')

voices = engine.getProperty('voices')
engine.setProperty('voice', voices[len(voices)-1].id)

def speak(audio):
    print('Computer: ' + audio)
    engine.say(audio)
    engine.runAndWait()

def greetMe():
    currentH = int(datetime.datetime.now().hour)
    if currentH >= 0 and currentH < 12:
        speak('Good Morning!')

    if currentH >= 12 and currentH < 18:
        speak('Good Afternoon!')

    if currentH >= 18 and currentH !=0:
        speak('Good Evening!')
```

6.2.2 DESCRIPTION :

The above code provide the application with two functions, speak(audio) and greetMe().

The speak () function pass audio as argument with the help of which the computer assistant gets voice to speak the respective response depending upon users command. The greetMe() function will start the application where the computer assistant Jarvis greets the user.

6.3.1 LINE OF CODE :

```
greetMe()

speak('Hello Sir, I am your digital assistant JARVIS the
speak('How may I help you?')


def myCommand():

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
       # r.pause_threshold =1
        audio = r.listen(source)
    try:
        query = r.recognize_google(audio)
        print('User: ' + query + '\n')

    except sr.UnknownValueError:
        speak('Sorry sir! I didn\'t get that! Try typing
        query = str(input('Command: '))
    except sr.RequestError as e:
        query = str(input('Command: '))

    return query
```

6.3.2 DESCRIPTION :

The above line of code starts with the computer assistant introduction herself as ‘Jarvis’ and asking for the commond by the user.

The code is followed by myCommand() function where the user gives input in the form of voice through microphone and with the help of libraries of google recognizer the voice is converted in to machine understandable form and the

machine give an output response on the basis of command. The code throws exception if any.

6.4.1 LINE OF CODE :

```
if __name__ == '__main__':

    while True:

        query = myCommand();
        query = query.lower()

        if 'open youtube' in query:
            speak('okay')
            webbrowser.open('www.youtube.com')

        elif 'open google' in query:
            speak('okay')
            webbrowser.open('www.google.co.in')

        elif 'open gmail' in query:
            speak('okay')
            webbrowser.open('www.gmail.com')

        elif "what\'s up" in query or 'how are you' in query:
            stMsgs = ['Just doing my thing!', 'I am fine!', 'Nice!', 'I
am nice and full of energy']
            speak(random.choice(stMsgs))
```

6.4.2 DESCRIPTION

The above code provide us with the main() function where the assistant takes command in the form of voice from user, which is then converted in to query in lower_case. The query is then performed and the site is reached for which user asked for.

6.5.1 LINE OF CODE :

```
        elif 'email' in query:
            speak('Who is the recipient? ')
            recipient = myCommand()

            if 'mi' in recipient:
                try:
                    speak('What should I say? ')
                    content = myCommand()

                    server = smtplib.SMTP('smtp.gmail.com', 587)
                    server.ehlo()
                    server.starttls()
                    server.login("shyam1998s@gmail.com", '***********')
                    server.sendmail('shyam1998s@gmail.com', "sonalipriy
a1399@gmail.com", content)
                    server.close()
                    speak('Email sent!')

                except:
                    speak('Sorry Sir! I am unable to send your message
at this moment!')
```

6.5.2 DESCRIPTION :

The code also shows that we can even have access to emails through the computer assistant.

6.6.1 LINE OF CODE :

```
elif 'nothing' in query or 'abort' in query or 'stop' in query:
    speak('okay')
    speak('Bye Sir, have a good day.')
    sys.exit()

elif 'hello' in query:
    speak('Hello Sir')

elif 'bye' in query:
    speak('Bye Sir, have a good day.')
    sys.exit()

elif 'play music' in query:
    music_dir = 'C:\\Users\\DELL\\Music\\Music'
   # music = [music1, music2, music3, music4, music5]
    #random_music = music_folder + random.choice(music) + '.mp3'
    #os.system(random_music)
    songs = os.listdir(music_dir)
    print(songs)
    os.startfile(os.path.join(music_dir,songs[1]))

    speak('Okay, here is your music! Enjoy!')
```

6.6.2 DESCRIPTION :

The above code shows that through the assistant we can also have access to music present in our computer, which includes plying songs and in listing the songs present at our play list.

6.7.1 LINE OF CODE :

```
else:
    query = query
    speak('Searching...')
    try:
        try:
            res = client.query(query)
            results = next(res.results).text
            speak('WOLFRAM-ALPHA says - ')
            speak('Got it.')
            speak(results)

        except:
            results = wikipedia.summary(query, sentences=2)
            speak('Got it.')
            speak('WIKIPEDIA says - ')
            speak(results)

    except:
        webbrowser.open('www.google.com')

speak('Next Command! Sir!')
```

6.7.2 DESCRIPTION :

The above line of code shows that we can provide any query to the assistant to open that in to the search engine, if not that then it will open the query in Wikipedia else it will take we to google for search .

The assistant constantly asks for the next command after completion of one task. If the user is unable to provide the command then the assistant will greet the user as *have a nice day* and exit.

7.RESULTS

File - jarvisfinal

```
 1  C:\Users\DELL\Anaconda3\envs\jarvis\python.ex
    Users\DELL\PycharmProjects\jarvis\jarvisfinal.py
 2  Computer: Good Morning!
 3  Computer: Hello Sir, I am your digital assistant JARVIS
    the Lady Jarvis!
 4  Computer: How may I help you?
 5  Listening...
 6  User: hello
 7
 8  Computer: Hello Sir
 9  Computer: Next Command! Sir!
10  Listening...
11  User: hello how are you
12
13  Computer: Nice!
14  Computer: Next Command! Sir!
15  Listening...
16  User: are you
17
18  Computer: Searching...
19  Computer: Got it.
20  Computer: WIKIPEDIA says -
21  Computer: Fifty Shades Freed is a 2018 American erotic
    romantic drama film directed by James Foley and written
     by Niall Leonard, and based on E. L. James's 2012 novel
     of the same name. It is the third and final installment in
     the Fifty Shades film trilogy, following Fifty Shades of
    Grey (2015) and Fifty Shades Darker (2017).
22  Computer: Next Command! Sir!
23  Listening...
24  User: WhatsApp WhatsApp
25
26  Computer: Searching...
27  Computer: Got it.
28  Computer: WIKIPEDIA says -
29  Computer: WhatsApp Messenger, or simply  WhatsApp,
    is an American freeware, cross-platform messaging and
    Voice over IP (VoIP) service owned by Facebook, Inc. It
    allows users to send text messages and voice messages
    , make voice and video calls, and share images,
    documents, user locations, and other media.
```

1/12

Page 1 of 12

```
70  of exchanging messages ("mail") between people using
     electronic devices. Email entered limited use in the
    1960s, but users could only send to users of the same
    computer, and some early email systems required the
    author and the recipient to both be online
    simultaneously, similar to instant messaging.
71  Computer: Next Command! Sir!
72  Listening...
73  User: send email
74
75  Computer: Who is the recipient?
76  Listening...
77  Computer: Sorry sir! I didn't get that! Try typing the
    command!
78  Command: me
79  Computer: What should I say?
80  Listening...
81  User: project bhejna hai
82
83  Computer: Email sent!
84  Computer: Next Command! Sir!
85  Listening...
86  Computer: Sorry sir! I didn't get that! Try typing the
    command!
87  Command: play music
88  ['(webmusic-6.mp3', '(webmusic.in)_Chand-Sifarish.
    mp3', '(webmusic.in)_Guzarish.mp3', '(webmusic.in)
    _Kaise-Mujhe.mp3', '(webmusic.in)_Mere-Hath-Mein.
    mp3', '001 Ek Raasta Hai Zindagi.mp3', '001. Phool
    tumhe bheja hai.mp3', '002. sawan ka mahina.mp3', '
    004. Kya khoob lagte ho.mp3', '006. tere honton ke do
    phool.mp3', '007 Kya Yahi Pyar Hai.mp3', '007.
    mehboob mere.mp3', '01 - 1920 ER - Apnaa Mujhe Tu
    Lagaa [MP3Khan.Com].mp3', '01 - Aaj Dil Shayarana-
    Holiday  [Pagalworld.com]-190Kbps.mp3', '01 - Bheegi
    Si Bhaagi Si.m4a', '01 - Highway - Patakha Guddi (
    Female Version) [DJMaza.Info].mp3', '01 - Hornn Blow
     - 320 Kbps - DownloadMing.SE.mp3', '01 - Kabhi
    Kabhi(wapking.cc).mp3', '01 - Labon Ka Karobaar (128
     Kbps) - DownloadMing.SE.mp3', '01 - Piya O Re Piya (
    Tere Naal Love Ho Gaya)-(MirchiFun.Mobi).mp3', '01 -
```

3/12

DESCRIPTION :

The above output shows the interaction between the user and the computer assistant .

8. CONCLUSION

- Artificial Intelligence is about replacing human decision making with more sophisticated technologies

- Jarvis is a digital and **virtual assistant** with artificial intelligency. It is very flexible and useful technology. It provides a better interface to deal with it.

- Virtual desktop assistant is an awesome thing. If we want our machine to run on our command like Jarvis did for Tony. Yes it is possible. It is possible using Python.

- Python offers a good major library so that we can use it for making a virtual assistant.

- Python is a suitable language for script writers and developers. We can write a script for Personal Voice Assistant using Python.

9. FUTURE ENHANCEMENT

- Set up more Jarvis Voice Terminals
- Build a Jarvis Android App. Make Jarvis to learn more new skills on its own.
- To make it available to the world.
- Making it S.M.A.R.T
- adding functionality to scrape website.
- Play Gaana.com music with a voice command.
- Randomly giving coding question 1 easy, 1 hard, 1 mid level to hone programming skill, daily.

10. REFERENCES

- www.python.org/ftp/python/3.6.3/python-3.6.3rc1-amd64.exe (04/05/2020)
- www.anaconda.com/download/ (04/05/2020)
- https://www.dataquest.io/blog/jupyter-notebook-tutorial/ (04/05/2020)
- https://www.wikihow.com/Create-a-JARVIS-Like-AI-Assistant (07/06/2020)
- https://github.com/nihal111/J.A.R.V.I.S(07/06/2020)
- https://github.com/Uberi/speech_recognition/blob/master/reference/libraryreference.rst(07/06/2020)
- https://pypi.org/project/pyttsx3/(10/06/2020)
- https://www.instructables.com/id/Send-Email-Using-Python/(10/06/2020)
- https://www.afternerd.com/blog/how-to-send-an-email-using-python-andsmtplib/(11/06/2020)
- http://zetcode.com/python/smtplib/#:~:text=Python%20smtplib%20tutorial%201%20SMTP%202%20The%20smtplib,STARTTLS.%20...%208%20Sending%20mail%20via%20SSL.%20(03/07/2020)
- https://www.slideshare.net/charliedrow/jarsvi(11/07/2020)