

**Natural Language Processing**

**CSE4022**

**Question Answering System using Python libraries (pyttsx)Text to speech**

**Submitted by:**

**Priyanshu Sharma 15BCE1282**

**Aayush Pandey 15BCE1058**

**Kanishk Sankpal 15BCE1368**

**Submitted to:**

**Dr. G. Bharadwaja Kumar**

**Abstract:**

This project is developed as a python application, which will take input commands and perform tasks according to commands. The program will be using Natural language processing to process commands and then it will perform the tasks according to given commands.

Main objective of the project is that the system should respond to the human interaction. System responds in such a way that it completes actions ordered by user for example operations like ‘open word’, query searching, ‘display off’, etc.

Our project will work like Personal assistant which is generally used in many devices like mobiles and laptops and we will be able to control system and web activities using text commands.

**Introduction:**

There have been lot of work done on this field. Like Apple has SIRI, google has Google Now and Microsoft has Cortana. Amazon sells a stand-alone device that, among other things, plays music, reads books aloud and can help buy items through Amazon.

All these firms still have kinks to work out, but the sophistication and range of functions of their personal-assistant software are expanding.

According to Gartner, a research firm, around 38% of American consumers have used virtual-assistant services on their smartphones recently; by the end of 2016 an estimated two-thirds of consumers in developed markets will use them daily. Software robots are getting better at predicting what users need based on past behaviour and current location.

As technologists race to invent the next big thing after the smartphone, many have overlooked what is starting to seem obvious: That the human voice is a powerful, perhaps the most powerful, mechanism for controlling the world around us. And, more importantly, speaking is a behaviour that doesn’t require a user manual.

**Overview:**

User input command is taken as text from the bot\_input function,

in bot\_input speak() is called and then it is returned. After this returned text which is acquired from input is stored in variable data which further is sent to the main processing function Jarvis() which operates for all conditions i.e. operates for operations according to the satisfying condition. Likewise queries are answered according to the type question and further respective block is operated and thus user questions are answered by system functions.

**Methodology:**

* Google offers a restful way to do voice recognition and speech synthesis. The Python library that nicely wraps their text-to-speech API is gTTS.
* gTTS takes advantage of Google Translate’s voice capability and downloads its response to a parameterized GET request.
* There is also another similar way. Pyttsx it cross platform text to speech, which is a great offline option.
* For Windows, you need to install PyWin32 and make sure you have the Microsoft Speech API installed.
* Additionally, pyttsx has difficulty detecting the installed libraries on Windows 10 and must be manually initialized to use ‘sapi5’.

Display off function:

* With win32con it is easy to call Windows API dlls using the ctypes module, defining the constant values for message identifiers and parameter flags.
* From os module process id is acquired from that pid respective processes are stopped when function is called.
* Using sendmessage function of win32, message is sent to system to kill the process with specific time and particular process identification.

Main body:

* Main body runs continuously after each question is answered.
* Input from the used is taken from the bot\_input() function which further passed to variable called data.

Speak function:

* With help of pyttsx module which was downloaded earlier, whatever text which is passed to speak is passed as arguments in say() function of pyttsx.
* Thus system will repeat the text by speaking it aloud.

Input function:

* User input is taken as text, which is passed speak function, speak function repeats it and then it is returned which will further passed in main processing brain function Jarvis()

Processing function (Jarvis()):

* Text which was earlier returned by bot\_input() function is passed in data variable, which is then passed as arguments to jarvis function.
* In Jarvis firstly condition is applied on data to check type of question and then it further proceeded to the respective block of function.
* Using different modules of python questions are answered like in what time is it? block with help of time module it get the system time and is passed to speak function.
* Other functionalities are also done in similar manner.
* For accessing the browser, webrowser module is imported which is used searching directly the text of used like location or any other random text.
* For accessing other os application like word, power point, etc. os module is imported which is further used to start application through startfile() function.

**Brain: The Code**

import pyttsx

#import speech\_recognition as sr

from time import ctime

import time

import os

import webbrowser

from gtts import gTTS

import win32gui

import win32con

from os import getpid, system

from threading import Timer

speech\_engine = pyttsx.init()

def speak(text):

#import pyttsx

#engine = pyttsx.init()

speech\_engine.say(text)

speech\_engine.runAndWait()

def bot\_input():

text = raw\_input('ask your question:')

speech\_engine.say(text)

speech\_engine.runAndWait()

return text

def jarvis(data):

if "how are you" in data:

speak("I am fine")

if "what time is it" in data:

speak(ctime())

if "where is" in data:

data = data.split(" ")

location = data[2]

speak("Hold on kanishk, I will show you where " + location + " is.")

#os.system("mozilla https://www.google.nl/maps/place/" + location + "/&amp;")

webbrowser.open\_new\_tab("https://www.google.co.in/maps/place/" + location)

elif "open Media player" in data:

speak("Hold on, I will open media player")

os.startfile('C:\Program Files (x86)\Windows Media Player\wmplayer.exe', 'open')

elif "search for" in data:

data = data.split(" ")

searchfor = data[2]

speak("Hold on, I will search for " + searchfor)

webbrowser.open\_new\_tab('https://www.google.co.in/?gfe\_rd=cr&ei=c5cTWLeuOqLnugSekrfQAw#q=' + searchfor)

elif "open power point" in data:

speak("Hold on, I will open power point")

os.startfile('C:\Program Files (x86)\Microsoft Office\Office14\POWERPNT.exe', 'open')

elif "open word" in data:

speak("Hold on, I will open word")

os.startfile('C:\Program Files (x86)\Microsoft Office\Office14\WINWORD.exe', 'open')

elif "open excel" in data:

speak("Hold on, I will open excel")

os.startfile('C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.exe', 'open')

elif "display off" in data:#sys.platform.startswith('win'):

def force\_exit():

pid = getpid()

system('taskkill /pid %s /f' % pid)

t = Timer(1, force\_exit)

t.start()

SC\_MONITORPOWER = 0xF170

win32gui.SendMessage(win32con.HWND\_BROADCAST, win32con.WM\_SYSCOMMAND, SC\_MONITORPOWER, 2)

t.cancel()

time.sleep(2)

speak("Hey Kanishk, what can I do for you?")

while 1:

data = bot\_input()

jarvis(data)

"""

import pyttsx

engine = pyttsx.init()

#print "ask your question"

text = raw\_input('ask your question')

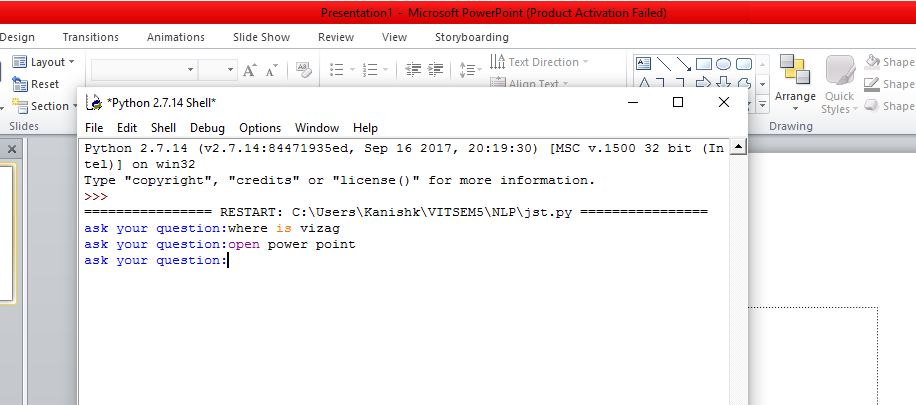
engine.say(text)

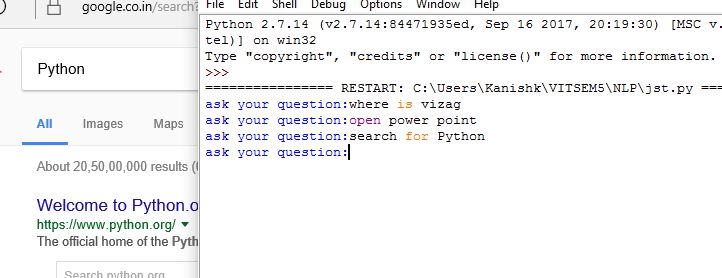
engine.runAndWait()

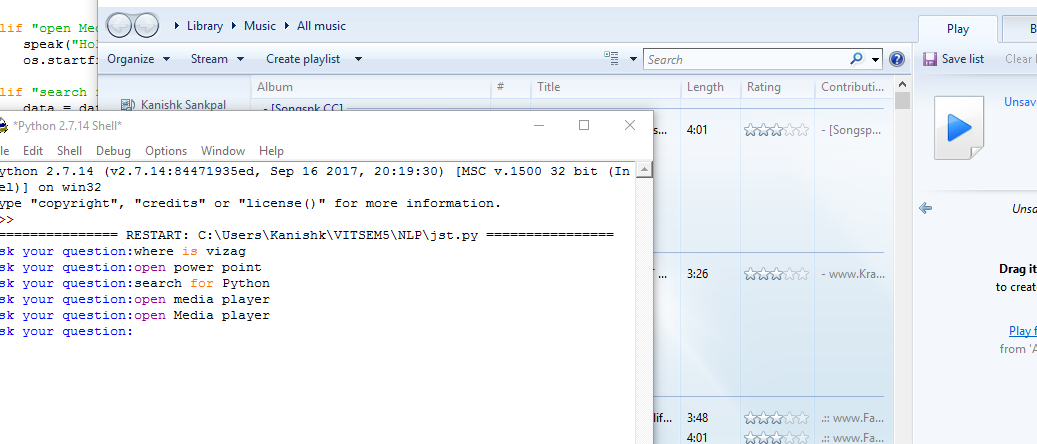
"""

**Snapshots of system while answering questions:**

****

****

****

****

**Conclusion:**

By working throughout this project we realized that in near future technology is going to bring a drastic change in human lifestyle. Study of human language and it use in technology will cut off lot of tedious work, which would result in enhancement of growth.

This simple python based application is able to control system then we can just imagine what will be result if it is carried further for fully automated application which will able to answer all human commands.

**Acknowledgement:**

As successful project completion here we would like to express our gratitude for expertise and excellence of our project mentor Dr. G. Bharadwaja Kumar. His vast knowledge in Natural Language Processing field helped us to resolve our queries during our project work.