1. ***Reduce the delays (means fast response ) between the previous services :***

Python has built-in support for putting your program to sleep. The [time module](https://realpython.com/python-time-module/) has a function [sleep()](https://docs.python.org/3/library/time.html" \l "time.sleep) that you can use to suspend execution of the calling thread for however many seconds you specify.

>>> import time

>>> time.sleep(2) **#** Sleep for 2 seconds

1. ***Merge speech to text , Waston assistant and text to speech in python language :***

*#import library*

import speech\_recognition as sr

*# Initialize recognizer class (for recognizing the speech)*

r = sr.Recognizer()

*# Reading Audio file as source*

*# listening the audio file and store in audio\_text variable*

with sr.AudioFile('I-dont-know.wav') as source:

audio\_text = r.listen(source)

*recoginize\_() method will throw a request error if the API is unreachable, hence using exception handling*

try:

*# using google speech recognition*

text = r.recognize\_google(audio\_text)

print('Converting audio transcripts into text ...')

print(text)

except:

print('Sorry.. run again...')

1. ***Use Python Language in IBM Waston to convert speech to text and text to speech :***

* *Program Python that converts text to speech and speech to text :*

# Python program to translate

# speech to text and text to speech

     import speech\_recognition as sr

import pyttsx3

# Initialize the recognizer

r = sr.Recognizer()

# Function to convert text to

# speech

def SpeakText(command):

# Initialize the engine

    engine = pyttsx3.init()

     engine.say(command)

 engine.runAndWait()

# Loop infinitely for user to

# speak

  while(1):

    # Exception handling to handle

    # exceptions at the runtime

    try:

        # use the microphone as source for input.

        with sr.Microphone() as source2:

           # wait for a second to let the recognizer

            # adjust the energy threshold based on

            # the surrounding noise level

            r.adjust\_for\_ambient\_noise(source2, duration=0.2)

            #listens for the user's input

            audio2 = r.listen(source2)

            # Using ggogle to recognize audio

            MyText = r.recognize\_google(audio2)

            MyText = MyText.lower()

            print("Did you say "+MyText)

            SpeakText(MyText)

    except sr.RequestError as e:

        print("Could not request results; {0}".format(e))

    except sr.UnknownValueError:

        print("unknown error occured")

1. ***Input :*** voice speech (Exemple : Hi , how are you ? ) as **.txt** File
2. ***Output :*** Did you say Hi , how are you , as **mp3** File

Speech Recognition is an important feature in several applications used such as home automation, artificial intelligence, etc.

Finally, to run the speech we use runAndWait() All the say() texts won’t be said unless the interpreter encounters runAndWait().