**Libraries used:**

* Googletrans
* Pyaudio:
* <https://people.csail.mit.edu/hubert/pyaudio/docs/>
* <https://pypi.org/project/PyAudio/>
* PyAudio provides Python bindings for PortAudio v19, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a variety of platforms, such as GNU/Linux, Microsoft Windows, and Apple macOS.
* SpeechRecognition
* <https://pypi.org/project/SpeechRecognition/>
* <https://www.geeksforgeeks.org/speech-recognition-in-python-using-google-speech-api/>
* <https://realpython.com/python-speech-recognition/>
* Library for performing speech recognition, with support for several engines and APIs, online and offline.
* Speech Input Using a Microphone and Translation of Speech to Text: **Allow Adjusting for Ambient Noise:**Since the surrounding noise varies, we must allow the program a second or two to adjust the energy threshold of recording so it is adjusted according to the external noise level. **Speech to text translation:** This is done with the help of Google Speech Recognition. This requires an active internet connection to work. However, there are certain offline Recognition systems such as PocketSphinx, that have a very rigorous installation process that requires several dependencies. Google Speech Recognition is one of the easiest to use. You can now invoke recognize\_google() to attempt to recognize(transcribe: audio to text) any speech in the audio. Depending on your internet connection speed, you may have to wait several seconds before seeing the result.
* Recognizing speech requires audio input, and SpeechRecognition makes retrieving this input really easy. Instead of having to build scripts for accessing microphones and processing audio files from scratch, SpeechRecognition will have you up and running in just a few minutes.
* The SpeechRecognition library acts as a wrapper for several popular speech APIs and is thus extremely flexible. One of these—the Google Web Speech API—supports a default API key that is hard-coded into the SpeechRecognition library. That means you can get off your feet without having to sign up for a service.
* **Requirements**
* Python 3.8+ (required).
* PyAudio 0.2.11+ (required only if you need to use microphone input, Microphone).
* PocketSphinx (required only if you need to use the Sphinx recognizer, recognizer\_instance.recognize\_sphinx)
* Google API Client Library for Python (required only if you need to use the Google Cloud Speech API, recognizer\_instance.recognize\_google\_cloud)
* FLAC encoder (required only if the system is not x86-based Windows/Linux/OS X)
* Vosk (required only if you need to use Vosk API speech recognition recognizer\_instance.recognize\_vosk)
* Whisper (required only if you need to use Whisper recognizer\_instance.recognize\_whisper)
* openai (required only if you need to use Whisper API speech recognition recognizer\_instance.recognize\_whisper\_api)
* The following requirements are optional, but can improve or extend functionality in some situations:If using CMU Sphinx, you may want to install additional language packs to support languages like International French or Mandarin Chinese.
* Gtts:
* <https://pypi.org/project/gTTS/>
* <https://gtts.readthedocs.io/en/latest/>
* gTTS (Google Text-to-Speech), a Python library and CLI tool to interface with Google Translate text-to-speech API
* Write spoken mp3 data to a file, a file-like object (bytestring) for further audio manipulation, or stdout.It features flexible pre-processing and tokenizing.
* **Features:**
* Customizable speech-specific sentence tokenizer that allows for unlimited lengths of text to be read, all while keeping proper intonation, abbreviations, decimals and more;
* Customizable text pre-processors which can, for example, provide pronunciation corrections;

**Installation:**

pip install googletrans

pip install pyaudio

pip install SpeechRecognition

pip install gtts

**Code Snippet:**

# Importing necessary modules required

import speech\_recognition as spr

from googletrans import Translator

from gtts import gTTS

import os

# Creating Recogniser() class object

recog1 = spr.Recognizer()

# Creating microphone instance

mc = spr.Microphone()

# Capture Voice

with mc as source:

    print("Speak 'hello' to initiate the translation !")

    print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

    recog1.adjust\_for\_ambient\_noise(source, duration=0.2)

    audio = recog1.listen(source)

    MyText = recog1.recognize\_google(audio)

    MyText = MyText.lower()

# Here initialising the recorder with

# hello, whatever after that hello it

# will recognise it.

if 'hello' in MyText:

    # Translator method for translation

    translator = Translator()

    # short form of english in which

    # you will speak

    input\_lang = 'en'

    # In which we want to convert, short

    # form of hindi

    output\_lang = 'hi'

    with mc as source:

        print("Speak a stentence.......")

        recog1.adjust\_for\_ambient\_noise(source, duration=0.2)

        # Storing the speech into audio variable

        audio = recog1.listen(source)

        # Using recognize.google() method to

        # convert audio into text

        get\_sentence = recog1.recognize\_google(audio)

        # Using try and except block to improve

        # its efficiency.

        try:

            # Printing Speech which need to

            # be translated.

            print("Statement to be Translated :"+ get\_sentence)

            # Using translate() method which requires

            # three arguments, 1st the sentence which

            # needs to be translated 2nd source language

            # and 3rd to which we need to translate in

            text\_to\_translate = translator.translate(get\_sentence,

                                                     src= input\_lang,

                                                     dest= output\_lang)

            # Storing the translated text in text

            # variable

            text = text\_to\_translate.text

            # Using Google-Text-to-Speech ie, gTTS() method

            # to speak the translated text into the

            # destination language which is stored in to\_lang.

            # Also, we have given 3rd argument as False because

            # by default it speaks very slowly

            speak = gTTS(text=text, lang=output\_lang, slow= False)

            # Using save() method to save the translated

            # speech in capture\_voice.mp3

            speak.save("translated\_speech.mp3")

            # Using OS module to run the translated voice.

            os.system("start translated\_speech.mp3")

        # Here we are using except block for UnknownValue

        # and Request Error and printing the same to

        # provide better service to the user.

        except spr.UnknownValueError:

            print("not able to understand the input speech")

        except spr.RequestError as e:

            print("not able to provide required uutput".format(e))