# A PROJECT REPORT ON

# LANGUAGE RECOGNITION FROM SPEECH USING MACHINE LEARNING AND AUDIO TRANSLATION

***Major project submitted in partial fulfillment of the requirements for the award of the degree of***

## MASTER OF TECHNOLOGY

## IN

## DATA SCIENCE
## (2020-2022)

## BY

**Summayya Banu 20241DB007**

***Under the esteemed guidance of***

**Dr. K. Prasanna Lakshmi**

**Head of the Department, Dept of IT**


---

**DEPARTMENT OF INFORMATION TECHNOLOGY**
**GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY**
**(AUTONOMOUS)**
**HYDERABAD**


# CERTIFICATE

This is to certify that it is a bonafide record of Mini Project work entitled **"Language Recognition From Speech Using Machine Learning And Audio Translation"** done by **Summayya Banu (20241DB007),** student of **M.Tech (DS)** in the Department of Information Technology, Gokaraju Rangaraju Institute of Engineering and Technology during the period 2020-2022 in the partial fulfillment of the requirements for the award of degree of **MASTER OF TECHNOLOGY IN DATA SCIENCE** from GRIET, Hyderabad.

Dr. K. Prasanna Lakshmi
(Internal Project Guide)

Dr. K. Prasanna Lakshmi
(Head of the Department)

(Project External)

# ACKNOWLEDGEMENT

We take the immense pleasure in expressing gratitude to our Internal guide **Dr. K. Prasanna Lakshmi, Head of the Department,** Information Technology, GRIET. We express our sincere thanks for her encouragement, suggestions and support, which provided the impetus and paved the way for the successful completion of the project work.

We wish to express our gratitude to **Dr. K. Prasanna Lakshmi**, **P. Gopala Krishna,** our Project Co-coordinator **K. Archana,** for her constant support during the project.

We express our sincere thanks to **Dr. Jandhyala N Murthy,** Director, GRIET**,** and **Dr. J. Praveen,** Principal, GRIET**,** for providing us the conductive environment for carrying through our academic schedules and project with ease.

We also take this opportunity to convey our sincere thanks to the teaching and non-teaching staff of GRIET College, Hyderabad.

Email: summibanu16@gmail.com
Contact No: 7674019697
Address: Tolichowki, Hyderabad

# DECLARATION

This is to certify that the project entitled "**Language Recognition From Speech Using Machine Learning And Audio Translation"** is a bonafide work done by us in partial fulfillment of the requirements for the award of the degree **MASTER OF TECHNOLOGY IN DATA SCIENCE** from **Gokaraju Rangaraju Institute of Engineering and Technology**, Hyderabad.

We also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites, books and paper publications are mentioned in the Bibliography.

This work was not submitted earlier at any other University or Institute for the award of any degree.

***Summayya Banu*** ***20241DB007***

# TABLE OF CONTENTS

# LIST OF FIGURES

# ABSTRACT

Language identification is used to determine the language being spoken in audio when compared against a list of provided languages. This project displays the application and implementation of machine learning in speech recognition. A program will be developed to identify the language of the real time audio. It will provide information about the method of speech recognition and get the desired results.

With the use of machine learning, we will help in the real world to prepare automated voice translation without having to manually select the language or even when we are unaware of the language being spoken. We will be creating an application to trigger the program and then using the operating system sound player, we will play the translated audio.

We aim to simplify the audio detection and translation process by creating a user-friendly interface so that it can be implemented by anyone.

# 1. INTRODUCTION

## 1.1 Introduction to the Project

An average person around speaks eleven thousand to twenty five thousand words per day making speech the easiest way to express ourselves. No matter if it is a conversation, a dialogue, or a speech, or any other presentations or general talks, we make use of speech to make others and as well as our own selves understand the actions and thoughts. If any of the side in a conversation, is ignorant of the language being spoken, the purpose of the conversation is not achieved. Therefore, we require a tool that can connect such this language barriers. Speech-to-speech translation is a system that can play a predominant role by ease of communication between people speaking in many languages. Efforts are being made throughout the world to achieve the goal and implement it practically for use by common man.

With the era of global and endless economies, the exchange of information has become necessary and communication is the traditional and best way. Global scripting increases the need for communication between native speakers of different languages. In fact, one of the biggest challenges facing information technology is to overcome language barriers in the global community and allow them to express their emotions.

The objective of this project is to explore the field of automated speech translation to English, in particular. And in this modern world, people from different parts of the world use different languages. The only language which is globally recognized and use popularly is English. Many international companies and officials organize meetings in English. To make everyone familiar of English, universities offer courses for the same. English is spoken as a common language for tourists and travelers around the world.

In the age of globalization, learning english is very important and it can help improve communication in India and beyond. Although India has the second largest number of English speakers in the world, research has shown that many young children from India are not fluent in English.

## 1.2 Existing System

The use of manually translating texts is limited to crucial official documents, press articles and few award-winning literary works. The complications of speech recognition is wide spread in our time. Latest tools and technologies helped engineers create several voice assistants, modules which are designed to work with specific voice commands and speech identifiers. Other devices are also intended to work with processed text such as translators and classifiers.

Despite readymade solutions from well-known world companies, the language must be selected manually in all translators.

## 1.3 Proposed System

With the use of machine learning, we will help in the real world to prepare automated voice translation without having to manually select the language or even when we are unaware of the language being spoken. We will be creating an application start the program and then using the operating system sound player, we will play the translated audio.

We aim to simplify the audio detection and translation process by creating a user-friendly interface so that it can be implemented by anyone, for any language.

# 2. PROJECT REQUIREMENT ENGINEERING

## 2.1 Software Requirements

- Python
- Anaconda Navigator
- Windows 10
- Anaconda IDE (Jupyter Notebook)

## 2.2 Hardware Requirements

- 4GB Ram
- Microphone
- Audio Player
- Internet connection (Wired/Wireless)

# 3. LITERATURE SURVEY

In 1952, Audrey system was developed at Bell Laboratories as the first speech recognition system that only recognized numbers spoken by a person. Ten years later, in 1962, IBM created another system that recognizes 16 English words. In collaboration with the Soviet Union, the United States, and England Japan has developed hardware that had the capability of detecting and analyzing 4 voices and 9 letters.

Later on, Carnegie Mellon's "harpy" speech system recognized 1,011 words created in the years 1971 and 1976. Threshold Technology and Bell Laboratories are the first commercial speech recognition companies that were able to perform interpreting voices of different people. A new statistical method called Hidden Markov model (HMM) was introduced in 1980, expanded to match one hundred words to several thousand words. In 1985, Kurzweil, the speech synthesis recognized 5000 words, that is founded by IBM. After that, children can practice on their voices teach ‘Julie doll’ from Worlds of Wonders in 1987.

Ten years later, Google Integrated English voice search system with 230 billion words of the actual user. In 2015, voice recognition was introduced by Google experimented with connectionist temporal classification Long-term memory approaches (LSTM) (CTC) that's implemented in Google Voice.

There are also several practices and cases regarding the translation from one language to the other. If translation is an art, it is not easy. In the thirteenth century, Roger Bacon wrote that the translation is correct, that the translator must know languages as well as knowledge of the translation; and after several interpreters did that, he wanted to accomplish the translation process without the need of translators.

In the 1950's, mechanical research for translation became a reality, although information about it was not available in the early 17th century. The Georgetown experiment, in which more than sixty Russian phrases were translated in 1954, is one of the most important projects to date. Georgetown study researchers have confirmed that the reliability of the transmission problem was resolved within three to five years. Such a study was recently conducted in the Soviet Union. This success began with a major investment in engineering research in the United States. Progress is slower than expected; In 1966, an ALPAC survey report showed that 10 years of experience in Georgetown was unpredictable and led to a sharp decline in funding. Interest in computer translation grew, and in the 1980's it became commonplace and cheap as existing computer power increased.

Although there are only “fully automatic, with great free transfer capabilities”, there are many programs today that can provide tangible results with strict limits. Many of these programs are available online, such as Google Translate and SYSTRAN, which enables AltaVista's BabelFish (now Yahoo's Babelfish as of May 9, 2008).

# 4. TECHNOLOGY

## 4.1 Python - Programming Language

In our project, we will be using Python programming language to create our project. Python is an Object Oriented, easy to code, fun programming language. The number of lines or codes that we enter in this language to perform a function is far lesser in Python, in comparison with other languages.

They are also many open-source free libraries available to make our job a lot easier. We can directly use the packages that we need and use the inbuilt functions for most of the operations, instead of defining functions every time. Apart from this, it allows easy collaboration between the predefined and the user defined functions.

Python is also an for web application development, graphical user interface (GUI) and software development. In fact, it was clearly in demand and has been used by Instagram, YouTube and Spotify on board at a rapid pace.

## 4.2 Use of Python in the field of Data science

Data Science analyses data to gather insights by using various tools, algorithms, processes and systems. Both structured and unstructured data are converted into information that can be used in a wide range of areas.

- Flexiblity

  Python allows programmers to focus on a task or activity without having to worry about achieving the goal.

- Understandable

  Python focuses on simplicity and readability. With the comfort of learning makes Python an exemplary tool for Data Scientish aspirant and Data Science programmers. Python offers developers the advantage of using fewer lines of code to perform more than one task when compared to using older languages. In other words, you spend more time playing with it and less time on code.

- Open Source

  Python is an open source language, that means it's free and uses social media for development. Python is designed to work in both Windows and Linux environments. It can also be easily broadcast to multiple locations. There are plenty of open Python libraries such as data manipulation, data drawing, statistics, math, machine learning, and natural language manipulation, to name a few.

## 4.3 Python IDE - Jupyter Notebook

Jupyter is a free, open-source, interactive web computational notebook, that researchers can use to write, combine software code, computational output, explanatory text and multimedia resources in a single document. It acts as interface using which we can write and execute our python code. Computational notebooks have been around for decades, but Jupyter in particular has exploded in popularity over the past couple of years. This rapid uptake has been aided by an enthusiastic community of user–developers and a redesigned architecture that allows the notebook to speak dozens of programming languages — a fact reflected in its name, which was inspired, according to co-founder Fernando Pérez, by the programming languages Julia (Ju), Python (Py) and R.

### 4.3.1 Installation of Python and Jupyter Notebook

The easiest way to get started with Jupyter Notebooks is by installing Anaconda. Anaconda is the most widely used Python distribution for data science and comes pre-loaded with all the most

popular libraries and tools. Some of the biggest Python libraries included in Anaconda include NumPy, pandas, and Matplotlib, though the full 1000+ list is exhaustive.
Anaconda thus lets us hit the ground running with a fully stocked data science workshop without the hassle of managing countless installations or worrying about dependencies and OS-specific (read: Windows-specific) installation issues.
To get Anaconda, simply:

1. Download the latest version of Anaconda for Python 3.8 using the link: https://www.anaconda.com/products/individual
2. Install Anaconda by following the instructions on the download page and/or in the executable.

After installing Python already installed and prefer to manage your packages manually, you can just use pip function in the computer's command prompt:

- $ pip3 install jupyter

## 4.4 Python Packages

There are many open source free python packages available that we can easily install for our usage. The packages that we will be using in our project are as follows:

- Speech_recognition Package
- Googletrans Translator Package
- Natural Language Toolkit (NLTK) Package
- Google Text-To-Speech (gTTS) Package
- Tkinter Package
- OS Package
- Sys Package

### 4.4.1 Speech_Recognition Package

Speech recognition, as the name suggests, refers to automatic recognition of human speech by machine. Speech recognition is one of the most important tasks in the domain of human computer interaction. This task of detecting the human speech is accomplished by this package.

We will be using speech recognizer to recognize the audio and convert it into text. To perform this operation, we will be using the python inbuilt library SpeechRecognition.

We SpeechRecognition Library is used to perform speech recognition, with support for several engines and APIs, online and offline.

- Speech recognition engine/API support:
- Google Speech Recognition
- Google Cloud Speech API
- Microsoft Bing Voice Recognition
- Houndify API
- IBM Speech to Text

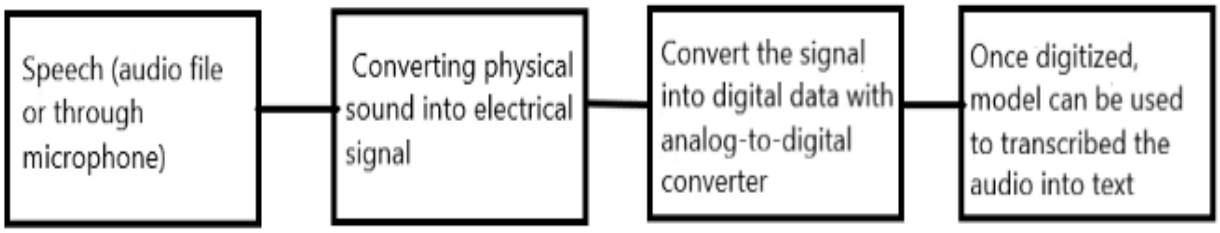


Fig 1: Speech Recognition process

Speech Recognizer package can be installed using the command:

- $ pip install speech_recognition

### 4.4.2 Googletrans Translator Package

After the audio is recognized and converted into text in the first stage, the text needs to be translated. Googletrans is a free and unlimited python library that implemented Google Translate API. This uses the Google Translate Ajax API to make calls to such methods as detect and translate.

Features:

- Fast and reliable - it uses the same servers that translate.google.com uses
- Bulk translations
- Auto language detection
- Customizable service URL

This package needs to be installed using the following command:

- $ pip install googletrans

### 4.4.3 Natural Language Toolkit (NLTK) Package

After recognizing the text, we will process the text to create meaningful sentences. To perform this operation, we will be using the NLTK Library.

Natural language processing (NLP) is a field that focuses on making natural human language usable by computer programs. NLTK is a leading platform for building Python programs to work with human language data. The NLTK module is a massive tool kit, aimed at helping you with the entire Natural Language Processing (NLP) methodology. NLTK helps the computer to analyse, preprocess, and understand the written text.

To install this, we need to run the following command.

- $ pip install nltk

### 4.4.4 Google Text-To-Speech (gTTS) Package

After the text is translated, we need our program to read the text. This can be accomplished by using the Google Text-To-Speech package. **GTTS (**Google Text-to-Speech) is a Python library interface with Google Translate's text-to-speech API.

Here, it writes the spoken data to a file, a file-like object for further audio manipulation, or to simply play the audio.

The gTTS package can be installed using the command:

- $ pip install gTTS

### 4.4.5 Tkinter Package

Our front end GUI will work by using Tkinter library. Python offers various options to develop a Graphical User Interface. Out of all the GUI methods, tkinter is the most commonly used method. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

The Tkinter library ("Tk interface") is the default Python interface for the Tcl/Tk GUI toolkit. This package is easily available for most Unix platforms, including macOS and Windows systems. Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python and to build something that’s functional and cross-platform quickly.

We need to add this library by using the following command:

- $ pip install tk

Note that we may have to check the python version when installing this package.

### 4.4.6 Other Packages

Apart from the mentioned packages, we also need to install the ‘os’ and ‘sys’ Package to interact with the computer operating system and use the resources such as starting the microphone, triggering the operating system to play the audio etc.

To install these packages, we need to execute the following commands:

- $ pip install os
- $ pip install sys

## 4.5 Languages Supported

- Our program supports up to 79 languages for translation. These language packages are taken from the Google Translator API.
- The Languages supported are: Afrikaans, Albanian, Arabic, Armenian, Assamese, Azerbaijani, Basque, Belarusian, Bengali, Bonsian, Bulgarian, Burmese, Cantonese, Catalan, Chinese (Simplified), Chinese (Traditional - Taiwan), Corsican Croatian, Czech, Danish, Dutch, English, Esperanto, Estonian, Filipino, Finnish, French, German, Greek, Gujarati, Hebrew, Hindi, Hmong, Hungarian, Icelandic, Indonesian, Irish, Italian, Japanese, Kannada, Kazakh, Khmer, Korean, Kurdish, Kyrgyz, Ladakhi, Lao, Latin, Latvian, Lithuanian, Luxembourgish, Macedonian, Malagasy, Malay, Malayalam, Maltese, Marathi, Mongolian, Nepali, Norwegian, Polish, Portuguese (Brazil), Portuguese (Portugal), Romanian, Russian, Serbian, Sindhi, Sinhala, Slovak, Slovenian, Spanish, Sundanese, Swahili, Swedish, Tamil, Telugu, Thai, Turkish, Ukrainian, Urdu, Uzbek, Vietnamese, Welsh.

# 5. DESIGN REQUIREMENT ENGINEERING

## 5.1 Architecture of the Project

This section shares the basic architecture of the project. It shows the flow of operations in our program.

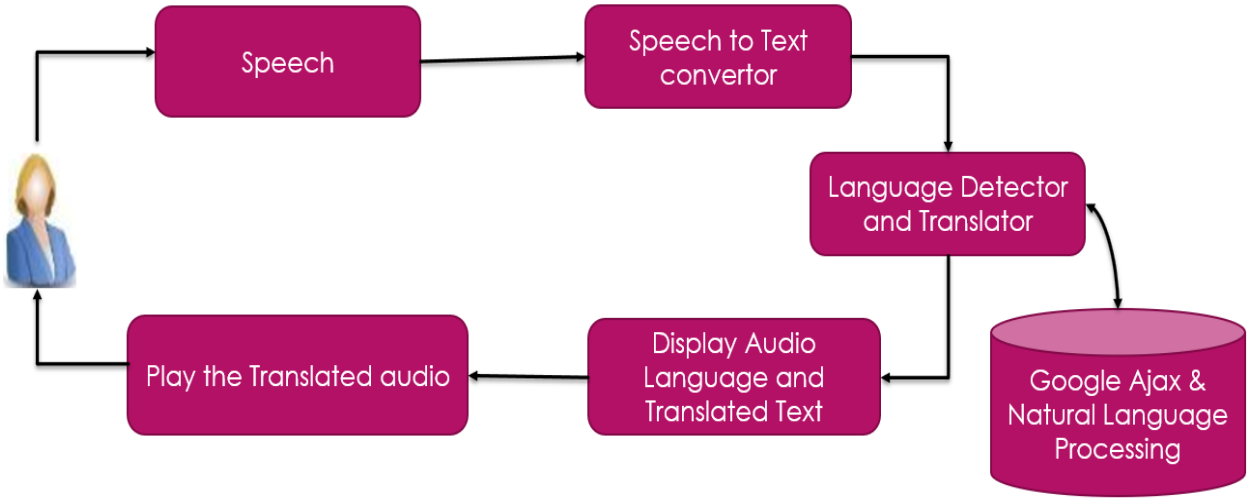


Fig 2: Architecture of the project

## 5.2 UML Diagrams

UML is an abbreviation that stands for Unified Modeling Language. UML diagrams are based on the Unified Modeling Language with the intention of visually portraying a system along with the main actors, actions, functions or classes. This aims to better understand, and maintain the document information about the system. As the old adage says, "A picture is worth a thousand words."

Through visual and graphical representations, we can better understand and detect any possible malfunctions in the software or in the implementation of our project.

### 5.2.1 Use Case Diagram

Use Case diagrams displays the simple overview of the actors and their functions in our project.

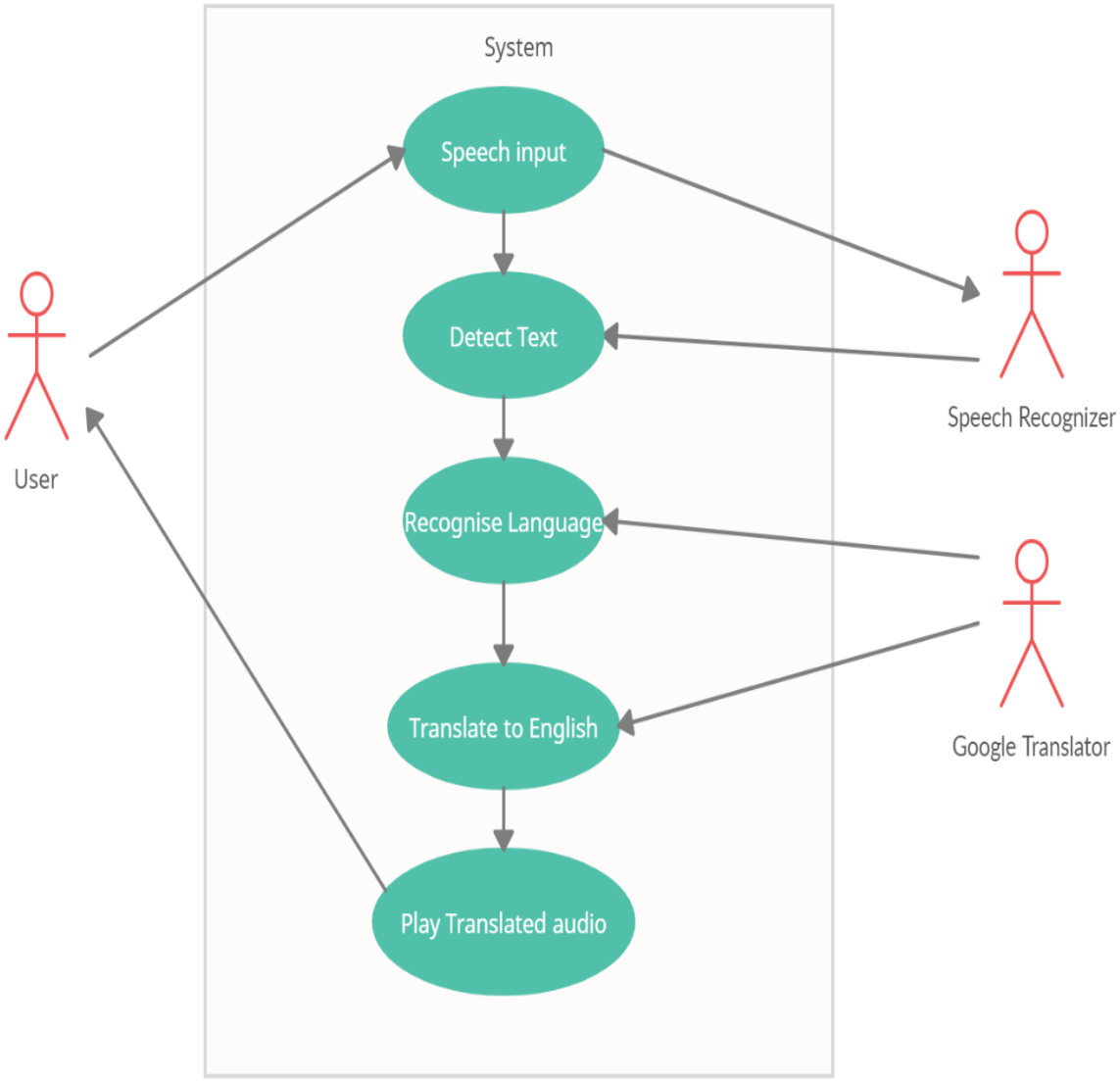


Fig 3: Use Case Diagram

### 5.2.2 Class Diagram

Class diagrams shows the classes with their attributes (called as fields) and characters (also called member functions). Precisely, each class has 3 areas: the name of the class, the attributes of the class directly below the name, and the operations/characters below the class.

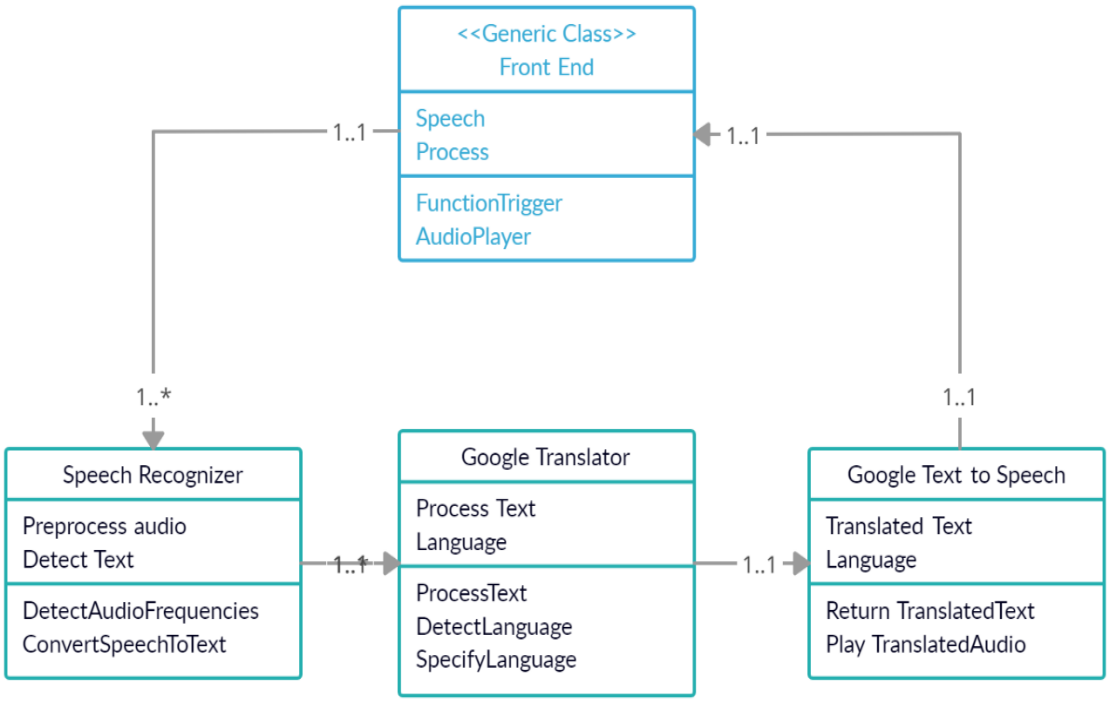


Fig 4: Class Diagram

### 5.2.3 Activity Diagram

Activity diagrams are nowadays, the one of crucial UML diagrams for program modeling processes. It is often used in software development to describe a sequence of different actions and activities. They describe how the objects are used, consumed/produced during any activity of the project and the connection between different activities.

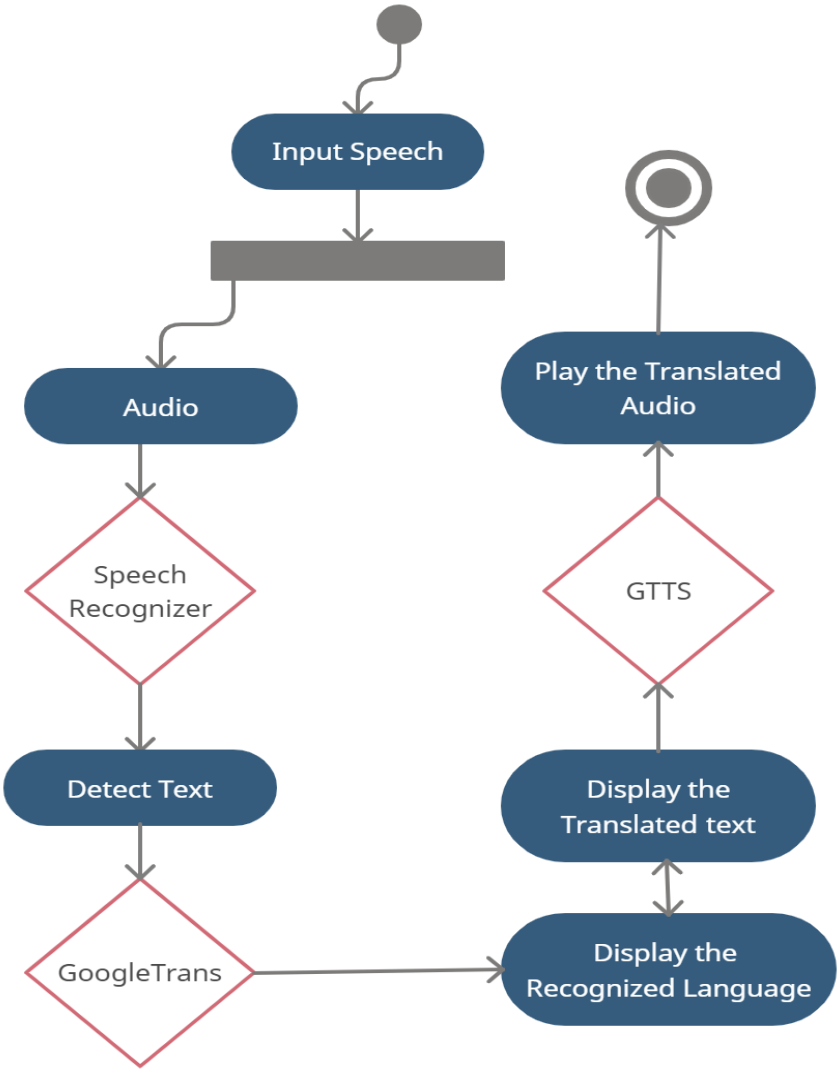


Fig 5: Activity Diagram

### 5.2.4 Sequential Diagram

Sequence diagrams are used the order of messages and communications between all the participants and objects. The actors or objects can be active when needed or when another object wants to interact with them. Here, all the messages are presented in chronological order.

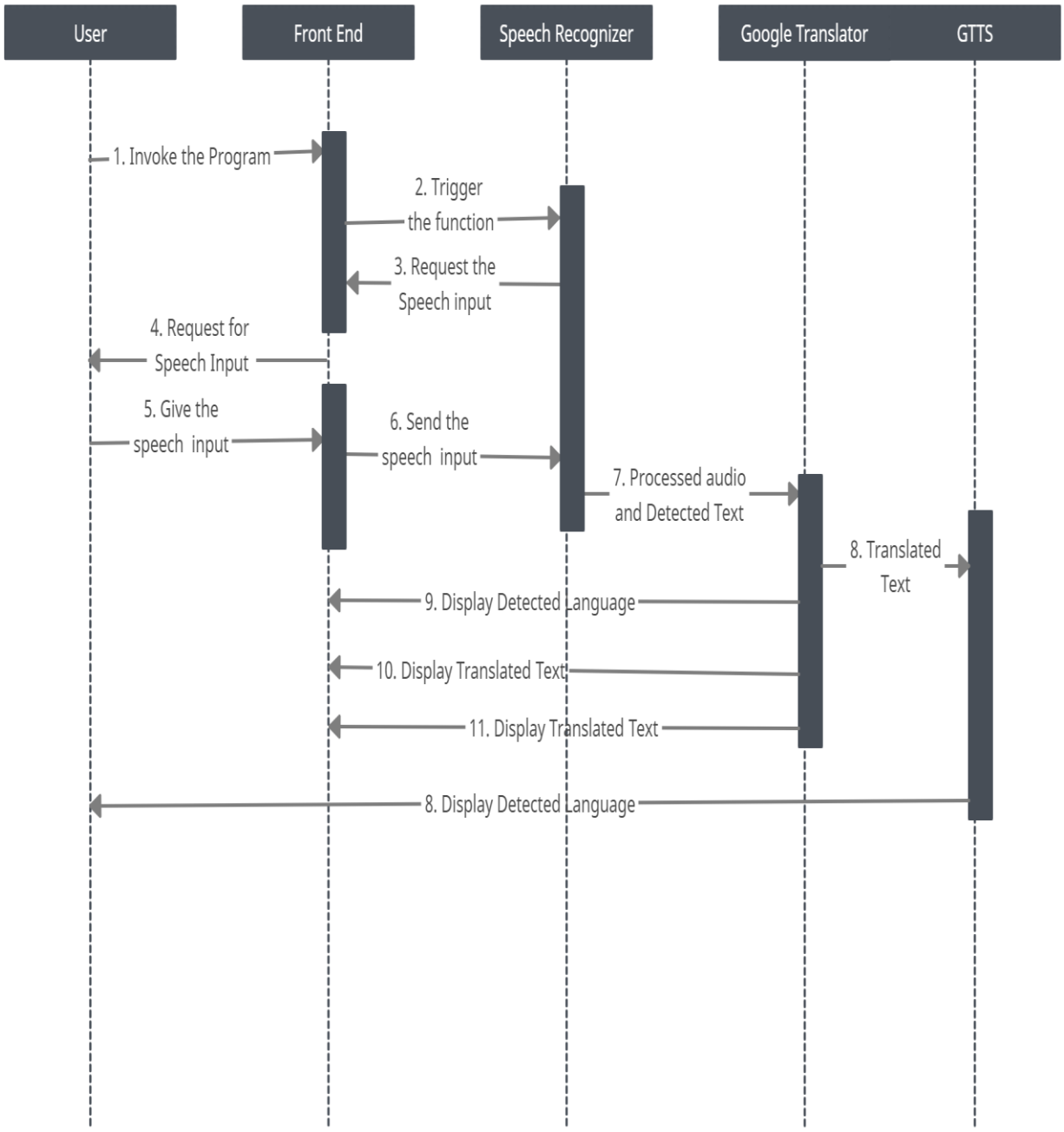


Fig 6: Sequence Diagram

# 6. IMPLEMENTATION

The basic implementation of our project can be represented as follows:

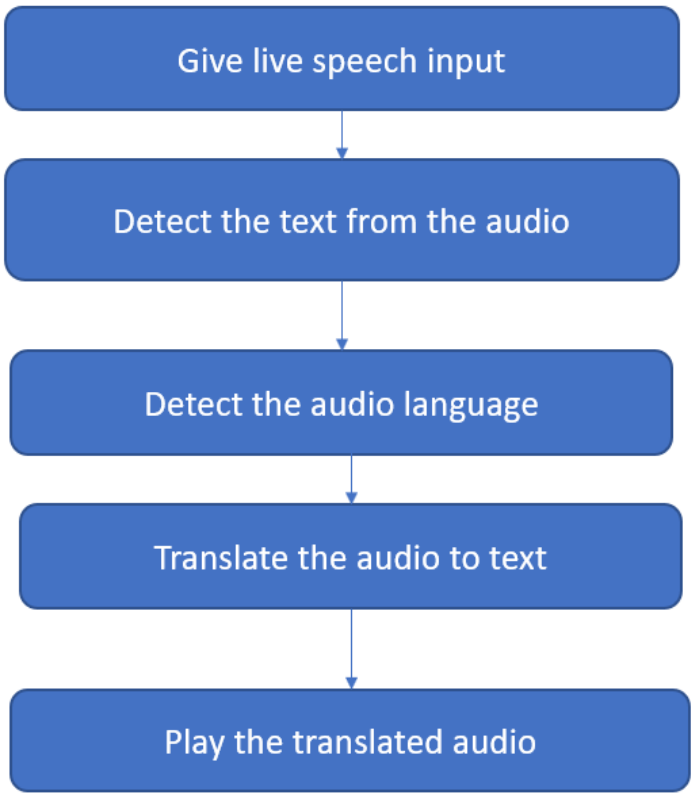


Fig 7: Implementation Overview

## 6.1 Source Code

After the implementation of our project is picturized, we need to write the code in Python to execute our project.

The source code is the set of human-understandable instructions which are written usually to process the words when he is developing a program. The source code is given to a compiler to turn it into machine understandable code, also known as the object code, that a computer can understand and execute. Object code is usually constituted of 1s and 0s, so it isn't human-readable.

It stored information on declarations, functions and instructions etc, that act as instructions for our program implementation.

Here, we will be mentioning parts of the code and its related explanation.

The first step would be to import all the packages in our program.

```
import speech_recognition as spr
from googletrans import Translator
import nltk as nlp
from gtts import gTTS
import os
import sys
```

These are the packages that we discussed in the Python Packages section. Each package has its own purpose and comes with a set of predefined functions. They are usually defined at the start of the program.

The Speech_recognition package helps to execute functions and recognize the audio.

GoogleTrans is used to identify the text, auto detect the language of the text and then translate that text into any desired language. In our project, we will display the translation of the input speech to English.

The nltk (Natural language toolkit) is used to process the text into meaningful sentences.

The gTTS is the package name for Google Text to speech. This package contains functions to convert the text into speech.

The os and sys packages contains functions to trigger the operating system to start the microphone whenever required and then stop the microphone. Not just that, they also trigger the inbuilt sound player to play the translated audio whenever prompted.

Once we load all the necessary packages, we need to start the body of the code.

Next, we will be defining the __inti__ function. __init__ is the class constructor. The self parameter is used to point to the object instance.

```
def __init__(self ,translator):
    self.translator = Translator.translate()
```

Here, we are creating an object reference for the Translator function, taken from the package googleTrans.

The next step would be to create a user defined function definition.

```
def myTranslator(): #user defined functions
    recog1 = spr.Recognizer()
    mc = spr.Microphone() #setting microphone as our source for speaker
    to_lang = 'en'
    translator = Translator()
#Caling the translator function from the pac
kage GoogleTrans
    with mc as source:
        print("Speak a stentence...")
        recog1.adjust_for_ambient_noise(source)
         #adjusting for any ambient noise
        audio = recog1.listen(source,phrase_time_limit=5)
      #capturing the audio from the speaker. Here we have set the limit as
       5 secs. This can be changed as per our requirements
        get_sentence = recog1.recognize_google(audio)
      #using the recognize function to convert speech-to-text
    text_to_translate = translator.translate(text=get_sentence,dest='en')
      #Defining the Translate function
    print("Detected Language Code:",text_to_translate.src)
      # Printing the detected source language.
    print("Detected Text:"+ get_sentence)
      # Displaying the detected text from the speech input
    text = text_to_translate.text
      #Storing the Translated text to the variable Text, while calling the
 function to translate
```

```
    print("Translated Text:",text)
      # Printing the Translated Text
    speak = gTTS(text=text, lang=to_lang, slow= False)
      #converting text to speech and storing it in the variable speak

    speak.save("captured_voice.mp3")
      # saving the converted speech into an .mp3 format for further usage.
    os.system("start captured_voice.mp3")
       #trigerring the Operating system to run the audio file from the def
ault sound player
```

Here, we have explained the purpose of each sentence using the # command. Each and every command has its purpose. The sequence of these commands also plays a major role for the flow of our program.

So far, we have only defined the functions and created our on translator function. However, we haven't yet called these functions.

In our project, we want to create a user friendly interface to execute our program. Let's now go through the coding of the Graphical User interface.

We will have to first import the packages required to create our GUI.

```
from tkinter import*
import tkinter as tk
```

Once the package is loaded, we next customize our User interface and display how our front end will look like.

```
window = Tk() #defining a name for our GUI for refernce and calling the GU
I function
```

```
window.title('GRIET Project') #defining a title of our GUI
window.geometry("450x150")  #defining a size for our GUI window
window.config(background = "white") #Selecting the background color
```

So far, we have defined functions to customize the appearance of our frond end. Now, let's insert some text inside our user interface to make it easily understandable.

```
Display_label= Label(window,text = "Audio Language Detector and translator
",
                     width = 100, height = 4,font=('Arial Rounded MT Bold'
,15),
                     fg = "Blue")  #Defining a label for the project name
Display_label.pack() #used to execute the label function created
```

Now, we shall find our way to connect our graphical user interface application with our user defined functions and execute our program.

To do so, we shall be creating a button to trigger the user defined function that we declared in the previous section.

```
button = Button(window, text='Click Here To Start',bg="green", width=25,)
#creating a button
button.config(command=myTranslator) #defining what needs to be done when c
licked
button.pack() #Displaying the button in the Tkinter app
```

After we define, all the labels and button, we now want the GUI app to run to start our execution.

```
window.mainloop() # Execute our front end
```

Now, let's put all the pieces of our code in one place and execute our program:

```
import speech_recognition as spr
from googletrans import Translator
import nltk as nlp
```

```
from gtts import gTTS
import os
import sys

def __init__(self,translator):
    self.translator=Translator.translate()


def myTranslator(): #user defined functions
    recog1 = spr.Recognizer()
    mc = spr.Microphone() #setting microphone as our source for speaker
    to_lang = 'en'
    translator = Translator()
      #Caling the translator function from the package GoogleTrans
    with mc as source:
        print("Speak a stentence...")
        recog1.adjust_for_ambient_noise(source)
            #adjusting for ambient noise
        audio = recog1.listen(source,phrase_time_limit=5)
            #capturing the audio from the speaker. Here we have set the li
      mit as 5 secs. This can be changed as per our requirements
        get_sentence = recog1.recognize_google(audio)
            #using the recognize function to convert speech-to-text
    text_to_translate = translator.translate(text=get_sentence,dest='en')
      #Defining the Translate function
    print("Detected Language Code:",text_to_translate.src)
      # Printing the detected source language.
    print("Detected Text:"+ get_sentence)
      # Displaying the detected text from the speech input
    text = text_to_translate.text
      #Storing the Translated text to the variable Text, while calling the
 function to translate
    print("Translated Text:",text)
            # Printing the Translated Text
    speak = gTTS(text=text, lang=to_lang, slow= False) #converting text to
 speech and storing it in the varialble speak
```

```
    speak.save("captured_voice.mp3") # saving the converted speech into an
 .mp3 format for further usage.
    os.system("start captured_voice.mp3") #trigerring the Operating system
 to run the audio file from the default sound player

from tkinter import*
import tkinter as tk

window = Tk()
#defining a name for our GUI for refernce and calling the GUI function

window.title('GRIET Project') #defining a title of our GUI
window.geometry("450x150")  #defining a size for our GUI window
window.config(background = "white") #Selecting the background color

Display_label= Label(window,text = "Audio Language Detector and translator
",
                     width = 100, height = 4,font=('Arial Rounded MT Bold'
,15),
                     fg = "Blue")  #Defining a label for the project name
Display_label.pack() #used to execute the label function created

button = Button(window, text='Click Here To Start',bg="green", width=25,)
#creating a button
button.config(command=myTranslator) #defining what needs to be done when c
licked
button.pack() #Displaying the button in the Tkinter app

window.mainloop() # Execute our front end
```

## 6.2 Output

After completing our code, we need to execute our program to get our desired output. In our program, a tkinter application pops as our user interface.

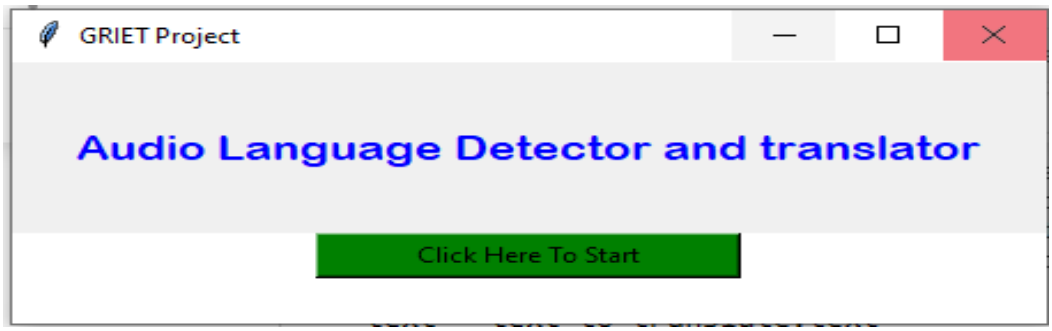


Fig 8: Front End Output

After we click on the Button ‘Click Here to start’, it will invoke our myTranslator() function and start the program.

After the function is triggered, the system microphone is triggered to record live audio.

Using the detected audio, the language of the speech is detected, detected text is displayed, along with the English Translated Text.

```
Speak a stentence...
Detected Language Code: hi
Detected Text:aaj mera din bahut busy tha
Translated Text: I had a very busy day today
```

Fig 9: Output Screen 1

```
Speak a stentence...
Detected Language Code: hi
Detected Text:mera naam Soumya hai
Translated Text: my name is saumya
Speak a stentence...
Detected Language Code: hi
Detected Text:mera naam kya hai
Translated Text: What is my name
Speak a stentence...
Detected Language Code: te
Detected Text:Naa Peru enti
Translated Text: What's my name?
```

Fig 10: Output Screen 2

As we can see here, different audio languages are automatically detected and then translated to the English language.

Also, notice that clicking on the button again will start the execution again without having to run the entire program again.

While the written text is shown in the console, the translated audio is played using the computer's audio player.

The output audio can also be replayed as many times as required.

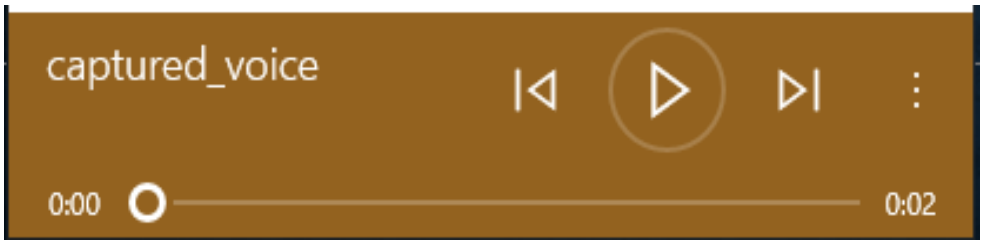


Fig 11: Output Audio

The program automatically reads our translated text without any further human inputs. However, if the user wants to listen to the sentence again, they can just click on the play button and play the translated audio again.

# 7. ADVANTAGES

- Users can use this application to easily translate the audio from their native language to English for better communication.
- Speech recognition allows the elderly and visually impaired users to communicate with products and services quickly.
- They can also use the application to learn the language as well, by using the replay translated audio feature.
- A customized GUI allows the users to easily understand what needs to be done next.
- Live speech input is enabled. Users can direct speak into the microphone anything that they wish to speak and get the translation for the same.
- A variety of language support makes it useful for wide range of users.
- Different languages of the speech input are automatically recognized Users do not have to manually select the source language.
- Fast and efficient.

# 8. CONCLUSION

A Speech Recognition program is created that converts the live speech input given into text format, allowing the machine to process the human input. Using the text produced, our program automatically detected the language of the source input and then converts it into English. After that, the translated text is again, converted into speech. Our finally produced translated speech audio is then played through the computer speakers.

This is a complete end – to – end application which does not require any human intervention throughout the process.

# 9. FUTURE ENHANCEMENTS

- More features can be added to improve the quality of the audio processing.
- More customized Graphical User Interface can be created.
- More languages could be added for translation.
- A comparison of different available speech recognizers and packages can be portrayed.
- This project can further be extended to speech Emotion detection.

# 10. BIBLIOGRAPHY


- "British English definition of voice recognition". *Macmillan Publishers Limited.* Archived *from the original on 16 September 2011*. Retrieved 21 February 2012 https://www.macmillandictionary.com/dictionary/british/voice-recognition
- *Juang, B. H.; Rabiner, Lawrence R.* "Automatic speech recognition–a brief history of the technology development"(PDF): 10. Archived from the original on 17 August 2014. https://web.archive.org/web/20140817193243/http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/354_LALI-ASRHistory-final-10-8.pdf
- *Benjamin, Martin (2019).* "Source data for Teach You Backwards: An In-Depth Study of Google Translate for 103 Languages". *Teach You Backwards*. Retrieved December 24,2019. https://docs.google.com/spreadsheets/d/1fJQLMj8O5z3Q7eKDxi1tNNrFipiEL0UDyaEF0fleZ54/edit?pli=1#gid=0
- *Henderson, Fergus* (November 5, 2010). "Giving a voice to more languages on Google Translate". Google Blog. Retrieved December 22, 2011.
- Supported voices and languages | Cloud Text-to-Speech Documentation: https://cloud.google.com/text-to-speech/docs/voices
- *Shipman, John W.* (2010-12-12), Tkinter reference: a GUI for Python, New Mexico Tech Computer Center, retrieved 2012-01-11
- *"Tkinter — Python interface to Tcl/Tk — Python v2.6.1 documentation"*. Retrieved 2009-03-12. https://docs.python.org/3/library/tkinter.html
- Vinyals, Oriol; et al. (2014). "Grammar as a Foreign Language" (PDF): https://proceedings.neurips.cc/paper/2015/file/277281aada22045c03945dcb2ca6f2ec-Paper.pdf
- https://link.springer.com/article/10.1007/s40012-013-0014-4
- https://www.simplilearn.com/why-python-is-essential-for-data-analysis-article
- http://www.ijetajournal.org/volume-4/issue-2/IJETA-V4I2P9.pdf
- *Dmitry L. Khabarov, Vadim V. Bazanov, Anna V. Kuchebo, Maksim Zavgorodnii, Anastasia Y. Rybakova.* "Research of Language Recognition Methods Based on Machine Learning" , 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), 2021.
- http://www.ijetajournal.org/volume-4/issue-2/IJETA-V4I2P9.pdf