**PROJECT REPORT**

**on**

**MULTILINGUAL SPEECH AND TEXT TRANSLATOR**

**(CSE IV Semester Mini project )**

**2023-2024**



**Submitted to: Submitted by:**

MS.PREETI CHODHARY Mr.KARAN PANWAR

(CC-CSE-G2-IV-Sem) Roll. No:2218923

CSE-G2-IV-Sem

Session: 2023-2024

**GRAPHIC ERA HILL UNVERSITY, DEHRADUN**

**CERTIFICATE**

Certified that Mr. KARAN PANWAR (Roll No.- 2218923) has developed mini project on ”“MULTILINGUAL SPEECH AND TEXT TRANSLATOR” for the CS Iv Semester Mini Project Lab in Graphic

Era Hill University, Dehradun. The project carried out by

Students is their own work as best of my knowledge.

Date: 12/07/2024

**MS. PREETI CHODHARY**

**Class Co-ordinator**

**CSE-G2-IV-Sem**

**(CSE Department)**

**GEHU Dehradun**

**ACKNOWLEDGMENT**

We would like to express our gratitude and appreciation to all those who gave me the possibility to complete this report.

We wish to thank our parents for their continuing support and encouragement. We also wish to thank them for providing us with the opportunity to reach this far in our studies.

We would like to thank particularly our project Co-ordinator

MS.PREETI CHODHARYfor her patience, support and encouragement throughout the completion of this project and having faith in us.

At last but not the least We greatly indebted to all other persons who directly or indirectly helped us during this work.

**Mr. KARAN PANWAR**

**Roll No.- 2218923**

**CSE-G2-IV-Sem**

**Session: 2023-2024**

**GEHU, Dehradun**

**Project Report: Text-to-Speech Conversion with Language Detection and Translation**

**Table of Contents**

1. **Introduction**
2. **Project Overview**
3. **Libraries Used**
   * gTTS
   * playsound
   * googletrans
4. **Implementation Details**
5. **User Interaction**
6. **Challenges and Solutions**
7. **Conclusion**
8. **Future Work**

**1. Introduction**

In today's multilingual world, text-to-speech (TTS) technology plays a crucial role in bridging communication gaps by converting written text into spoken words in various languages. This project demonstrates a simple yet powerful application of TTS, where text input by a user is detected, translated if necessary, and then converted into speech in a chosen language.

**2. Project Overview**

The objective of this project is to create a Python-based application that:

* Detects the language of the given text.
* Translates the text into a user-specified language.
* Converts the translated text into speech.
* Plays the speech output.

The application makes use of several Python libraries to achieve these functionalities, including gtts for text-to-speech conversion, playsound for playing the audio, and googletrans for language detection and translation.

**3. Libraries Used**

**gTTS (Google Text-to-Speech)**

The gtts library is a Python interface to Google Text-to-Speech API. It allows users to convert text to speech, supporting multiple languages and accents.

**Key Features:**

* Converts text to speech in various languages.
* Easy-to-use and integrates well with other libraries.
* Provides high-quality audio output.

**Installation:**

bash

Copy code

pip install gtts

**Example Usage:**

python

Copy code

import gtts

text = "Hello, how are you?"

tts = gtts.gTTS(text, lang='en')

tts.save("hello.mp3")

**playsound**

The playsound library is a simple module that provides a way to play sound files, with no dependencies other than standard libraries.

**Key Features:**

* Plays sound in any format supported by the underlying operating system.
* Minimalistic and easy to use.

**Installation:**

bash

Copy code

pip install playsound

**Example Usage:**

python

Copy code

import playsound

playsound.playsound("hello.mp3")

**googletrans**

The googletrans library is a Python client for Google Translate. It supports translation between multiple languages and can also detect the language of a given text.

**Key Features:**

* Provides translation between many languages.
* Can detect the language of input text.
* Simple API for integration into other projects.

**Installation:**

bash

Copy code

pip install googletrans==4.0.0-rc1

**Example Usage:**

python

Copy code

from googletrans import Translator

translator = Translator()

translated = translator.translate("Hello, how are you?", dest='es')

print(translated.text) # Output: Hola, ¿cómo estás?

**4. Implementation Details**

The project consists of several key functions:

**get\_language\_code**

This function prompts the user to select a language from a predefined list and returns the corresponding language code.

python

Copy code

def get\_language\_code():

print("Select the language for text-to-speech conversion:")

print("1. Hindi")

print("2. Bengali")

print("3. Tamil")

print("4. Telugu")

print("5. Kannada")

print("6. Gujarati")

print("7. Marathi")

print("8. Malayalam")

print("9. Punjabi")

while True:

choice = input("Enter the number corresponding to the language: ")

if choice == '1':

return 'hi'

elif choice == '2':

return 'bn'

elif choice == '3':

return 'ta'

elif choice == '4':

return 'te'

elif choice == '5':

return 'kn'

elif choice == '6':

return 'gu'

elif choice == '7':

return 'mr'

elif choice == '8':

return 'ml'

elif choice == '9':

return 'pa'

else:

print("Invalid choice. Please enter a number from 1 to 9.")

**detect\_language**

This function detects the language of the input text using googletrans.

python

Copy code

def detect\_language(text):

try:

translator = Translator()

lang = translator.detect(text).lang

return lang

except ValueError as e:

print(f"Error during language detection: {e}")

return None

except ConnectionError as e:

print(f"Error during language detection: {e}")

return None

except Exception as e:

print(f"Error during language detection: {e}")

return None

**main**

The main function coordinates the workflow: it takes user input, detects the language, translates the text if necessary, converts it to speech, and plays the audio.

python

Copy code

def main():

text = input("Enter the text you want to convert to speech: ")

desired\_lang\_code = get\_language\_code()

# Detect the language of the input text

input\_lang = detect\_language(text)

if input\_lang:

# Translate the text to the desired language if it's not already in that language

if input\_lang != desired\_lang\_code:

try:

translator = Translator()

translated\_text = translator.translate(text, dest=desired\_lang\_code).text

except Exception as e:

print(f"Translation error: {e}")

translated\_text = text

else:

translated\_text = text

try:

sound = gtts.gTTS(translated\_text, lang=desired\_lang\_code)

sound.save("output.mp3")

playsound.playsound("output.mp3")

except ValueError:

print("Error: Language code not recognized or supported.")

**5. User Interaction**

The application interacts with the user through the command line interface. Users input text, select their desired language, and the application provides audio output in the selected language. The user experience is straightforward and intuitive.

**6. Challenges and Solutions**

**Language Detection and Translation Errors**

* **Challenge:** Sometimes the googletrans library may fail to detect the language or translate the text due to API limits or connection issues.
* **Solution:** Implementing exception handling ensures that the program gracefully handles errors and provides informative messages to the user.

**Playing Audio Across Different Platforms**

* **Challenge:** Audio playback may vary across different operating systems due to varying support for audio formats and playback mechanisms.
* **Solution:** The playsound library was chosen for its cross-platform compatibility and ease of use.

**7. Conclusion**

This project successfully demonstrates how to use Python libraries to create a multilingual text-to-speech application. By leveraging gtts, playsound, and googletrans, the application can detect, translate, and convert text into speech in multiple languages, making it a versatile tool for users across different linguistic backgrounds.

**8. Future Work**

Future improvements to the project could include:

* Adding support for more languages.
* Implementing a graphical user interface (GUI) for a more user-friendly experience.
* Enhancing error handling and providing more detailed user feedback.
* Integrating additional TTS services for better voice quality and more language options.