BOL-2

(Text to speech & Translator)

Final Project

Group Members

1. Naveed Hayder
2. Muhammad Asad
3. Muhammad Farhan

Submitted to: Sir. Abdullah Arif

Python programming - Bano Qabil 2.0

CONTENTS

Abstract

The development of technology connects everyone from all around the worlds. The problem is, people cannot really mingle with one another because they have communication problems. Some of the problems are with other traveler, disabled peoples, Friends in social media, and International business partners. This device invented to solve this entire problem that faced by people in today's life.

This device invented to make people more knowledgeable, reduce miscommunication among people all around the world, connects people, get maximum profit and give job opportunity to people.

Translation is a medium to transfer the knowledge or information. It can be a bridge which connects the people from the different languages and cultures. By using translation, people can learn and understand each other's languages and cultures. Translation is not merely at changing words, but also transferring of cultural equivalence with the culture of the original language and the recipient of that language as well as possible. The better translation must be accepted by all people in logic and based on fact; thus, the message which contained in the source language (SL) can satisfy the target language (TL) reader with the information within.

Translation is necessary for the spreading new information, knowledge, and ideas across the world. It is absolutely necessary to achieve effective communication between different cultures. In the process of spreading new information, translation is something that can change history.

About this project

Text to speech is a process to convert any text into voice. Text to speech project takes words on digital devices and convert them into audio with a button click. Text to speech python project is very helpful for people who are struggling with reading.

Translation is necessary for the spreading new information, knowledge, and ideas across the world. It is absolutely necessary to achieve effective communication between different cultures. In the process of spreading new information, translation is something that can change history. So we have added translation model into it.

The Python Text-to-Speech project is a versatile and engaging application that leverages the power of text-to-speech synthesis to convert written text into spoken words. This project incorporates the capabilities of Python programming language and text-to-speech libraries to create a user-friendly tool for various applications.

The Python Text-to-Speech project combines the power of programming with real-world applications, offering users a dynamic tool for converting text into natural and customizable speech.

Objectives

• To extract effective communication between people around the world.

• To provide ability for two parties to communicate and exchange the ideas.

• To encourage learners to discuss the meaning and use of language at the deepest possible levels.

• To get a challenging position in reputed organization where we can learn a skills by communicating.

• To perform and translate our native language.

Proposed System

• The aim of the proposed system is to develop a system that has capability to perform Translation & Converting text to speech. The system proposed here will be developed for a small domain of English words.

• A translator is a programming language processor that modifies a computer program from one language to another.

Key Features

User-Friendly Interface: The project offers an intuitive and easy-to-use interface, allowing users to input text and convert it into speech with a simple click or command.

Speech Customization: Users can customize the speech output by adjusting parameters such as pitch, speed, and voice selection. This feature enhances the naturalness and personalization of the synthesized speech.

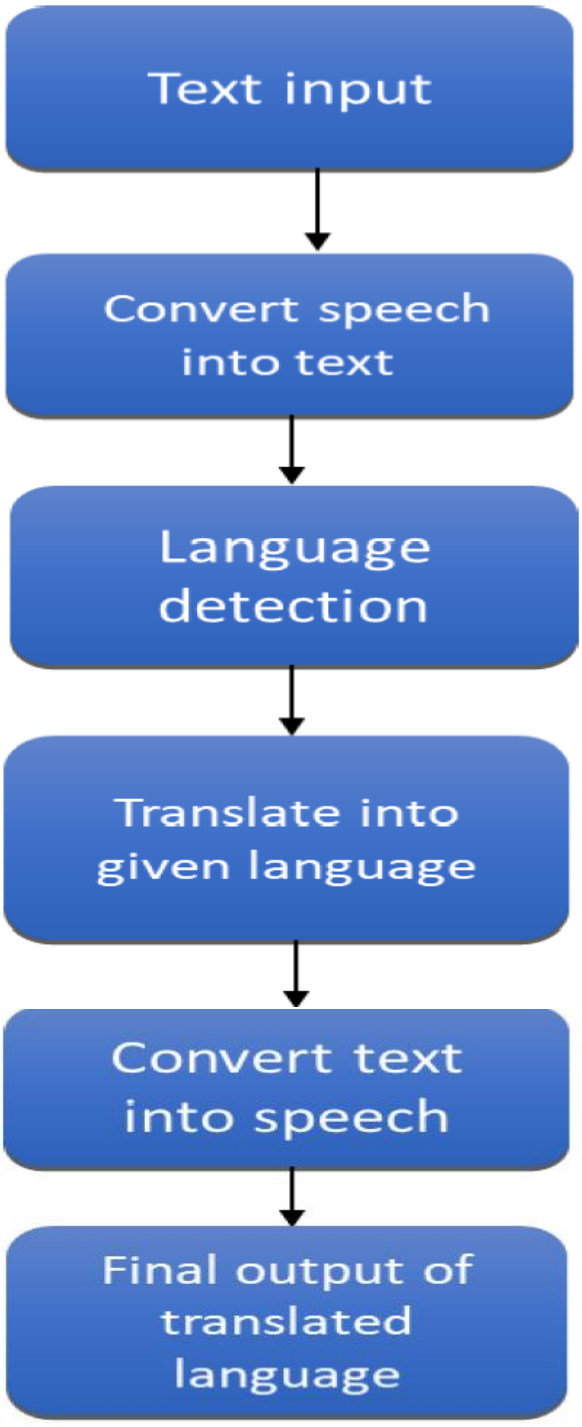
Multi-Language Support: The application supports multiple languages, enabling users to convert text into speech in their preferred language. This feature broadens the project's accessibility and usability.

File Input/Output: Users can input text from various sources, including direct input, text files, or other data formats. Additionally, the project allows users to save the synthesized speech as audio files for later use.

Educational Applications: The text-to-speech project can find applications in the education sector, facilitating the creation of educational resources, audiobooks, and language learning materials.

Versatility across Platforms: As a Python-based project, it can run seamlessly on various platforms, including Windows, macOS, and Linux, ensuring broad compatibility and accessibility.

BLOCK DIAGRAM



Buttons

When we click on the Translate button it will call the translate function Button and same is for speech button () widget used to display button on our window command is called when we click the button active background sets the background color to use when the button is active root.mainloop().

TECHNOLOGY:

- We Important Libraries and python modules.
- Python frontend
- API calls
- Speech Recognition module.
- We Use VS Code Software as code editor

MODULES:

USER MODULE

- Text/speak: that he/she want to translate.
- Language change: According to understanding they can change language of the translation.

Program screenshots:

Libraries:

```
import tkinter as tk
from tkinter import ttk
from googletrans import Translator
import gtts
import playsound
import os
import uuid
```

Coding:

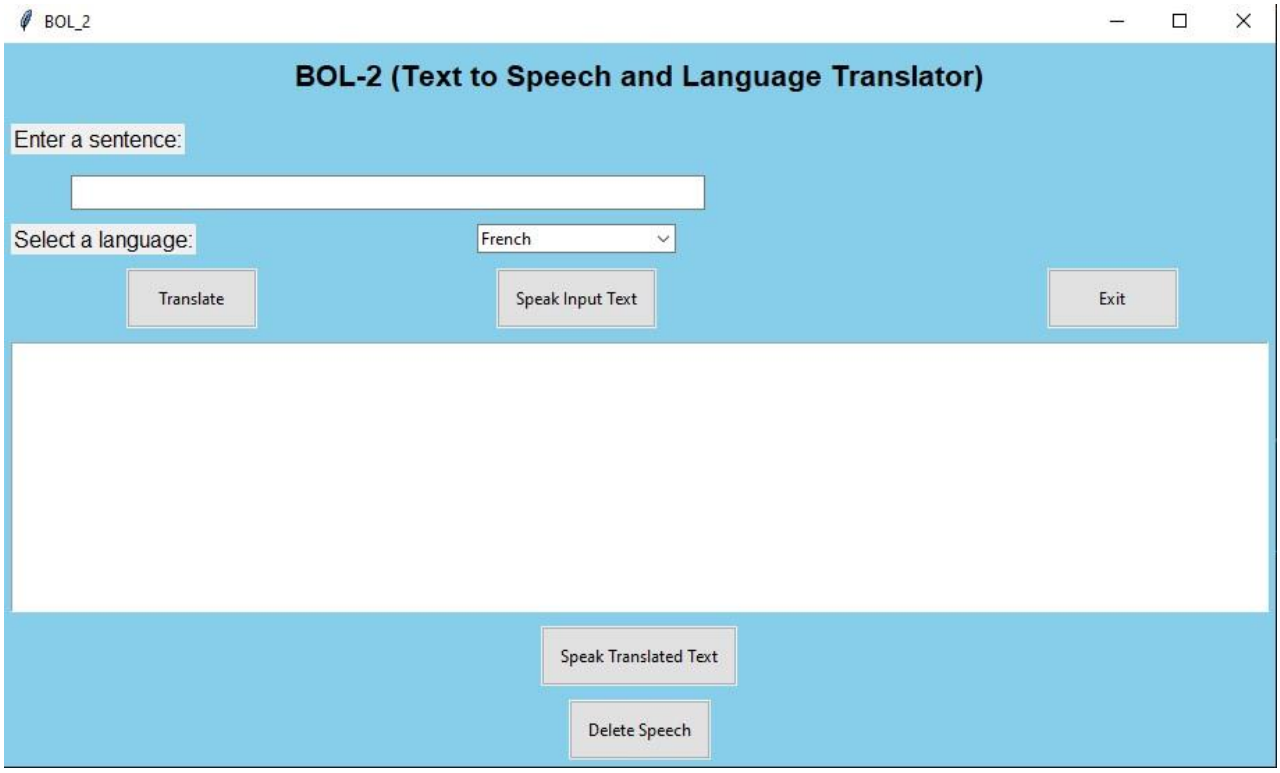
```
class TranslatorApp:
    def __init__(self, master):
        self.master = master
        master.title("BOL_2")

        # Set background color to sky green
        master.configure(background="#87CEEB")

        # Set up the heading label
        self.heading_label = ttk.Label(master, text="BOL-2 (Text to Speech and Language Translator)", font=("Helvetica", 16, "bold"), background="#87CEEB")
        self.heading_label.grid(row=0, column=0, columnspan=4, pady=10, padx=10)

        # Set up the label for entering a sentence
        self.label = ttk.Label(master, text="Enter a sentence:", font=("Helvetica", 12))
        self.label.grid(row=1, column=0, columnspan=2, pady=10, padx=5, sticky="w")

        # Entry widget for user input
        self.entry = ttk.Entry(master, font=("Helvetica", 13), width=50)
        self.entry.grid(row=2, column=0, columnspan=2, pady=5, padx=5)

        # Label for selecting language
        self.language_label = ttk.Label(master, text="Select a language:", font=("Helvetica", 12))
        self.language_label.grid(row=3, column=0, sticky="w", pady=5, padx=5)

        # Combobox for selecting language
        self.language_var = tk.StringVar()
        self.language_combobox = ttk.Combobox(master, textvariable=self.language_var, state="readonly", width=20)
        self.language_combobox['values'] = ["French", "Arabic", "Korean", "Hindi", "Urdu", "Chinese", "German", "Italian", "Japanese", "Russian", "Spanish", "Portuguese"]
        self.language_combobox.current(0)
        self.language_combobox.grid(row=3, column=1, pady=5, padx=5)

        # Button for translating text
        self.translate_button = ttk.Button(master, text="Translate", command=self.translate_text, style="Custom.TButton")
        self.translate_button.grid(row=4, column=0, pady=5, padx=5)

        # Button for speaking input text
        self.speak_button_input = ttk.Button(master, text="Speak Input Text", command=self.speak_input_text, style="Custom.TButton")
        self.speak_button_input.grid(row=4, column=1, pady=5, padx=5)
```

```
        # Button for exiting the application
        self.exit_button = ttk.Button(master, text="Exit", command=self.master.quit, style="Custom.TButton")
        self.exit_button.grid(row=4, column=3, pady=5, padx=5)

        # Text widget for displaying output
        self.output_text = tk.Text(master, wrap=tk.WORD, font=("Helvetica", 13), height=10, width=100)
        self.output_text.grid(row=5, column=0, columnspan=4, pady=5, padx=5, rowspan=2)

        # Button for speaking translated text
        self.speak_button_translated = ttk.Button(master, text="Speak Translated Text", command=self.speak_translated_text, style="Custom.TButton")
        self.speak_button_translated.grid(row=7, column=0, columnspan=4, pady=5, padx=5)

        # Button for deleting speech files
        self.delete_button = ttk.Button(master, text="Delete Speech", command=self.delete_mp3_files, style="Custom.TButton")
        self.delete_button.grid(row=8, column=0, columnspan=4, pady=5, padx=5)

        # List to store audio file paths
        self.audio_files = []

        # Custom style for the buttons to add padding
        s = ttk.Style()
        s.configure("Custom.TButton", padding=(10, 10))

    # Translate text entered by the user
    def translate_text(self):
        text = self.entry.get()
        dest_lang = self.get_dest_lang()
        translator = Translator()
        translated_text = translator.translate(text, dest=dest_lang).text
        self.output_text.insert(tk.END, f"{translated_text}\n")
```

```
    # Speak the input text entered by the user
    def speak_input_text(self):
        text = self.entry.get()
        lang = self.get_dest_lang()
        tts = gtts.gTTS(text, lang=lang)
        filename = f"input_{uuid.uuid4().hex}.mp3"
        tts.save(filename)
        playsound.playsound(filename)
        self.audio_files.append(filename)

    # Speak the translated text
    def speak_translated_text(self):
        translated_text = self.output_text.get("end - 2 lines linestart", "end")
        lang = self.get_dest_lang()
        tts = gtts.gTTS(translated_text, lang=lang)
        filename = f"translated_{uuid.uuid4().hex}.mp3"
        tts.save(filename)
        playsound.playsound(filename)
        self.audio_files.append(filename)

    # Delete speech files
    def delete_mp3_files(self):
        for filename in self.audio_files:
            if os.path.exists(filename):
                os.remove(filename)
                self.output_text.insert(tk.END, f"{filename} deleted\n")
        self.audio_files.clear()
```

```
    # Get the language code corresponding to the selected language
    def get_dest_lang(self):
        languages = {
            "French": "fr",
            "Arabic": "ar",
            "Korean": "ko",
            "Hindi": "hi",
            "Urdu": "ur",
            "Chinese": "zh-CN",
            "German": "de",
            "Italian": "it",
            "Japanese": "ja",
            "Russian": "ru",
            "Spanish": "es",
            "Portuguese": "pt"
        }
        return languages.get(self.language_combobox.get())

def main():
    root = tk.Tk()
    app = TranslatorApp(root)
    root.mainloop()

if __name__ == "__main__":
    main()
```

OUTPUT:



RESULT

We develop this application for desktop application. Here we are integrating the speech to speech, text to text, speech to text and language translator in one system so user doesn't have to download for the different application. You can also give voice input to translate language.

CONCLUSION

In this proposed system, we implemented the system for user who phasing problems of language barrier and also it user interface is also user friendly so that user can easily interact with this system so it automatically reduce the user task for understanding the languages for communication.

Translation is not merely at changing words, but also transferring of cultural equivalence with the culture of the original language and the recipient of that language as well as

possible. The better translation must be accepted by all people in logic and based on fact; thus, the message which contained in the source language (SL) can satisfy the target language (TL) reader with the information within.

When you understand the importance of translation for everyone, you will be able to see it as a necessary and worthy investment.

----------xxxxxxxxxxxxxxxxxxxxxxxxxxx----------