**“Simultaneous English to French and Hindi Translation for 10-Letter Words”**

**Internship Report**

**Code**

import tkinter as tk

from tkinter import messagebox

import speech\_recognition as sr

from googletrans import Translator

import datetime

# Function to check if the current time is after 6 PM IST

def is\_after\_6pm():

current\_time = datetime.datetime.now().time()

return current\_time >= datetime.time(18, 0)

# Function to capture and translate speech

def translate\_audio():

if not is\_after\_6pm():

messagebox.showwarning("Access Denied", "Audio translation is only available after 6 PM IST.")

return

recognizer = sr.Recognizer()

with sr.Microphone() as source:

status\_label.config(text="Listening...", fg="#4CAF50") # Green

root.update()

try:

audio = recognizer.listen(source, timeout=5)

status\_label.config(text="Processing...", fg="#FFC107") # Amber

root.update()

spoken\_text = recognizer.recognize\_google(audio, language="en-IN")

translated\_text = translator.translate(spoken\_text, dest='hi').text

output\_text.config(state=tk.NORMAL) # Enable text widget for editing

output\_text.delete(1.0, tk.END)

output\_text.insert(tk.END, translated\_text)

output\_text.config(state=tk.DISABLED) # Disable text widget for editing

status\_label.config(text="Translation complete", fg="#2196F3") # Blue

except sr.UnknownValueError:

messagebox.showerror("Error", "Could not understand the audio. Please try again.")

except sr.RequestError:

messagebox.showerror("Error", "Could not request results; check your network connection.")

except Exception as e:

messagebox.showerror("Error", f"An error occurred: {str(e)}")

# Initialize the main window

root = tk.Tk()

root.geometry("700x500")

root.resizable(0, 0)

root.config(bg='#2C3E50') # Dark gray

root.title('Audio Translator to Hindi')

translator = Translator()

# Title Label

title\_label = tk.Label(root, text="Audio Translator", font="Helvetica 24 bold", bg='#3498DB', fg="white")

title\_label.pack(pady=20, fill=tk.X)

# Status Label

status\_label = tk.Label(root, text="Click 'Translate' to start", font="Helvetica 16", bg='#2C3E50', fg="white")

status\_label.pack(pady=10)

# Output Text Box

output\_text = tk.Text(root, font='Helvetica 16', height=8, wrap=tk.WORD, padx=10, pady=10, bg='#ECF0F1', fg='#2C3E50')

output\_text.pack(pady=10, fill=tk.BOTH, expand=True)

output\_text.config(state=tk.DISABLED) # Initially disabled

# Translate Button

translate\_btn = tk.Button(root, text='Translate', font='Helvetica 16 bold', pady=10, command=translate\_audio,

bg='#E74C3C', fg='white')

translate\_btn.pack(pady=10)

# Main loop

root.mainloop()

**Introduction**

This report presents the development of an audio translation application as part of my internship . The project aimed to translate spoken English words into Hindi, incorporating a feature that restricts translation to after 6 PM IST. This innovative application leverages Python's Tkinter for the graphical user interface, `speech\_recognition` for audio processing, and `googletrans` for translation.

**Background**

The application was developed to address the need for real-time translation services that are both user-friendly and interactive. It was created to showcase the integration of multiple technologies, including natural language processing and GUI design. The decision to restrict translation functionality to after 6 PM IST was intended to add a unique feature, demonstrating the capability to incorporate time-based logic into software applications.

**Learning Objectives**

**GUI Development**: To acquire skills in creating interactive user interfaces using Tkinter.

**Speech Recognition**: To understand and implement speech-to-text technology using the `speech\_recognition` library.

**Translation Services**: To learn how to integrate translation APIs, specifically `googletrans`, into a Python application.

**Time-based Logic**: To apply conditional programming based on system time to control application functionality.

**Activities and Tasks**

**Design and Implementation**: Created a user interface with Tkinter, focusing on usability and visual appeal. Designed layout elements including labels, buttons, and text boxes.

**Speech Recognition Integration**: Configured the `speech\_recognition` library to capture and process spoken audio. Implemented error handling for various potential issues such as network failures and unrecognized speech.

**Translation Feature**: Incorporated the `googletrans` API to translate recognized speech from English to Hindi. Ensured the application provides accurate and real-time translations.

**Time-based Restriction**: Developed and integrated a time-checking mechanism to enforce the restriction on translation functionality based on the current time.

**Skills and Competencies**

**Technical Skills**: Advanced Python programming skills, proficiency with Tkinter for GUI development, and experience with `speech\_recognition` and `googletrans` libraries.

**Problem-solving:** Developed solutions to handle various challenges, including audio recognition issues and time-based functionality.

**Project Management**: Managed project timelines effectively, balancing development tasks with testing and debugging.

**User Experience:** Focused on creating an intuitive and responsive user interface, enhancing the overall user experience.

**Feedback and Evidence**

**User Feedback**: Initial feedback from users highlighted the application's ease of use and the unique feature of time-based access. Users appreciated the clear interface and the effectiveness of the translation.

**Evidence**: Attached are screenshots of the application in use, demonstrating the translation process and user interface. A link to the source code repository is also provided for review.

**Challenges and Solutions**

**Challenge**: Ensuring accurate time-based functionality in the application.

**Solution**: Utilized Python’s `datetime` module to manage time-based conditions accurately, and tested the functionality across different time zones.

**Challenge**: Handling varying audio quality and recognizing diverse accents.

**Solution:** Implemented comprehensive error handling and feedback mechanisms to guide users and improve the robustness of speech recognition.

**Outcomes and Impact**

**Outcome**: Delivered a functional and reliable audio translation application with a unique time-based feature. The project demonstrated the ability to integrate multiple technologies into a cohesive solution.

**Impact**: The project provided valuable experience in developing real-world applications, enhancing my skills in Python programming and software development. The knowledge gained will be beneficial for future projects and career development.

**Conclusion**

The internship project was a significant learning experience, allowing me to apply theoretical knowledge in a practical context. The development of the audio translation application showcased my ability to create functional software solutions and integrate various technologies effectively. This project has prepared me for future challenges in software development and has contributed to my growth as a software engineer.