# **Language Translation App Documentation**

## **1. Introduction**

An application which can be used to translate text in the images to desired language and output the text as a speech.

## **2. Use Case**

Use the OpenAI API to translate text from one language to another, given an image as input. The AI should first recognize the text in the image and identify its language. Then, it should translate the text to another language and return the translation as audio.

Requirements:

1. The AI must be able to recognize text in images accurately, even if the text is handwritten, blurry, or distorted.
2. The AI must be able to identify the language of the text in the image with a high degree of accuracy.
3. UI for this ask is not necessary, however please feel to enhance your solution if you face any issue while passing other language text/script as input from terminal.

## **3. Components**

The app consists of several key components:

* **Image Text Extraction**: Utilizes Google Cloud Vision API to extract text from images.
* **Language Translation**: Uses Azure Translator Text API to translate the extracted text.
* **Text-to-Speech Conversion**: Employs Azure Speech SDK to convert translated text into speech.

## **4. How the App Works**

1. Upload Image: Users upload an image containing text through the app's interface, and selects the language in which he wants to translate from the dropdown menu.
2. Extract Text: The app utilizes Google Vision AI services to extract text from the uploaded image.
3. Translation: The extracted text is then translated into the user-specified language using the Azure Translator Text API.
4. Text-to-Speech: If the selected language is supported for text-to-speech, the translated text is converted into speech using the Azure Speech SDK.

## **5. Setup and Configuration**

Before running the app, make sure to set up the following:

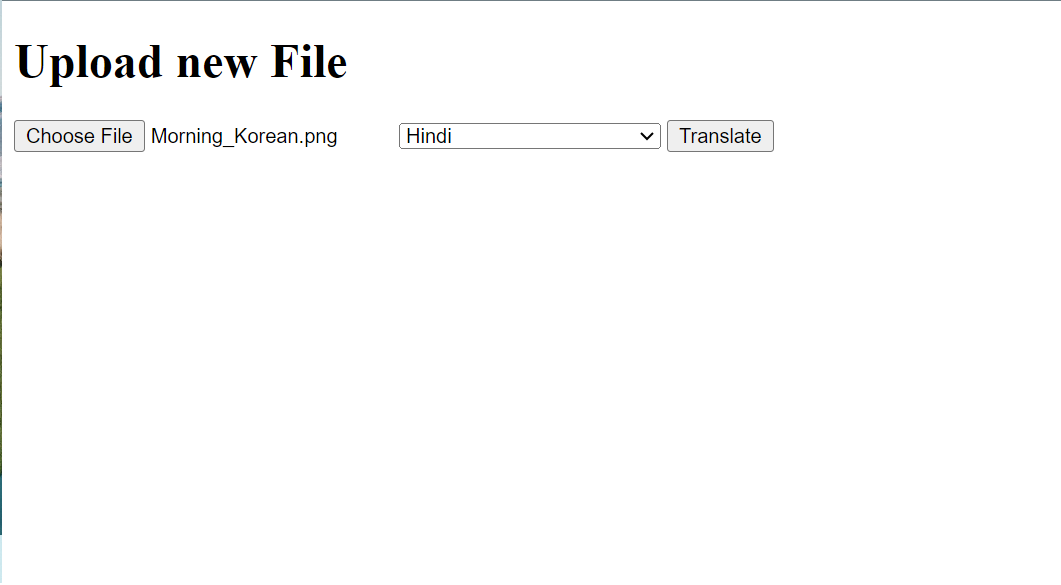
* Install python, flask.
* Set up Google Vision AI services (OCR) and Azure services (Translator Text API and Speech service).
* Configure environment variables for Google & Azure subscription keys.

## **6. Running the App**

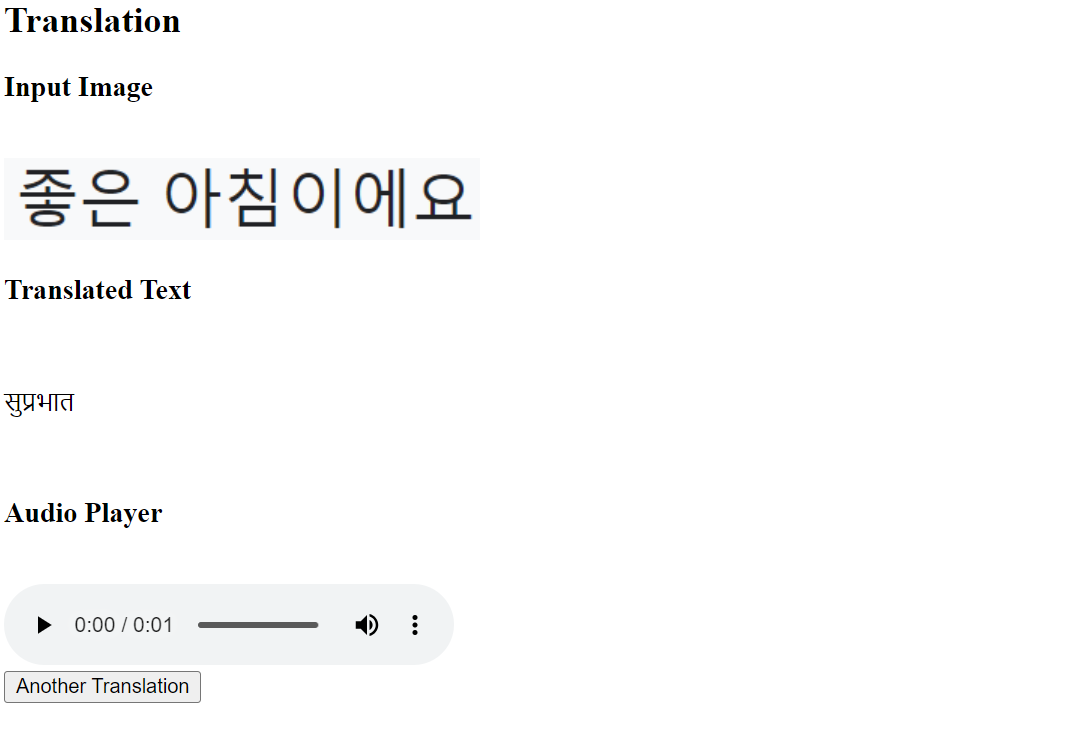
1. Clone the repository and navigate to the project directory.
2. Follow setup and configuration to install dependencies.
3. Set up GCP services & Azure services and obtain subscription keys.
4. Configure environment variables for Azure keys.
5. Run the Flask app using `python app.py`.

## **7. Screenshots and Working**

1. **UI for user to upload image and language in which he wants to translate.**

****

1. **UI for user to get the translated text and audio prompt.**



## **8. Supported Languages**

The app supports a wide range of languages, including but not limited to:

* English
* Korean
* French
* German
* Chinese (Simplified)
* Japanese
* Russian
* Hindi

Check the app interface for the complete list of supported languages.

## **9.** **Learnings:**

1. Open AI API: The Open AI does not provide OCR services for text extraction from images. However, it has a model called CLIP, which learns concept from natural language supervision. It is not specifically designed for OCR, rather it can be used to describe the images. In essence, the Open AI does not provide OCR services.
2. Microsoft Azure Cognitive Services: Microsoft do provide the services for text extraction for different languages (around 150). It was even able to work effectively for Korean, Japanese, English, Chinese languages and able to extract the text from the images but was not able to extract text for Hindi, Telugu, and other regional languages. Also, the OCR model was not working for blurry images.
3. Google Cloud Vision API: The OCR model worked well with all the regional languages and even with the blurry images and was able to extract text from them.
4. Azure Translator Text: supports around 150 languages which it can translate.
5. Azure Speech SDK: supports multiple languages for speech conversion and saving the audio as a wav file.

## **10. Test Cases:**

1. The code works well for blurry images and is also even to generate meaningful translations.
2. Tested mainly on languages such as Hindi, English, Chinese, French, Korean, Japanese, Telugu.
3. For long text it might be just translate but might not follow grammar of the language.
4. Works good with numbers and handwritten images as well.
5. For milestone and signboards, it sometimes reads the unnecessary information and even speaks that too.
6. Works with noisy images also.