# Translation App Report

By

Muhammad Ahmed Zaheer

Nao Medical

Pre Interview Junior Software Engineer

November, 2024.

# Table of Contents

# List of Figures

# Abstract

This report documents the development and limitations of a prototype titled *Healthcare Translation Web App with Generative AI*. The objective was to create a web-based application capable of real-time multilingual translation between healthcare providers and patients, featuring voice-to-text conversion, live transcription, translation, and audio playback. Despite challenges in deployment and real-time interactivity, key components such as speech recognition, translation, and text-to-speech synthesis were implemented successfully using Google Cloud APIs. This report discusses the project workflow, features, limitations, and future directions to expand the prototype into a functional application.

# 1 Introduction

The *Healthcare Translation Web App with Generative AI* is designed to address the critical communication barriers in multilingual healthcare settings. The project aimed to develop a web-based prototype that enables real-time, multilingual translation between patients and healthcare providers. By integrating cutting-edge generative AI tools and APIs, the application supports the following core functionalities:

1. **Voice-to-Text Conversion**: Using AI-enhanced speech recognition to transcribe spoken input, ensuring accuracy for medical terminology.
2. **Real-Time Translation**: Providing seamless translation of transcripts into the desired language.
3. **Audio Playback**: Enabling playback of the translated text in audio form.
4. **Dual Transcript Display**: Offering live, side-by-side display of both the original and translated transcripts.

The project was intended to assess technical proficiency, development speed, and the innovative use of generative AI for efficient application development. Although the deployment stage posed challenges, the prototype successfully demonstrates key processing capabilities locally, setting a solid foundation for future enhancements.

# 2 Accomplishments

The development of the *Healthcare Translation Web App with Generative AI* saw significant milestones achieved despite challenges with deployment. Below is an overview of what was accomplished:

1. **Audio Processing and Transcription**
   - Accurate speech-to-text conversion for audio files in French and Hindi was achieved using Google Cloud Speech-to-Text. This process was fine-tuned to accommodate different accents and medical terminology.
2. **Translation and Dual Transcripts**
   - Text translation was performed seamlessly using Google Cloud's Translation API.

- Generated transcripts were saved locally as .txt files for both the original and translated languages, enhancing usability and accessibility.

3. **Audio Playback Generation**

- Google Cloud Text-to-Speech API was used to generate audio files from the translated text, saved in .mp3 format.
- Silence intervals were programmatically added between sentences for clarity during playback.

4. **Integration with Cloud Storage**

- A Google Cloud Storage bucket was configured and used to store audio files, enabling streamlined access for processing.

# 3 Limitations

While significant progress was made, several limitations impacted the overall development of the *Healthcare Translation Web App with Generative AI*. These constraints can be categorized as technical, logistical, and resource-related:

**1. Deployment Challenges**

Deployment faced significant obstacles on various platforms:

- **Vercel**: Presented a persistent 250 MB memory limit issue, even after attempts to modularize dependencies and optimize file sizes.
- **Cursor**: Incompatibility with Python 3.12 hindered its use, as the required CLI commands and configurations were not functional in this environment. Attempts to install cursor cli were made using pip and pip3 libraries, though Node.js and even after installing Docker but to no avail.
- **V0**: Due to time limitations, V0 was not explored as a viable deployment option.

**2. Time Constraints**

- The 48-hour project timeframe presented a significant challenge, particularly for troubleshooting deployment issues and fine-tuning advanced features.
- Certain optimizations and refinements, such as UI/UX enhancements or more robust error handling, could not be fully addressed within the allotted time.

**3. Limited Access to Premium AI Tools**

- While generative AI coding assistants were recommended for the project, access was limited to free-tier models such as GPT-3.5.
- The absence of the superior GPT-4 model restricted the level of precision and support available for handling complex coding scenarios and optimizing the workflow.

**4. Resource Constraints**

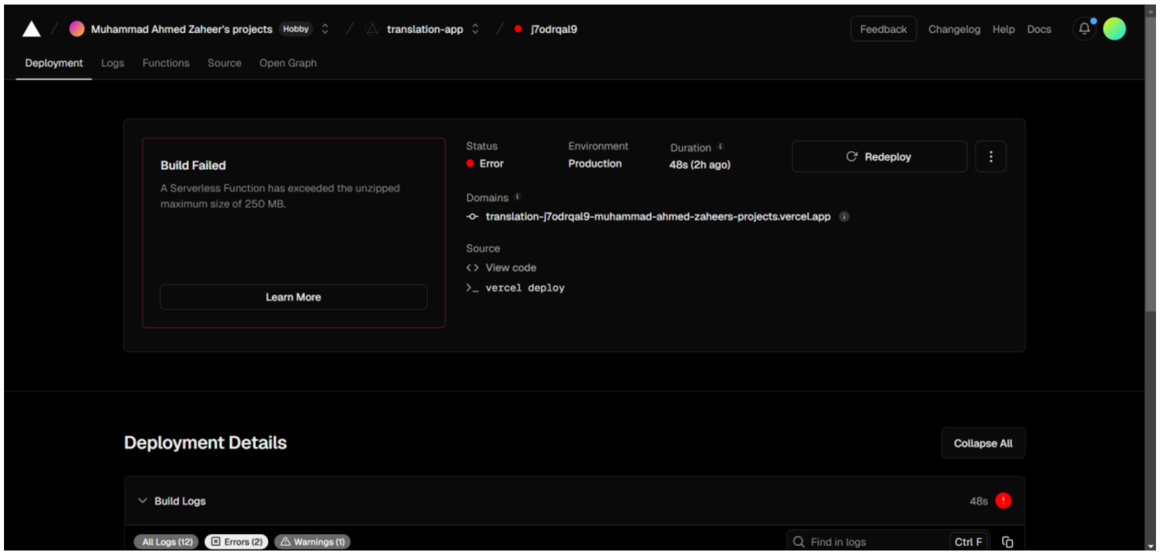
- Development heavily relied on free trial services, including Google Cloud Platform, to access speech recognition, translation, and text-to-speech APIs.
- The reliance on free tiers imposed limitations on API quotas, affecting the ability to scale and test larger datasets or longer audio files comprehensively.

**5. Lack of Real-Time Interactivity**

- The prototype processes pre-recorded audio files rather than live, interactive voice input and output.
- Implementing real-time microphone inputs and live transcription was considered, but the added complexity exceeded the project's scope within the limited timeframe.

**6. Security and Privacy Considerations**

- While the project utilized Google Cloud for storage and processing, more robust measures to ensure end-to-end encryption and patient confidentiality could not be implemented within the timeframe.



Build Failed

A Serverless Function has exceeded the unzipped maximum size of 250 MB.

Status: Error

Environment: Production

Duration: 48s (2h ago)

Domains: translation-j7odrqal9-muhammad-ahmed-zaheers-projects.vercel.app

Source: View code

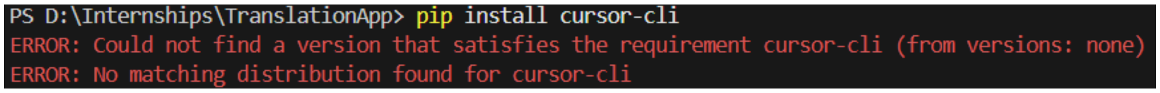
```
vercel deploy
```

Deployment Details

Build Logs 48s

All Logs (12) Errors (2) Warnings (1)

*Figure 1: Project could not be deployed on Vercel as the size limit kept exceeding despite many attempts to solve this*



```
PS D:\Internships\TranslationApp> pip install cursor-cli
ERROR: Could not find a version that satisfies the requirement cursor-cli (from versions: none)
ERROR: No matching distribution found for cursor-cli
```

*Figure 2: One of the many attempts to install cursor cli*

# 4 Workflow and Features

This section outlines the workflow of the *Healthcare Translation Web App with Generative AI* and its key features, highlighting the technical implementation and intended user experience.

## Workflow

1. **Audio Input Processing**
   - The prototype processes audio files stored in a Google Cloud Storage (GCS) bucket.
   - **Reason for GCS Usage**: Since the audio files were longer than a minute, direct uploads to Colab failed due to size and processing limitations. GCS provided a robust solution for hosting and accessing large audio files.
   - The application retrieves audio files from the designated GCS bucket using their URI.
2. **Speech-to-Text Conversion**
   - Google Cloud Speech-to-Text API was employed to transcribe spoken input into text.
   - The transcription process supports multiple languages, and the project successfully tested with both French and Hindi samples.
3. **Text Translation**
   - The transcribed text is translated into the desired target language using Google Cloud Translation API.
   - The application prioritizes accuracy, especially for medical terms, leveraging AI's contextual understanding capabilities.
4. **Audio Synthesis (Text-to-Speech)**
   - The translated text is converted into spoken audio using Google Cloud Text-to-Speech API.
   - The output audio file is stored locally in MP3 format, allowing playback of the translated content.

5. **Result Output**
   - The prototype generates the following outputs:
     - Original language transcript (saved as a .txt file).
     - Translated language transcript (saved as a .txt file).
     - Audio playback of the translated transcript.

## Features

1. **Core Functionalities**
   - **Voice-to-Text**: Processes audio files and generates a transcription in the original language.
   - **Translation**: Converts the transcription into the desired language, ensuring grammatical correctness and contextual relevance.
   - **Audio Playback**: Provides a translated audio output, simulating real-time language interpretation.
2. **Dual Transcript Output**
   - Transcripts in both the original and translated languages are saved and presented as separate .txt files for easy reference.
3. **Google Cloud Integration**
   - Seamless integration with Google Cloud APIs for Speech-to-Text, Translation, and Text-to-Speech functionalities.
   - Efficient and accurate processing of multilingual audio samples.
4. **Scalable Design**
   - The modular structure of the code allows for scaling to include additional languages or complex workflows in the future.
5. **Error Handling**
   - Basic error handling to manage transcription or translation failures, providing informative logs for debugging.
6. **Responsive Output Formats**
   - The generated text and audio files are compatible across multiple devices and platforms, enhancing accessibility and usability.

# 5 Future Work

While the prototype achieved several key objectives, there is substantial room for improvement and expansion to meet the requirements of a fully functional, real-time multilingual healthcare translation app. Below are the proposed future enhancements:

**1. Real-Time Interactivity**

- **Microphone Integration**: Enable the app to accept direct audio input from a user's microphone. This would significantly enhance interactivity and real-time usage.
    - *Challenges*: Introducing this feature requires managing real-time audio streaming and processing, which adds complexity and goes beyond the current prototype's scope.
- **Live Display of Dual Transcripts**: Implement live transcription and translation displays on the user interface for seamless user experience.

**2. Improved Deployment**

- **Scalable Deployment**: Successfully deploy the app on platforms like Vercel, Cursor, or V0 without hitting memory or compatibility limits.
    - Addressing the **Vercel Issue**: Explore solutions for the 250 MB memory limit that hindered deployment. This could include optimizing dependencies or switching to a backend-based deployment model.
    - Overcoming **Cursor CLI Challenges**: Ensure compatibility with Python 3.12+ or consider containerized environments for CLI operations.
    - Explore V0 as a fallback option if time constraints permit its setup.

**3. Enhanced User Experience**

- **Customizable Language Options**: Add a dropdown menu for users to select source and target languages dynamically instead of hardcoding them.
- **User Interface Enhancements**:
    - Mobile-first design optimization.
    - Adding visual feedback for processes like transcription and translation status.

**4. Model Optimization**

- **Custom Speech-to-Text Model**: Train domain-specific models for better recognition of medical terminology and dialect-specific nuances.
- **Advanced Translation**: Integrate cutting-edge generative AI models (e.g., GPT-4 or Llama 2) for more context-aware and accurate translations.

**5. Security and Privacy**

- **Data Privacy Compliance**: Implement end-to-end encryption for data in transit and at rest to ensure patient confidentiality.
- **Secure Authentication**: Introduce user authentication and access controls for enhanced security.

**6. Extended Functionalities**

- **Sentence-Level Playback**: Allow playback of specific sentences for improved usability in medical settings.
- **Pause/Resume Options**: Add pause and resume functionalities for live audio interactions.
- **Offline Mode**: Enable basic functionalities offline by integrating pre-trained language models and voice processing libraries locally.

**7. Expanded Testing and Error Handling**

- **Robust Error Logging**: Enhance error logs to provide detailed insights for debugging during transcription, translation, or audio synthesis processes.
- **Extensive Quality Assurance**: Conduct thorough testing with diverse audio samples to improve performance across various accents, languages, and medical terms.

These enhancements aim to bridge the gap between the prototype and a fully realized application, aligning with the original vision of a real-time multilingual translation tool for healthcare professionals and patients.