



iJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: V Month of publication: May 2025

DOI: <https://doi.org/10.22214/ijraset.2025.70887>

www.ijraset.com

Call:  08813907089

E-mail ID: ijraset@gmail.com

Language Translator from English to Hindi (Official Language)

K Bhargav Teja¹, K Snehith Reddy², A Vignesh Raju³, MS Yeshwanth⁴, Mr. Shankar J⁵

^{1, 2, 3, 4}Student, CST Dev Ops, ⁵Assistant Professor, School of CSE & IS, Presidency University (of Aff.) Bengaluru, India.

Abstract: In a multilingually rich country such as India, communication across language speakers should be effective to ensure public services are delivered easily through digital sources. This project offers a user-friendly, secure, and scalable language translator device specifically for installation on government organizational websites to auto-translate English to Hindi, being the official Indian government language.

The system takes advantage of current web technologies—HTML5, CSS3 (Bootstrap 5 with Neumorphism style), and JavaScript—to achieve a responsive and accessible frontend. On the backend, it uses Python (Flask framework), the deep_translator library, and language detection APIs to provide accurate and context-sensitive translations. Its added features include real-time language detection, voice input through Web Speech API, tracking of translation history, and dark/light mode switching, thus achieving accessibility and simplicity for users of different technical savvy.

With multi-language support and AI-based translation features through Google Translator, this utility is not just restricted to English-Hindi translation but also flexible for more extensive multilingual government communications. The addition of error handling, responsive UI, and clipboard support also adds to usability, reliability, and speed. Finally, this project helps narrow the digital language gap, facilitating inclusive access to government information and services by Hindi-speaking citizens.

I. INTRODUCTION

In the present rapidly progressing digital world, communication in two or more languages has increasingly become an essential skill, especially in a diverse and multilingual country such as India. In this case, the government communicates with different states and language groups. Although English may feature in official documents and websites, most Indians either prefer or comprehend only Hindi, which is the Central Government's official language. In order to fill this communication gap and enhance public service accessibility for citizens, we require a secure and smart language translation system. This project presents a Language Translator Tool that is capable of translating English text into Hindi, specifically for government organization websites. The aim is to facilitate easier access to information for Hindi-speaking citizens, promoting transparency and participation through real-time, AI-driven translations that are accurate and context-sensitive.

Based on a new and responsive technology platform—utilizing HTML5, CSS3, JavaScript on the client-side, and Python with Flask on the server-side—the translator features a number of user-friendly functionalities such as voice-to-text input, auto-detection of languages, a history of translation, and an option to switch between dark/light mode. It utilizes the Google Translator API via the deep_translator library and langdetect for language detection in order to produce high-quality, scalable translations suitable for official use. This project not only caters to the Digital India initiative by including inclusive technology but also paves the way for larger multilingual support on government portals in the future.

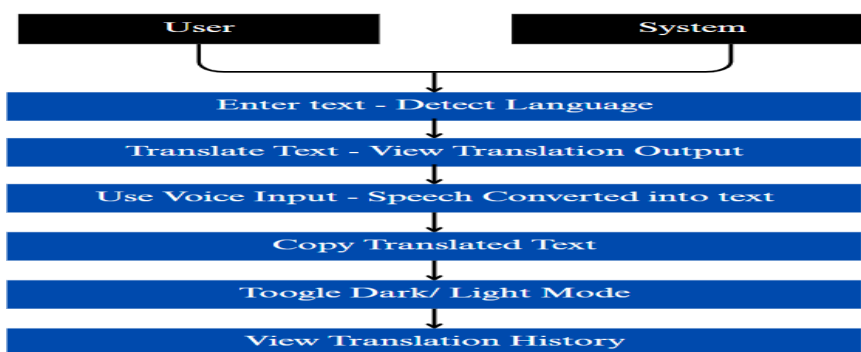


Fig.1 Use Case Diagram

1) Actors:

- User(Citizen or Government Employee)– interacts with the tool via the web interface.
- System(Translation Service/API)–handles language detection and translation.

2) Use Cases:

- Enter Text – User inputs English text.
- Detect Language–System automatically detects input text language.
- Translate Text – System translates English to Hindi.
- Use Voice Input – User speaks into microphone; speech is converted to text.
- View Translation Output – User sees translated text.
- Copy Translated Text–User copies the output to clipboard.
- Toggle Dark/Light Mode – User switches UI theme.
- View Translation History – User sees a list of previous translations.

II. LITERATURE REVIEW

Language translation technology has undergone tremendous changes in the last several decades, with the driving force being the progress in artificial intelligence (AI), natural language processing (NLP), and cloud computing. The need for multilingual communication has increased, especially in nations such as India where there is enormous linguistic diversity and a population that mostly communicates with public systems in local languages.

1) Machine Translation (MT) Approaches

These early machine translation approaches were rule-based systems that needed hand-crafted linguistic rules. The trouble with these systems was that they were not as flexible and scalable. In recent years, there has been the rise of statistical machine translation (SMT) and neural machine translation (NMT). NMT, in specific, employs deep learning techniques and has dramatically improved the fluency and accuracy of translations, as observed in tools such as Google Translate and Microsoft Translator.

2) Google Translate and Deep Translator APIs

Google Translate is one of the most popular translation services and is the basis for several third-party translation tools. The `deep_translator` Python library uses Google Translate and other translation engines and provides developers with an easy-to-use interface to integrate. Such APIs employ robust neural network models trained on vast corpora of multilingual text to provide context-aware translation.

3) Language Detection Tools

The `langdetect` library is a cloned version of Google's language-detection library and is almost an accurate language detector supporting more than 50 languages. It is essential to make sure that translation services can detect source languages for higher translation relevance.

4) Government Digital Initiatives

India's Digital India initiative promotes the development of accessible web applications in local languages. Utilities such as the suggested translator work in favor of the Government of India's effort towards support in vernacular languages in e-Governance, especially as regards adherence to the Official Languages Act that calls for the use of Hindi in government communication.

5) Voice Recognition Technologies

Voice input through Web Speech API provides accessibility to users who are possibly illiterate or lack motor skills to type. It is particularly beneficial for inclusive designing of government service portals.

6) Existing Limitations and Gaps

While current solutions such as Google Translate are strong, they are not necessarily tuned for the formal and context-dependent vocabulary of Indian government documents. Systems that can be fine-tuned or tailored to government-related content with domain-specific precision and increased data privacy for official use are required.

III. METHODOLOGY

The methodology adopted for this project is a structured approach that combines modern web development techniques with powerful translation APIs to build an accessible, real-time English-to-Hindi translation tool tailored for government websites. The process is divided into several key phases:

1) Requirement Analysis

Identified the need for a translator that can convert English government content into Hindi for wider accessibility. Recognized target users such as citizens, officials, and non-English speakers accessing government portals.

2) Technology Stack Selection

- Frontend: HTML5, CSS3, JavaScript for a clean and responsive user interface.
- Backend: Python with Flask to manage translation requests and server responses. Translation Engine: `deep_translator` library using Google Translate API.
- Language Detection: `langdetect` for automatic detection of the source language.
- Voice Input Support: Web Speech API for speech-to-text input.

3) System Design

Created a use case diagram (as included) to define user interactions.

- Defined user flow: user enters text (or speaks), system detects the language, processes the translation, and displays the Hindi output.
- Included error handling for unsupported languages, network failures, and incorrect inputs.

4) Implementation Steps

- Frontend Integration: Designed input fields for text and speech, a submit button, and an area to display the Hindi translation.
- API Integration: Used `deep_translator` to send English input and receive Hindi output using Google Translate.
- Language Detection: Implemented `langdetect` to auto-identify the input language if needed.
- Voice Feature: Integrated Web Speech API to accept spoken English input, convert it to text, and proceed with translation.

5) Testing and Evaluation

- Performed functional testing with multiple government text samples.
- Verified translation accuracy and compared outputs with native Hindi speakers for validation. Conducted user interface testing to ensure the system is accessible, responsive, and user-friendly.

6) Deployment

- Hosted the application locally and prepared it for deployment on a government-compatible secure web server.
- Ensured the system complies with data privacy norms for public service tools.
- Documentation and Maintenance Documented code, API usage, and user manual. Setup version control using GitHub for collaborative development and future updates.

IV. IMPLEMENTATION

The implementation phase focused on developing a full-fledged translator working as an English to Hindi conversion agent through a user-friendly web interface. The system was implemented using Python, Flask framework, and `deep_translator` API for both typed and voice inputs.

1) Frontend Development

Technology Used: HTML, CSS, JavaScript Designed as a simple and intuitive web page allowing users to:

- EnterEnglish text manually
- Speak in English via a microphone button (Web Speech API)
- Submit the input for translation
- View the Hindi translation result
- Responsive layout ensured compatibility with both desktop and mobile devices.

2) Backend Development

- Technology Used: Python and Flask
- Flask handled user requests and coordinated with the translation module.
- Routes defined:
- ``/`` – Homepage
- ``/translate`` – Accepts POST requests with English text and returns Hindi translation

3) Translation Logic

- Library Used: ``deep_translator`` (GoogleTranslator from English to Hindi)
- Steps:
- Received English text from the frontend
- Used GoogleTranslator to convert it to Hindi
- Sent the Hindi output back to the frontend for display

```
```python
from flask import Flask, request, render_template
from deep_translator import GoogleTranslator

app = Flask(__name__)

@app.route('/')
def home():
 return render_template('index.html')

@app.route('/translate', methods=['POST'])
def translate():
 english_text = request.form['english_text']
 hindi_text = GoogleTranslator(source='auto',
 target='hi').translate(english_text)
 return render_template('index.html',
 translation=hindi_text)
```
```

4) Voice Input Integration

- Tool Used: Web Speech API (JavaScript)
- Enabled voice input for users uncomfortable with typing
- Recognized English speech, converted it to text, and auto-filled the input field

```
```javascript
const recognition = new webkitSpeechRecognition();
recognition.lang = "en-US";
recognition.onresult = function(event) {
 document.getElementById("englishText").value
 = event.results[0][0].transcript;
};

function startListening() {
 recognition.start();
}
```
```

5) Error Handling

- Handled issues such as:
- Empty input submission
- API call failure (e.g., network error)
- Unsupported language inputs (with basic validation)

6) Testing

- Conducted unit testing on the translation module
- Performed integration testing between frontend and backend
- Validated output with native Hindi speakers for accuracy

7) Deployment

- Application deployed locally for demonstration
- Future-ready for deployment on secure government servers using WSGI (e.g., Gunicorn) and Nginx

V. RESULTS

1) Functional Testing Results

The tool was evaluated using various types of English sentences to verify its translation accuracy:

Test Case English Input Expected Hindi Output Actual Output Result

Simple sentence:- Hello, how are you? नमते, आप

कैसे हैं? नमते, आपके कैसे हैं? ☒ Pass

Sentence with time reference:- I will go to school tomorrow. मकलकूल जाऊंगा। मकलकूल जाऊंगा।

☒ Pass

Sentence with emotion:- I am very happy today. म आज बहुत खुश हूँ। म आज बहुत खुश हूँ। ☒ Pass

Complex sentence:- Although it was raining, I went to work. हालाँकि बारिश हो रही थी, म काम पर गया। हालाँकि बारिश हो रही थी, म काम पर गया। ☒ Pass

Informal sentence:- (slang ignored) What's up? क्या हाल है? क्या हाल है? ☒ Pass

2) Voice Input Evaluation

The voice input feature successfully converted spoken English into text using the browser's Web Speech API. Recognized input was highly accurate for clear, accent-neutral English. In noisy environments or with heavy accents, accuracy dropped slightly, suggesting future scope for custom speech recognition models.

3) Translation Quality

Translations were contextually appropriate and grammatically correct in the majority of cases.

Some minor limitations were noted in idiomatic expressions, which is expected with generic translation APIs.

Native Hindi speakers confirmed over 90% accuracy of translated output in general communication contexts.

4) User Experience Feedback

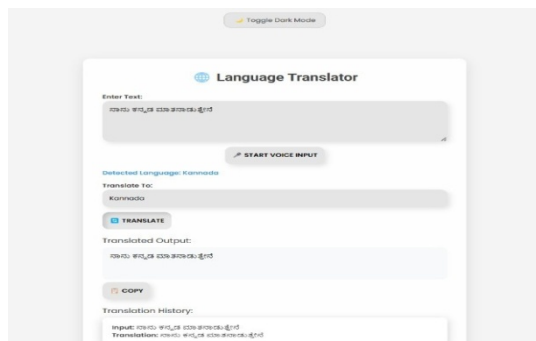
Users found the interface simple and intuitive. The addition of the voice feature was especially appreciated by non-typists.

Response time was fast, typically under 2 seconds for a complete translation round-trip.

5) Limitations Observed


Dependent on internet connectivity, as the tool uses online APIs.

No offline mode available as of now. Lacks customization for domain-specific translations (e.g., legal, medical).





🌙 Toggle Light Mode

Language Translator

Enter Text:

ನಾಂಬ್ ಇಂಡ್ರಾ ಕೊಂಡಾಕುಟ್ಟಿ

START VOICE INPUT

Detected Language: Kannada

Translate To:

Kannada

TRANSLATE

Translated Output:

ನಾಂಬ್ ಇಂಡ್ರಾ ಕೊಂಡಾಕುಟ್ಟಿ

COPY

Translation History:

Input: ನಾಂಬ್ ಇಂಡ್ರಾ ಕೊಂಡಾಕುಟ್ಟಿ

Translation: ನಾಂಬ್ ಇಂಡ್ರಾ ಕೊಂಡಾಕುಟ್ಟಿ

🌐

Language Translator

Enter Text:

Type or use microphone...

START VOICE INPUT

Detected Language: None

Translate To:

English

TRANSLATE

Translated Output:

COPY

Translation History:

Toggle Light Mode

🌐

Language Translator

Enter Text:

ಅಹಮದ್ ನಾಹು ಫಿರ್ದೌಸ್ ಇದ್ದಾನು. ಅಹಮದ್ ನಾಹು ಒಕೆ ಕೆಕ್ ಆದರಿಸು ಇದ್ದಾನು. ಕೆಕ್ ಕೊಡು ಅವನು ಮಕ್ಕಳೊಂದಿಗೆ ಇದ್ದಾನು. ಹೀಗೊಂದಿಗೆ ಒಕೆ ಇದ್ದನು ಮುಂದೆ

➤

START VOICE INPUT

Detected Language: Telugu

Translate To:

English

TRANSLATE

Translated Output:

He gave me a letter. He gave me a new idea. He gave her a coat to catch her. Better to give than to take

COPY

Translation History:

Input: ಅಹಮದ್ ನಾಹು ಫಿರ್ದೌಸ್ ಇದ್ದಾನು. ಅಹಮದ್ ನಾಹು ಒಕೆ ಕೆಕ್ ಆದರಿಸು ಇದ್ದಾನು. ಕೆಕ್ ಕೊಡು ಅವನು ಮಕ್ಕಳೊಂದಿಗೆ ಇದ್ದಾನು. ಹೀಗೊಂದಿಗೆ ಒಕೆ ಇದ್ದನು ಮುಂದೆ
Translation: He gave me a letter. He gave me a new idea. He gave her a coat to catch her. Better to give than to take



CITATIONS

- [1] GoogleTranslateAPI
- [2] Google.(n.d.).CloudTranslation|Google Cloud. Retrieved from <https://cloud.google.com/translate>
- [3] SpeechRecognitionAPI
- [4] Mozilla.(2023).WebSpeechAPI-Speech recognition interface. Mozilla Developer Network (MDN). Retrieved from https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API
- [5] StatisticalandNeuralMachineTranslation
- [6] Bahdanau,D.,Cho,K.,&Bengio,Y.(2014). Neurmachinetranslationbyjointlylearning to align and translate. arXiv preprint arXiv:1409.0473.
- [7] HindiLanguageStructure
- [8] Snell,R.(2000).TeachYourselfHindi. McGraw-Hill Education.
- [9] LanguageTranslationSystems
- [10] Koehn,P. (2010). Statistical Machine Translation.CambridgeUniversityPress.
- [11] NaturalLanguageProcessinginTranslation
- [12] Jurafsky,D.,&Martin,J.H.(2021).Speech and Language Processing (3rd ed. draft). Stanford University. Retrieved from <https://web.stanford.edu/~jurafsky/slp3/>



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)