# Hindi Text Refinement Tool

## About This Project

The **Hindi Text Refinement Tool** is a web-based application designed to process and refine Hindi text. It leverages advanced Natural Language Processing (NLP) models to perform the following tasks:

1. **Translation**: Translates Hindi text to English and vice versa.
2. **Paraphrasing**: Rephrases English text to generate alternative versions while preserving the original meaning.
3. **Grammar Correction**: Identifies and corrects grammatical errors in English text.
4. **Text Analysis**: Provides word and character counts, as well as similarity scores between the original and processed text.

This tool is built using **Python** (Flask for the backend) and **HTML/CSS** for the frontend. It integrates with the Hugging Face Transformers library to utilize state-of-the-art NLP models for text processing.

## Features

- **Hindi to English Translation**: Seamlessly translate Hindi text into English.
- **English Paraphrasing**: Generate paraphrased versions of English text.
- **Grammar Correction**: Automatically correct grammatical errors in English text.
- **Text Analysis**: Display word count, character count, and similarity scores.
- **User-Friendly Interface**: A clean and interactive UI for easy text input and output.

---

## How It Works

1. **Input Hindi Text**: Enter your Hindi text into the input box.
2. **Process Text**: The application translates the text to English, paraphrases it, corrects grammar, and translates it back to Hindi.

3. **View Results**: The processed text is displayed along with word counts, character counts, and similarity scores.

---

# Technologies Used

- **Backend**: Python, Flask, Hugging Face Transformers
- **Frontend**: HTML, CSS
- **NLP Models**: T5 for paraphrasing and grammar correction
- **Translation**: Google Translate API (via deep-translator)

# Installation

1. Clone the repository:

   git clone https://github.com/your-username/hindi-text-refinement-tool.git

2. Install the required dependencies:

   pip install -r requirements.txt

3. Run the Flask application:

   python app.py

4. Open your browser and navigate to http://127.0.0.1:5000.

---

# Usage

1. Enter Hindi text in the input box.
2. Click "Process Text" to see the refined output.
3. View the original and processed text along with analysis metrics.

---

## Contributing

Contributions are welcome! If you'd like to contribute, please follow these steps:

1. Fork the repository.
2. Create a new branch for your feature or bug fix.
3. Commit your changes.
4. Submit a pull request.

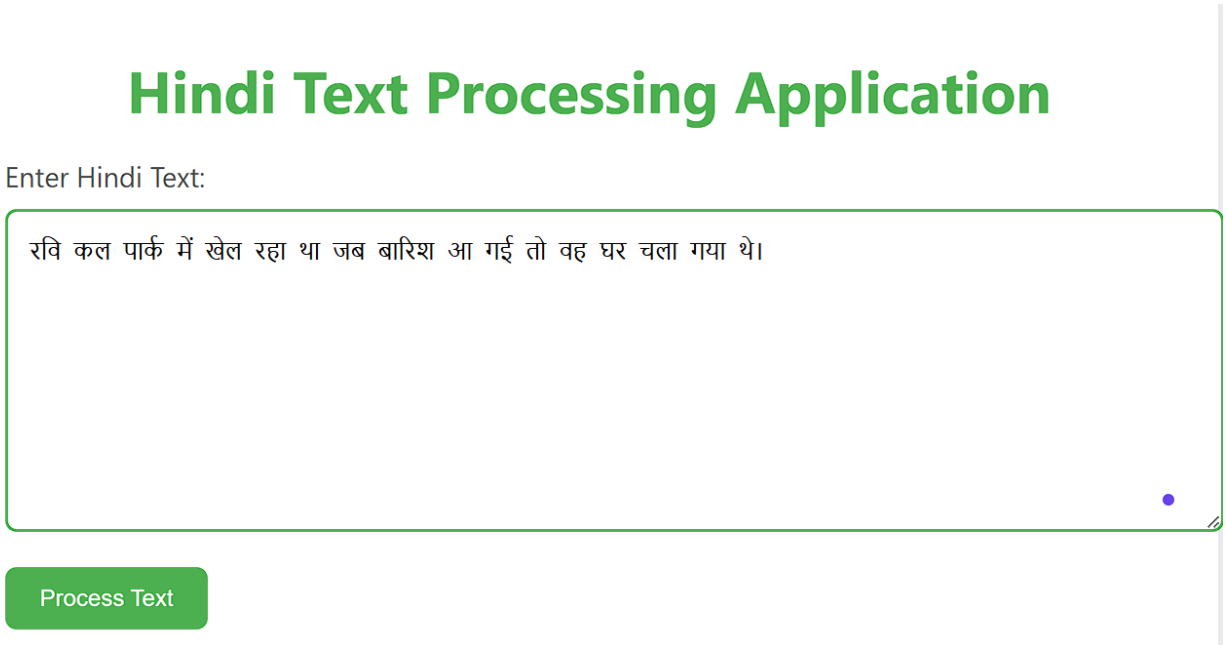
---

## Acknowledgments


- Hugging Face for providing pre-trained NLP models.
- Google Translate for translation services.
- Flask for the web framework.


---

## Screenshots



## Original Hindi Text:

> रवि कल पार्क में खेल रहा था जब बारिश आ गई तो वह घर चला गया थे।

| Word Count: | Character Count: |
| --- | --- |
| 17 | |

## Processed Hindi Text:

> रवि कल पार्क में खेल रहा था जब बारिश हुई, वह घर गया।

| Word Count: | Character Count: |
| --- | --- |
| 13 | |

## Similarity Scores:

> **Cosine Similarity Score:** 0.95837027

```
README.md: 100%|          | 3.73k/3.73k [00:00<?, ?B/s]
sentence_bert_config.json: 100%|          | 53.0/53.0 [00:00<?,
config.json: 100%|          | 629/629 [00:00<?, ?B/s]
model.safetensors: 100%|          | 90.9M/90.9M [00:18<00:00, 4
tokenizer_config.json: 100%|          | 314/314 [00:00<?, ?B/s]
vocab.txt: 100%|          | 232k/232k [00:00<00:00, 528kB/s]
tokenizer.json: 100%|          | 466k/466k [00:00<00:00, 759kB/
special_tokens_map.json: 100%|          | 112/112 [00:00<?, ?B/
1_Pooling%2Fconfig.json: 100%|          | 190/190 [00:00<?, ?B/
 * Serving Flask app 'app1'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production
 WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
2025-03-01 19:07:24.610934: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on.
You may see slightly different numerical results due to floating-point round-off errors from diff
erent computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0
`.
2025-03-01 19:07:26.333002: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on.
You may see slightly different numerical results due to floating-point round-off errors from diff
erent computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0
```

```
2025-03-01 19:07:26.333002: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on.
You may see slightly different numerical results due to floating-point round-off errors from diff
erent computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0
`.
WARNING:tensorflow:From C:\Users\LENOVO FLEX\AppData\Roaming\Python\Python312\site-packages\tf_ke
ras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use
 tf.compat.v1.losses.sparse_softmax_cross_entropy instead.

You are using the default legacy behaviour of the <class 'transformers.models.t5.tokenization_t5.
T5Tokenizer'>. This is expected, and simply means that the `legacy` (previous) behavior will be u
sed so nothing changes for you. If you want to use the new behaviour, set  * Debug * D * D * D *
Debugger is activ * Debug * D * D * D * D * D * Debug * Debugger is a * D * D * Debug * Debug
 * Debugger is active!
 * Debugger PIN: 288-098-155
127.0.0.1 - - [01/Mar/2025 19:08:09] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [01/Mar/2025 19:09:02] "POST /process HTTP/1.1" 200 -
127.0.0.1 - - [01/Mar/2025 19:14:22] "POST /process HTTP/1.1" 200 -
127.0.0.1 - - [01/Mar/2025 19:22:04] "POST /process HTTP/1.1" 200 -
```