Natural language processing - Text Error correction web application

Siddhartha Sandilya (Senior Specialist- Data Strategy & Governance)
Siddhartha.Sandilya@merck.com

Problem definition:

In this section we will discuss about the following sections.

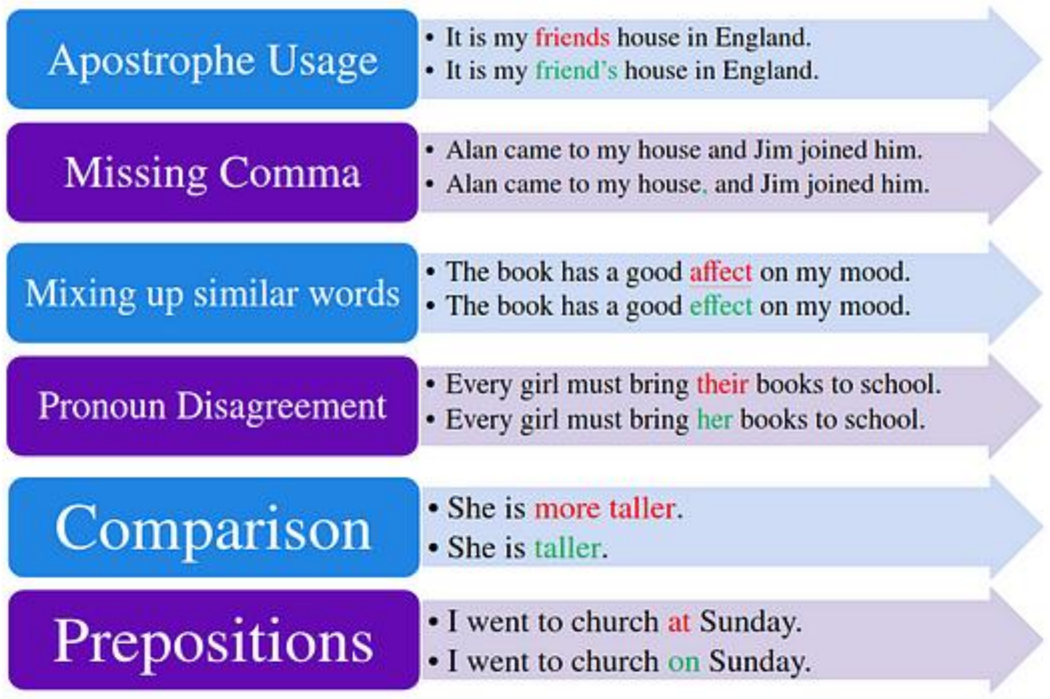
1. Develop a text error correction application that can detect and correct sentence structure mistakes, subject-verb agreement errors, punctuation issues, and incorrect word usage.
2. The application should include both a web-based interface and backend processing using Flask and Python libraries.

Introduction:

Grammatical Error Correction (GEC) systems aim to correct grammatical mistakes in the text. Grammarly is an example of such a grammar correction product. Error correction can improve the quality of written text in emails, blogs and chats.

GEC task can be thought of as a sequence to sequence task where a Transformer model is trained to take an ungrammatical sentence as input and return a grammatically correct sentence. In this blog, we show how you can train such a model and use Weights and Biases to monitor the performance of the model as it trains. The code is also present on Github here. And sample data code file is placed here.

The errors encountered in written language can be of different types as shown in the image below.



Apostrophe Usage

- It is my friends house in England.
- It is my friend's house in England.

Missing Comma

- Alan came to my house and Jim joined him.
- Alan came to my house, and Jim joined him.

Mixing up similar words

- The book has a good affect on my mood.
- The book has a good effect on my mood.

Pronoun Disagreement

- Every girl must bring their books to school.
- Every girl must bring her books to school.

Comparison

- She is more taller.
- She is taller.

Prepositions

- I went to church at Sunday.
- I went to church on Sunday.

Complete solution guide:

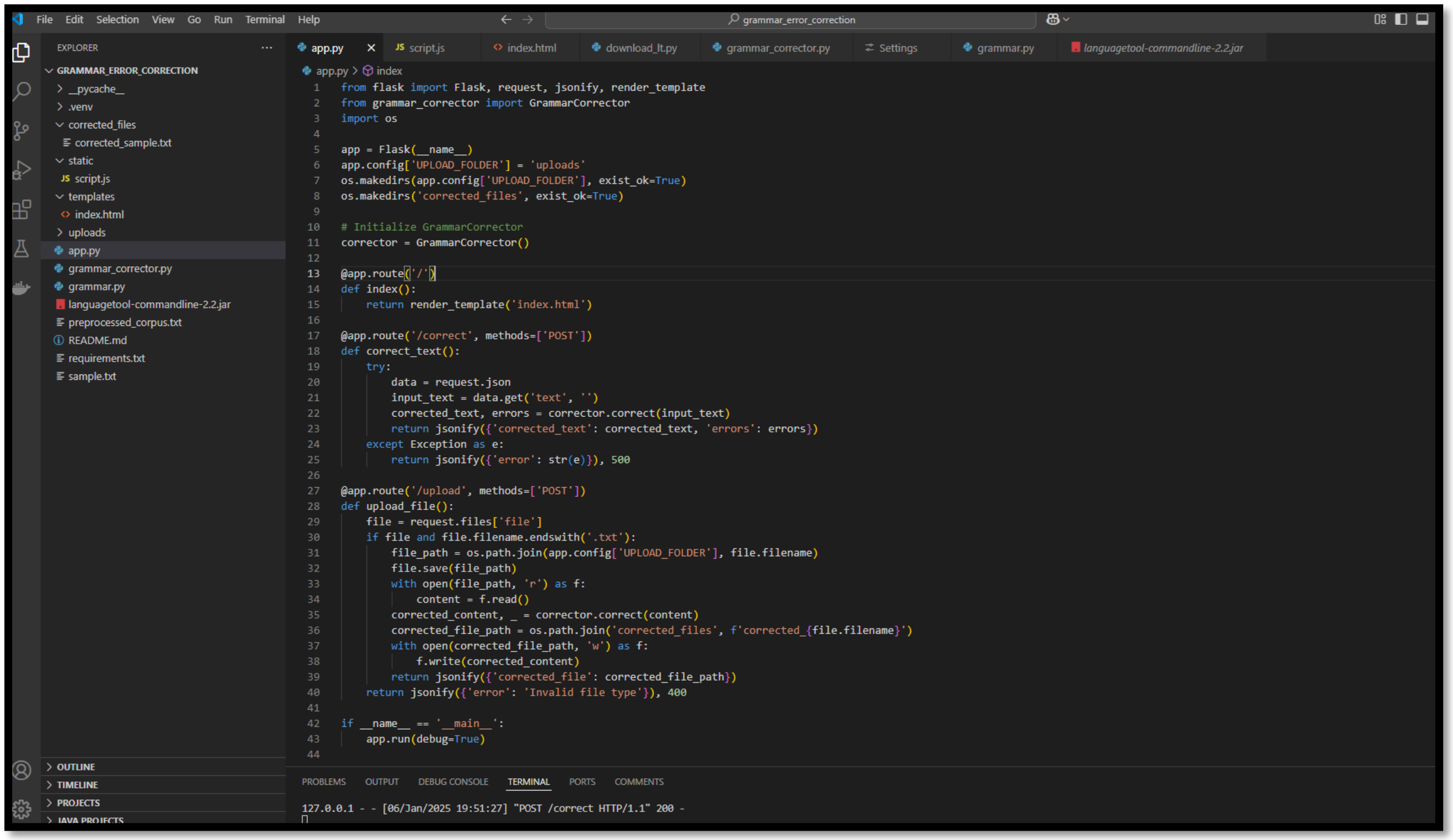
Pre-requisite:

1. Basic Knowledge of Python
2. Basic Knowledge of NLP
3. Java Installed
4. Basic Knowledge of Visual Studio

1. **Folder structure in VS:** We will be using the visual studio IDE for developing this application.
    a. text_error_correction/

    b. ├── app.py   # Flask backend
    c. ├── T5_Grammar.py   # Error correction logic
    d. ├── c4_dataprep200M.py   # DATA Preparation
    e. ├── requirements.txt   # Dependencies
    f. ├── templates/   # HTML templates
    g. │   └── index.html   # Front-end interface
    h. ├── static/   # Static files (CSS, JavaScript)
    i. │   └── script.js   # JavaScript for the front-end
    j. ├── uploads/   # Directory for uploaded files
    k. ├── corrected_files/   # Directory for corrected files

2. **Backend Application (Flask):** Light weight, easy to integrate, simple and rapid development framework.
    a. **Code link: https://github.com/SIDDHARTHAS05/AI-ML/blob/main/app.py**



```python
from flask import Flask, request, jsonify, render_template
from grammar_corrector import GrammarCorrector
import os

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = 'uploads'
os.makedirs(app.config['UPLOAD_FOLDER'], exist_ok=True)
os.makedirs('corrected_files', exist_ok=True)

# Initialize GrammarCorrector
corrector = GrammarCorrector()

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

@app.route('/correct', methods=['POST'])
def correct_text():
    try:
        data = request.json
        input_text = data.get('text', '')
        corrected_text, errors = corrector.correct(input_text)
        return jsonify({'corrected_text': corrected_text, 'errors': errors})
    except Exception as e:
        return jsonify({'error': str(e)}), 500

@app.route('/upload', methods=['POST'])
def upload_file():
    file = request.files['file']
    if file and file.filename.endswith('.txt'):
        file_path = os.path.join(app.config['UPLOAD_FOLDER'], file.filename)
        file.save(file_path)
        with open(file_path, 'r') as f:
            content = f.read()
        corrected_content, _ = corrector.correct(content)
        corrected_file_path = os.path.join('corrected_files', f'corrected_{file.filename}')
        with open(corrected_file_path, 'w') as f:
            f.write(corrected_content)
        return jsonify({'corrected_file': corrected_file_path})
    return jsonify({'error': 'Invalid file type'}), 400

if __name__ == '__main__':
    app.run(debug=True)
```

```
127.0.0.1 - - [06/Jan/2025 19:51:27] "POST /correct HTTP/1.1" 200 -
```

3. Grammar_corrector.py & c4_dataprep 200M. py :

Link has been provided earlier and Kaggle data has been used to train the model.
https://www.kaggle.com/datasets/dariocioni/c4200m/code

4. **Requirements.txt:** Keep all the packages to install in one notepad and

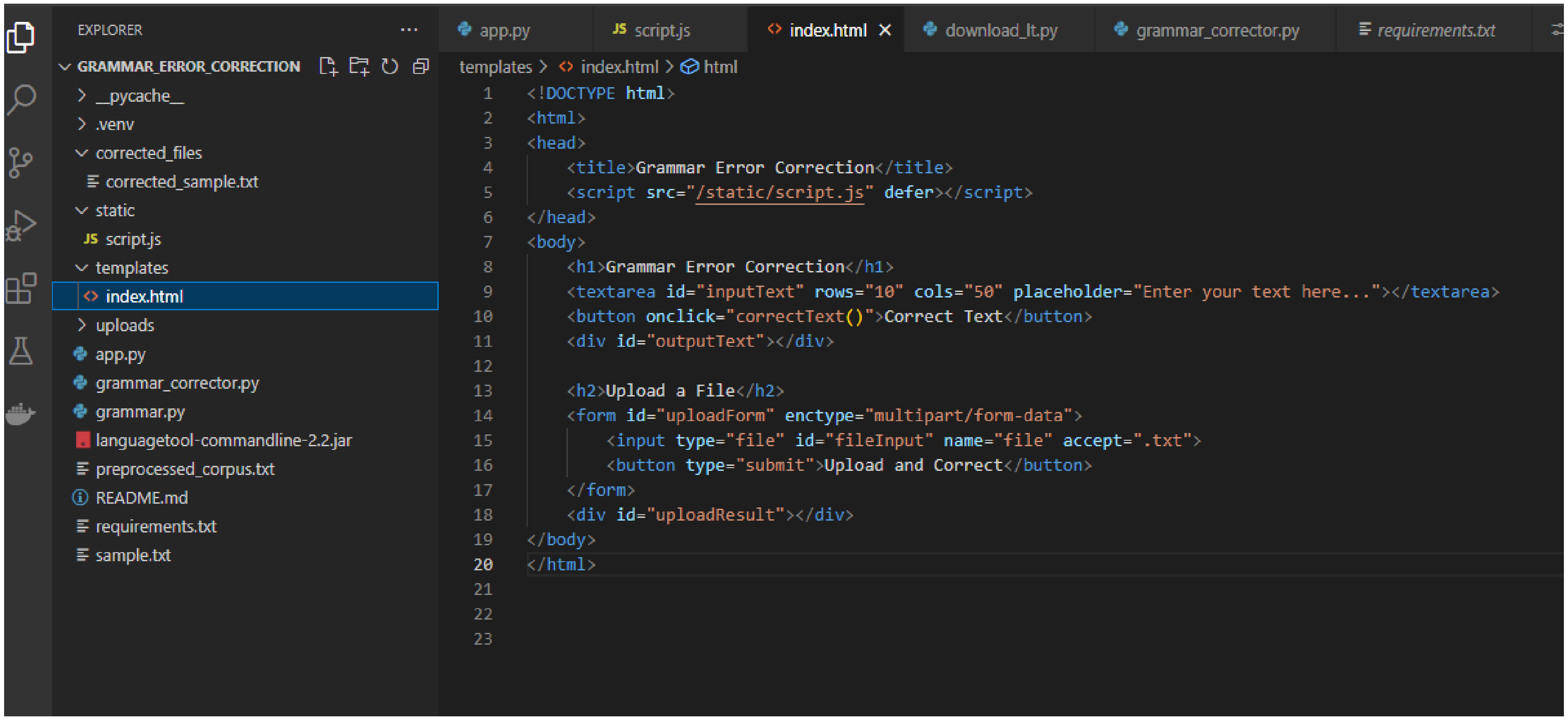
   Run in command prompt/git bash to install all packages at once : pip install -r requirements.txt

```
Flask==2.1.1
language-tool-python==2.7.0
Werkzeug==2.2.3
nltk
pandas
matplotlib
numpy
transformers
datasets
# evaluate rouge score
rouge
torch
pytorch-lightning
datasets
tqdm
pandas
sentencepiece
transformers
wandb
```

5. HTML & Java Script for frontend application:

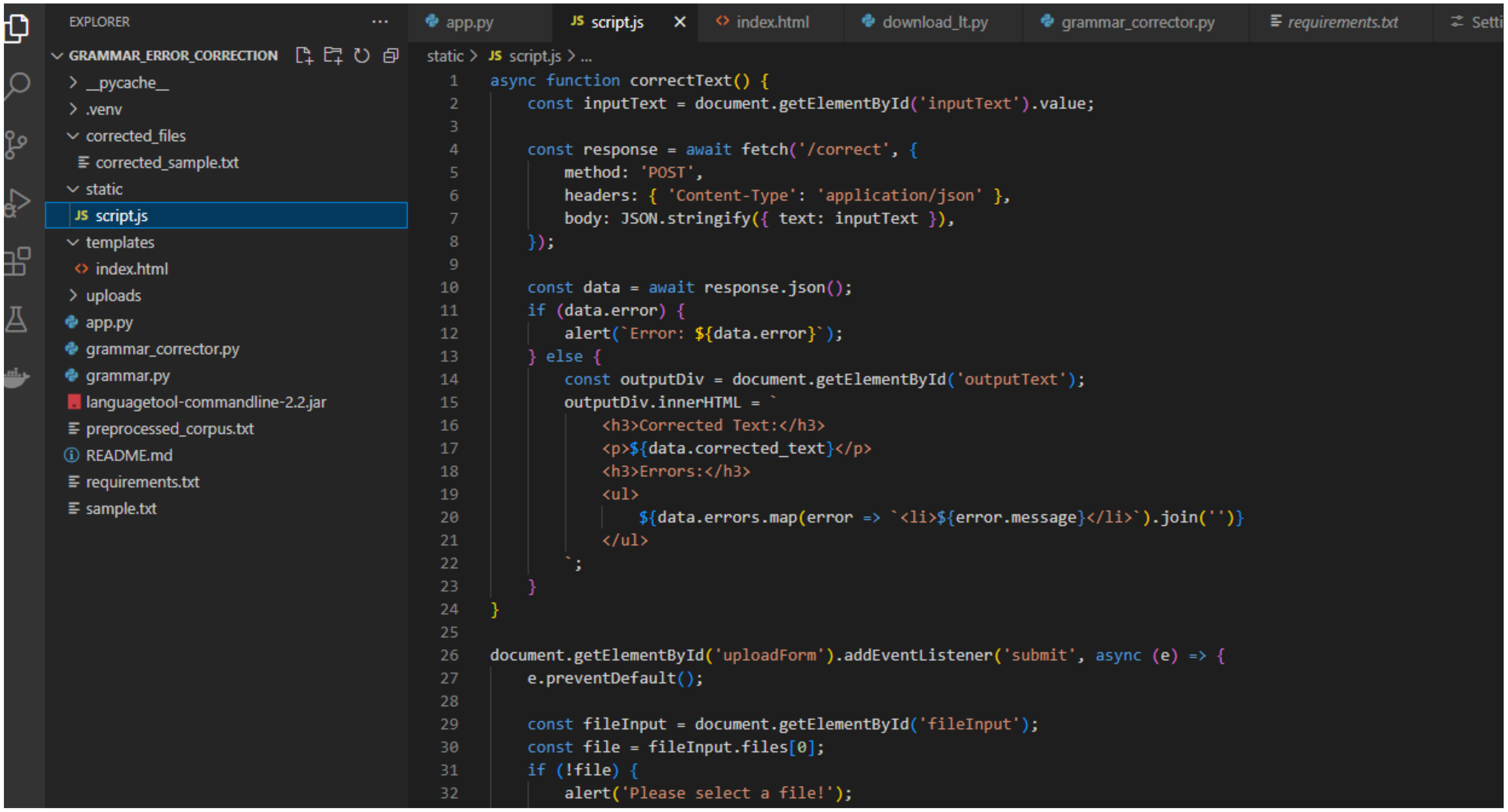
HTML code

Java script code:



```html
<!DOCTYPE html>
<html>
<head>
    <title>Grammar Error Correction</title>
    <script src="/static/script.js" defer></script>
</head>
<body>
    <h1>Grammar Error Correction</h1>
    <textarea id="inputText" rows="10" cols="50" placeholder="Enter your text here..."></textarea>
    <button onclick="correctText()">Correct Text</button>
    <div id="outputText"></div>

    <h2>Upload a File</h2>
    <form id="uploadForm" enctype="multipart/form-data">
        <input type="file" id="fileInput" name="file" accept=".txt">
        <button type="submit">Upload and Correct</button>
    </form>
    <div id="uploadResult"></div>
</body>
</html>
```

Java script:



EXPLORER

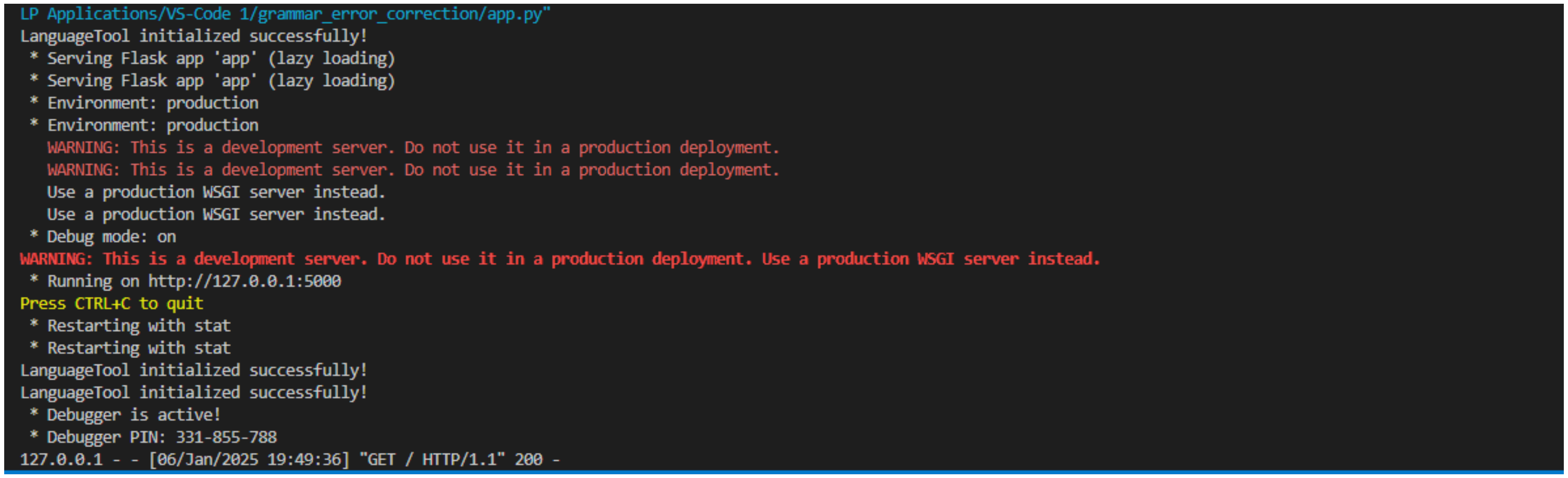
GRAMMAR_ERROR_CORRECTION
- __pycache__
- .venv
- corrected_files
  - corrected_sample.txt
- static
  - script.js
- templates
  - index.html
- uploads
- app.py
- grammar_corrector.py
- grammar.py
- languagetool-commandline-2.2.jar
- preprocessed_corpus.txt
- README.md
- requirements.txt
- sample.txt

app.py | script.js | index.html | download_lt.py | grammar_corrector.py | requirements.txt | Setti

static > script.js > ...

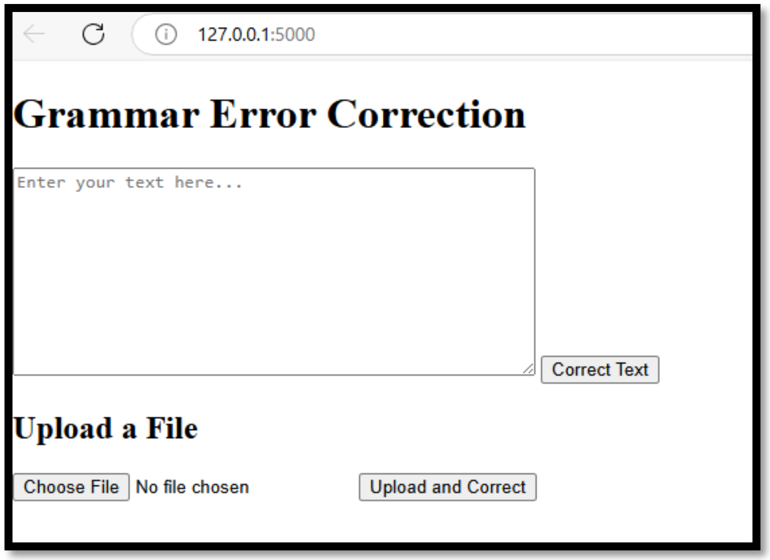
```javascript
async function correctText() {
    const inputText = document.getElementById('inputText').value;

    const response = await fetch('/correct', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ text: inputText }),
    });

    const data = await response.json();
    if (data.error) {
        alert(`Error: ${data.error}`);
    } else {
        const outputDiv = document.getElementById('outputText');
        outputDiv.innerHTML = `
            <h3>Corrected Text:</h3>
            <p>${data.corrected_text}</p>
            <h3>Errors:</h3>
            <ul>
                ${data.errors.map(error => `<li>${error.message}</li>`).join('')}
            </ul>
        `;
    }
}

document.getElementById('uploadForm').addEventListener('submit', async (e) => {
    e.preventDefault();

    const fileInput = document.getElementById('fileInput');
    const file = fileInput.files[0];
    if (!file) {
        alert('Please select a file!');
```

6. Run the files in command prompt terminal/git bash extension and click on http://127.0.0.1:5000



```
LP Applications/VS-Code 1/grammar_error_correction/app.py"
LanguageTool initialized successfully!
 * Serving Flask app 'app' (lazy loading)
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
   Use a production WSGI server instead.
 * 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
 * Restarting with stat
LanguageTool initialized successfully!
LanguageTool initialized successfully!
 * Debugger is active!
 * Debugger PIN: 331-855-788
127.0.0.1 - - [06/Jan/2025 19:49:36] "GET / HTTP/1.1" 200 -
```

7: Text error application:



127.0.0.1:5000

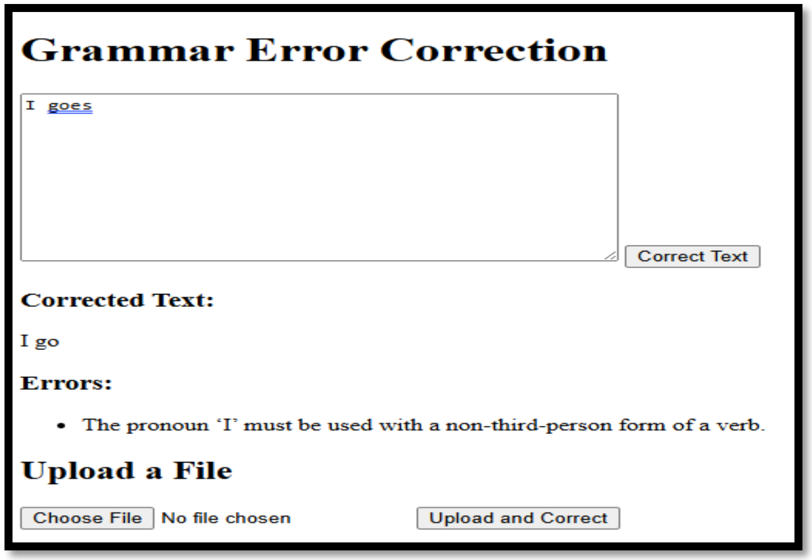
Grammar Error Correction

Enter your text here...

Correct Text

Upload a File

Choose File No file chosen Upload and Correct



Grammar Error Correction

I goes

Correct Text

Corrected Text:

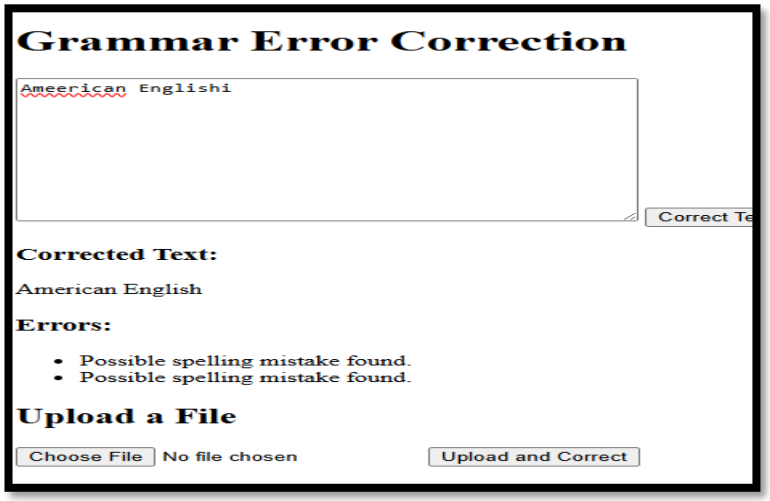
I go

Errors:

- The pronoun 'I' must be used with a non-third-person form of a verb.

Upload a File

Choose File No file chosen Upload and Correct



Grammar Error Correction

Ameerican Englishi

Correct Te

Corrected Text:

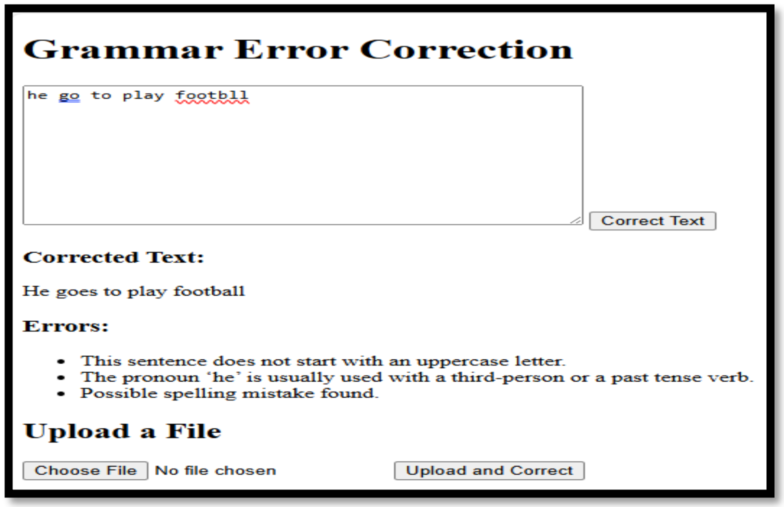
American English

Errors:

- Possible spelling mistake found.
- Possible spelling mistake found.

Upload a File

Choose File No file chosen Upload and Correct



Grammar Error Correction

he go to play footbll

Correct Text

Corrected Text:

He goes to play football

Errors:

- This sentence does not start with an uppercase letter.
- The pronoun 'he' is usually used with a third-person or a past tense verb.
- Possible spelling mistake found.

Upload a File

Choose File No file chosen Upload and Correct

:

8. Other packages: There are other python packages as well apart from Language Tool some of them are :

1. Gram former
2. Ginger
3. Pyaspeller

Useful Links: https://huggingface.co/deep-learning-analytics/GrammarCorrector

https://deeplearninganalytics.org/nlp-building-a-grammatical-error-correction-model/