AI-Powered Translation Application

1. Backend Code (Node.js + Express)

\*\*`server.js`\*\*

```javascript

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors');

const translate = require('@vitalets/google-translate-api'); // Install this package for translation

const app = express();

const PORT = process.env.PORT || 5000;

app.use(cors());

app.use(bodyParser.json());

// API to handle translation requests

app.post('/translate', (req, res) => {

const { text, sourceLanguage, targetLanguage } = req.body;

translate(text, { from: sourceLanguage, to: targetLanguage })

.then(response => {

res.json({ translation: response.text });

})

.catch(err => {

res.status(500).json({ error: err.message });

});

});

// API to get supported languages

app.get('/languages', (req, res) => {

const languages = [

{ code: 'en', name: 'English' },

{ code: 'es', name: 'Spanish' },

{ code: 'fr', name: 'French' },

// Add more languages as needed

];

res.json(languages);

});

app.listen(PORT, () => {

console.log(`Server is running on http://localhost:${PORT}`);

});

```

2. Frontend Code (React)

`App.js`

```javascript

import React, { useState, useEffect } from 'react';

import axios from 'axios';

const App = () => {

const [text, setText] = useState('');

const [sourceLanguage, setSourceLanguage] = useState('en');

const [targetLanguage, setTargetLanguage] = useState('es');

const [translation, setTranslation] = useState('');

const [languages, setLanguages] = useState([]);

useEffect(() => {

// Fetch supported languages

axios.get('http://localhost:5000/languages')

.then(response => {

setLanguages(response.data);

})

.catch(error => {

console.error('Error fetching languages:', error);

});

}, []);

const handleTranslate = () => {

axios.post('http://localhost:5000/translate', { text, sourceLanguage, targetLanguage })

.then(response => {

setTranslation(response.data.translation);

})

.catch(error => {

console.error('Error translating text:', error);

});

};

return (

<div>

<h1>AI-Powered Translation App</h1>

<textarea value={text} onChange={e => setText(e.target.value)} placeholder="Enter text here" />

<div>

<select value={sourceLanguage} onChange={e => setSourceLanguage(e.target.value)}>

{languages.map(lang => (

<option key={lang.code} value={lang.code}>{lang.name}</option>

))}

</select>

<select value={targetLanguage} onChange={e => setTargetLanguage(e.target.value)}>

{languages.map(lang => (

<option key={lang.code} value={lang.code}>{lang.name}</option>

))}

</select>

<button onClick={handleTranslate}>Translate</button>

</div>

<h2>Translation:</h2>

<p>{translation}</p>

</div>

);

};

export default App;

```

3. Setup Instructions

Backend Setup

1. Create a new directory for your backend project.

2. Initialize a new Node.js project:

```bash

npm init -y

```

3. Install required packages:

```bash

npm install express body-parser cors @vitalets/google-translate-api

```

4. Create the `server.js` file and paste the backend code above.

5. Start the server:

```bash

node server.js

```

Frontend Setup

1. Create a new React project using Create React App:

```bash

npx create-react-app translation-app

```

2. Navigate to the project directory:

```bash

cd translation-app

```

3. Install Axios for making API requests:

```bash

npm install axios

```

4. Replace the content of `src/App.js` with the React code provided above.

5. Start the React app:

```bash

npm start

```