WEBSITE

1. Frontend (HTML, CSS, JavaScript)

The frontend provides a simple, user-friendly interface for inputting text and viewing the sentiment analysis results. Here's how each part works:

- **HTML Structure (index.html)**:
  - The HTML file defines a form with a **textarea** for users to input text, a **submit button**, and a **result section** to display the sentiment analysis.
  - When the form is submitted, JavaScript intercepts the submission and sends the input to the backend for analysis.
- **CSS Styling (style.css)**:
  - CSS provides basic styling to make the interface clean and visually appealing.
  - The .container centers the form on the page and adds a background with padding and shadows for a polished look.
  - The form and button have basic styles for usability, and sentiment results are color-coded (green for positive, red for negative) to make them easily distinguishable.
- **JavaScript for Form Submission**:
  - The JavaScript code captures the form submission, preventing the default action so it can handle submission through JavaScript.

    - It reads the user’s text input and sends it to the backend API using **fetch** with a POST request.

2. Backend API (Flask and Hugging Face Transformers)

The backend API, created using Flask and the Hugging Face transformers library, performs the sentiment analysis on the user’s input text.

- **Flask Setup (app.py)**:
    - Flask provides a lightweight framework to create a REST API that can accept requests and respond with results.
    - The route /analyze accepts POST requests containing the user’s input text in JSON format.
- **Sentiment Analysis Model**:
    - Inside the /analyze route, the input text is passed to a **sentiment analysis model** provided by Hugging Face (e.g., distilbert-base-uncased-finetuned-sst-2-english), which analyzes whether the text is positive or negative.
    - The result from the model includes a label (sentiment: "POSITIVE" or "NEGATIVE") and a score (confidence level).
- **JSON Response**:
    - After analysis, the backend returns a JSON response to the frontend containing the label and score.

3. Bringing It All Together

When a user enters text on the website and submits the form:

1. **JavaScript** captures the input and sends it to the backend API via an AJAX request (using fetch).

2. The **backend API** receives the text, processes it with the sentiment analysis model, and returns a result.
3. The **JavaScript** then displays this result on the webpage, showing the sentiment (positive or negative) and the model's confidence score.

Example Interaction

- **User Action**: The user types, "I love this product, it's amazing!" and clicks the **Analyze Sentiment** button.
- **JavaScript Submission**: The text is sent to the Flask API.
- **Backend Processing**: The API processes the text and finds it to be positive with a high confidence score.
- **Display Result**: The result, e.g., "Sentiment: POSITIVE, Confidence Score: 0.98", is displayed in the result section, color-coded in green.