### **Sentiment Analysis Plugin Documentation**

### **1. Setup and Prerequisites**

Ensure you have the necessary tools and environment set up:

1. Python
2. Node.js and npm (for building the web extension)
3. Libraries: transformers, torch, scikit-learn, pandas, flask (for the backend)
4. Web extension development knowledge

### **2. Data Preparation**

Prepare your custom dataset for training the sentiment analysis model.

1. Collect Data:
   * Gather text data with sentiment labels (positive, negative, neutral).
   * Ensure the data is in a structured format (e.g., CSV or JSON).
2. Clean and Preprocess Data:
   * Clean the text data (remove special characters, normalize case, etc.).
   * Tokenize and preprocess the text for model training.

### **3. Train the Sentiment Analysis Model**

1. Install Necessary Libraries:  
   pip install transformers torch scikit-learn pandas
2. Train and Save the Model

### **4. Build the Backend API**

1. Setting up the Flask App: app = Flask(\_\_name\_\_)
2. Defining the Sentiment Analysis API Endpoint @app.route('/api/analyze', methods=['POST']) def analyze():
3. Handling the Input Data
4. Processing the Model’s Output
5. Running the Flask Application

### **5. Develop the Web Extension**

1. Create manifest.json: Defines the extension's name, version, permissions, and scripts.
2. Design UI: Build the popup interface using HTML, CSS, and JavaScript.
3. Develop Background Script: Optional script for handling background tasks.
4. Inject Content Scripts: Scripts to interact with web page content.
5. Set Permissions: Specify necessary permissions in the manifest.
6. Test and Debug: Load as an unpacked extension in Chrome for testing.
7. Package and Publish: Package the extension and submit it to the Chrome Web Store.