The Customer Emotion Analysis System

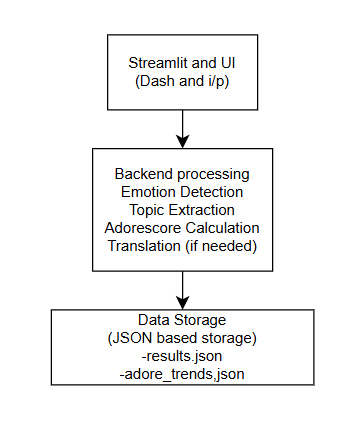
Overview

The customer emotion analysis system is an effective tool which processes customer feedback to basically do three operations.

1. Extract emotions
2. Identify topics and sub-topics
3. Calculate an engagement score

System Architecture

The **frontend** is done with **Streamlit & Plotly**, which provide an interactive dashboard. The **backend** is processed with **NLTK & Transformers & KeyBERT & BERTopic** which process all the emotion detection, topic extraction, translation and storing. The **data storage layer** is composed of **JSON- based storage** which allows historical trend analysis.



Core Features

1. Sentiment Analysis

Detects primary, secondary, and tertiary emotions in customer reviews and uses NLP models to analyze text sentiment.

1. Topic Extraction

Extracts main topics and subtopics from reviews. Calculates topic relevance scores using BERTopic and KeyBERT. Provides Top 5 most relevant topics in dataset.

1. AdoreScore Calculation

Computes AdoreScore (0-100) for customer satisfaction. Uses VADER Sentiment Analysis + Topic Relevance Weighting. Stores historical AdoreScores for trend comparison. Determines whether sentiment is Increasing, Decreasing, or Stable.

1. Multilingual Support

Accepts reviews in any language. Automatically translates non-English reviews to English using Transformer-based AI model (small100).

1. AdoreScore Trends

Line graph tracking customer satisfaction over time.

1. Emotion Distribution Radar Chart

Visualizes intensity of various emotions detected.

1. Top Topics Bar Chart

Displays most discussed topics in dataset.

1. CSV File Upload

Supports bulk analysis of customer reviews. Reads line-by-line and analyzes each review. Stores results in JSON format for download.

1. Real-Time Processing

As soon as a dataset is uploaded, it: Translates reviews (if necessary). Detects emotions. Extracts topics. Computes AdoreScore. Saves everything in results.json.

1. Downloadable Reports

Users can download the full JSON analysis of uploaded dataset. JSON output contains: Original Text, Translated Text, Emotions & Intensities, Topics & Relevance Scores & AdoreScore & Trend.

1. User Input Analysis

Users can enter any review manually.

* The system:
  + Translates (if needed).
  + Detects emotions.
  + Extracts topics.
  + Computes AdoreScore.
  + Displays JSON output on the dashboard.

1. Appending to Results

Custom analysis results are automatically stored in results.json. Allows dynamic updates as more reviews are analyzed.

The process is divided into three key modules:

**Input Module (User Interaction)**

This is where the user provides input, which can be in two forms:

* **User Input** → Manually entering a review.
* **Dataset Upload** → Uploading a CSV file containing multiple reviews.

Both inputs are then sent to the **Processing Module**.

**Processing Module (Core Analysis)**

This module handles the **data processing** and consists of multiple steps:

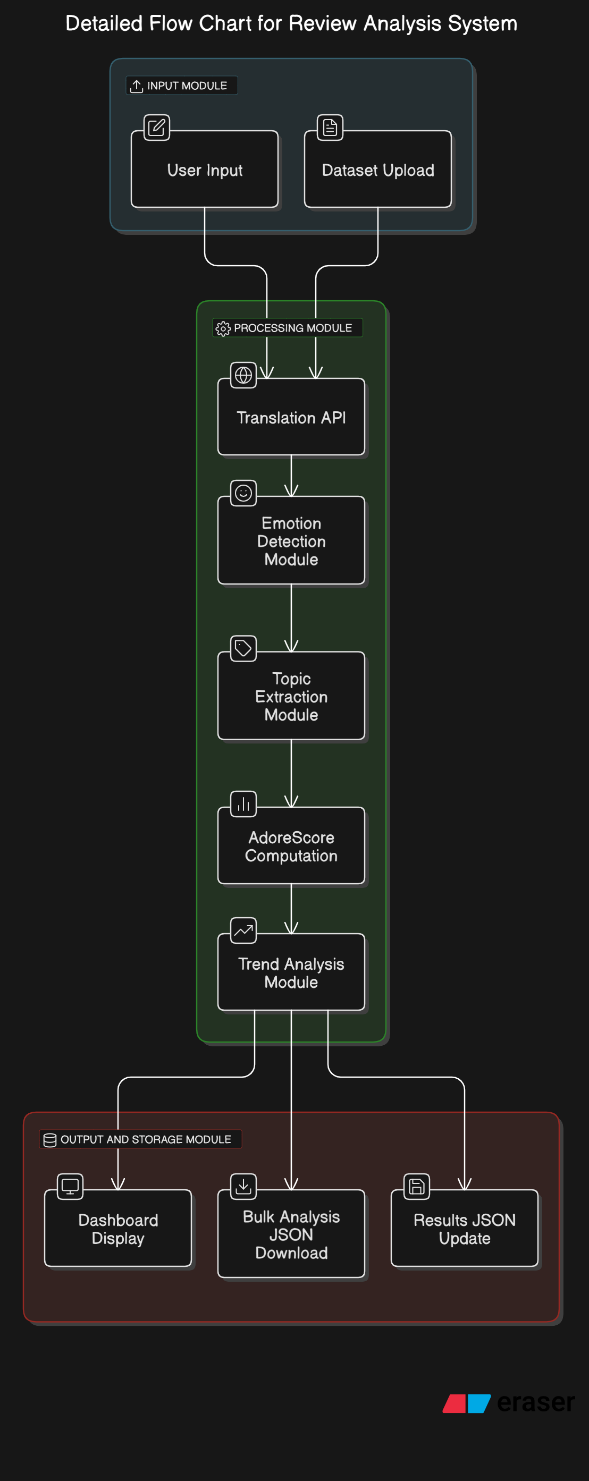
1. **Translation API**
   * If the input text is not in English, it is translated into English using an external translation model.
2. **Emotion Detection Module**
   * Identifies emotions from the review text (e.g., positive, negative, neutral, and specific emotions like joy, anger, or sadness).
3. **Topic Extraction Module**
   * Extracts main topics and subtopics from the text to understand the key themes discussed.
4. **AdoreScore Computation**
   * Calculates an overall **AdoreScore** based on sentiment analysis, topic weighting, and emotion strength.
5. **Trend Analysis Module**
   * Compares historical and recent **AdoreScores** to identify trends and fluctuations in sentiment.

Once processing is complete, the results are sent to the **Output and Storage Module**.

**Output and Storage Module (Results & Visualization)**

This module manages how processed data is displayed and stored:

* **Dashboard Display** → Visualizes insights such as emotion trends, topic distributions, and sentiment scores.
* **Bulk Analysis JSON Download** → Allows users to download the analyzed results in JSON format.
* **Results JSON Update** → Stores processed reviews in a structured JSON file for future use.

.

Challenges Faced & Solutions

First, I had actually tried to implement this project using hard code and basic functions, but they all didn’t have enough features to extract the output based on the expected one. Now the models were chosen from HuggingFace after careful considerations.

The problems which I had faced while developing this project by using pretrained models are as given as follows with the solution which I did.

1. Handling Multilingual Input (Translation Not Working)

The translation model (small100) sometimes returned incorrect translations or did not translate to English as expected.  
Solution: Integrated translate\_to\_english() function to ensure consistent translation of non-English texts.

1. AdoreScore Trend Not Working

The trend calculation wasn’t updating properly due to lack of timestamps in the dataset.  
Solution: Instead of timestamps, we sorted reviews from oldest to newest and compared past vs. latest AdoreScores.

1. Topics Not Being Displayed in Dashboard

The topics section sometimes showed "No valid topics found", even when topics existed.  
Solution: Adjusted the topic extraction logic to correctly parse "main" and "subtopics" from JSON data.

1. Incorrect Topic Analysis & Visualization

The pie chart for topic distribution was not providing a clear representation.

Solution: Replaced pie chart with a heatmap & bar chart for better insights into topic relevance.

1. Emotion Filtering Not Updating Table

When selecting an emotion from the filter, the table did not update dynamically.

Solution: Fixed the callback function so that the table refreshes based on selected emotion.

1. Spider Plot (Radar Chart) Display Issue

The spider plot was either:

* Not filling the area (looked like just a line graph).
* Not retaining its shape when filtering emotions.

Solution:

* Used filled shaded areas to create better-looking spider plots.
* Ensured the graph remains unchanged when filtering emotions.

1. Bulk Processing Output was Empty

The downloaded JSON file (bulk\_analysis.json) was empty, containing only a title.  
Solution: Modified logic to store analyzed results properly before writing to JSON file.

1. Missing Requirements in requirements.txt

Some packages like dash-bootstrap-components, plotly, and transformers were missing.  
Solution: Updated requirements.txt with all necessary dependencies.

9. AdoreScore Not Calculating for Some Reviews

Some reviews returned an error while computing AdoreScore.  
Solution:

* Implemented error handling in compute\_adore\_score().
* Used VADER sentiment analysis + topic weighting for better accuracy.

10. Performance Issues with Large Datasets

When uploading a large dataset, processing became slow.

Solution: Used multi-threading & JSON storage to speed up analysis and reduce memory usage.

11. Dashboard Styling & Layout Issues

* Tables not aligned properly.
* Spider plot overlapping other elements.

Solution:

* Improved alignment & spacing using Streamlit columns.
* Fixed table layout to show scrollable 10-row format.

12. Emotion Radar Plot Not Updating Properly

* The plot was static and did not update with new data.  
   Solution:
* Ensured the radar plot updates dynamically while keeping its overall shape.

13. File Uploads Not Processing Correctly

* Bulk CSV file processing was failing due to missing ‘review’ column.  
   Solution:
* Added validation to check for the correct dataset format before processing.

14. Custom Review Analysis Not Saving to results.json

Custom review results were not getting stored in the main JSON file.  
 Solution: Modified code to append custom analysis results to results.json dynamically.

15. AdoreScore Trend Comparison for Categories Not Working

AdoreScore trends were not showing a category-wise breakdown.  
 Solution: Added category-based filtering to compare AdoreScores dynamically.