INTERNATIONAL JOURNAL OF INFORMATIVE & FUTURISTIC RESEARCH (IJIFR)
ISSN: 2347-1697
VOLUME 13||ISSUE 9|| MAY 2026
An Open Access, Peer Reviewed, Cross Reference DOI & Google Scholar Indexed International Research Journal
UGC Approved || IJIFR Impact Factor= 8.057
Estd Year: 2013 || Email: editors@ijifr.org
Availaible Online at: https://www.ijifr.org/

DOI: https://doi.org/10.64672/IJIFR/26.05.13.09.040 PUBLISHED ON: MAY 30, 2026

AN AI-POWERED CUSTOMER SENTIMENT ANALYSIS PLATFORM INTEGRATING VADER AND RULE-BASED EMOTION DETECTION

Matoori Raja Sree [1] , B. Shireesha [2] , S. Usharani [3]

[1] Student, Department of Computer Applications, Viswam Engineering College, Andhra Pradesh, India
[2]Assistant Professor, [3] Professor,
[2, 3] Department of Computer Applications, Viswam Engineering College, Madanapalle, Andhra Pradesh

ABSTRACT

The proliferation of digital commerce has generated unprecedented volumes of customer feedback that cannot be manually processed at scale. This paper presents SentimentAI, a comprehensive AI-powered customer sentiment analysis dashboard that automates the acquisition, processing, classification, and visual representation of customer feedback in real time. The system implements a hybrid natural language processing pipeline combining VADER (Valence Aware Dictionary and sEntiment Reasoner) for sentiment polarity classification, a rule-based emotion detection module supporting five emotional categories, TextBlob-powered keyword extraction, and an automated contextual response generator. Built using Python, Django 4.2, and SQLite3, the application follows the Model-View-Template architecture and supports two user roles: customers who submit feedback through a simulated e-commerce storefront, and administrators who access an intelligence dashboard with sentiment distribution charts, keyword frequency analysis, negative feedback alerts, and AI-generated business insights. The system is deployed on Vercel cloud platform, demonstrating production readiness. Experimental evaluation confirms that the hybrid approach accurately classifies sentiment polarity and generates contextually appropriate automated responses, transforming unstructured textual feedback into structured, actionable business intelligence.

KEYWORDS *Sentiment Analysis; VADER; Emotion Detection; Natural Language Processing; Django; Customer Feedback; Automated Response Generation; Dashboard Analytics*

Recommended Citation:
Sree, M R, Shireesha, B, Usharani, S : "An AI-Powered Customer Sentiment Analysis Platform Integrating VADER And Rule-Based Emotion Detection", International Journal of Informative & Futuristic Research (IJIFR), Vol. (13) (9), May 2026, pp. 1655-1661. https://doi.org/10.64672/IJIFR/26.05.13.09.040

1. INTRODUCTION

The modern digital economy generates millions of customer reviews daily through product platforms, social media, and e-commerce portals. Manually processing this unstructured textual data to extract actionable intelligence is impractical for organizations operating in competitive markets. A mid-sized e-commerce platform may receive thousands of reviews per day, demanding prohibitive staffing resources while introducing inconsistency into classification.Sentiment analysis, a branch of natural language processing, applies computational methods to identify and quantify emotional tone expressed within text. In e-commerce, a negative review may indicate product defects, logistics failures, or service

breakdowns, and the ability to detect these root causes enables targeted corrective action. Swift automated response can de-escalate complaints and restore brand confidence before human intervention.

SentimentAI addresses this need by providing a complete, deployable system that automates sentiment analysis of customer reviews through a multi-stage NLP pipeline determining sentiment polarity, identifying dominant emotion, extracting keywords, and generating contextually appropriate automated responses. The system is implemented as a full-stack Django web application using VADER and TextBlob for NLP processing, deployed on Vercel cloud platform at a publicly accessible endpoint.

2. LITERATURE SURVEY

Hutto and Gilbert [1] introduced VADER, a parsimonious rule-based model specifically tuned for social media text, incorporating heuristic rules for capitalization, punctuation, and emphasis that improve accuracy over general-purpose lexicon systems. Liu [2] provided a comprehensive treatment of sentiment analysis and opinion mining, establishing the theoretical foundations for polarity classification. Pang and Lee [3] surveyed opinion mining approaches, documenting the evolution from keyword-based detectors to machine learning classifiers.

Vaswani et al. [6] introduced the transformer architecture underpinning modern NLP systems, while Devlin et al. [7] demonstrated BERT's superior performance on sentiment tasks through bidirectional pre-training. Pontiki et al. [13] introduced aspect-based sentiment analysis for fine-grained attribute-level classification. Existing solutions range from manual review management, which lacks scalability, to commercial cloud APIs requiring expensive integration and lacking automated response generation. SentimentAI bridges this gap by providing an integrated, open-source, self-hostable platform combining real-time analysis with automated responses and administrative analytics.

3. SYSTEM PROPOSAL

3.1 Problem Statement

E-commerce businesses face growing inability to process customer feedback at the speed and scale demanded by digital commerce. Manual processing is constrained by limited throughput, inconsistent classification, response latency, and linearly scaling costs. Existing automated tools are proprietary, expensive, provide batch-processing rather than real-time analysis, lack automated response generation, and offer insufficient customizability for domain-specific vocabulary.

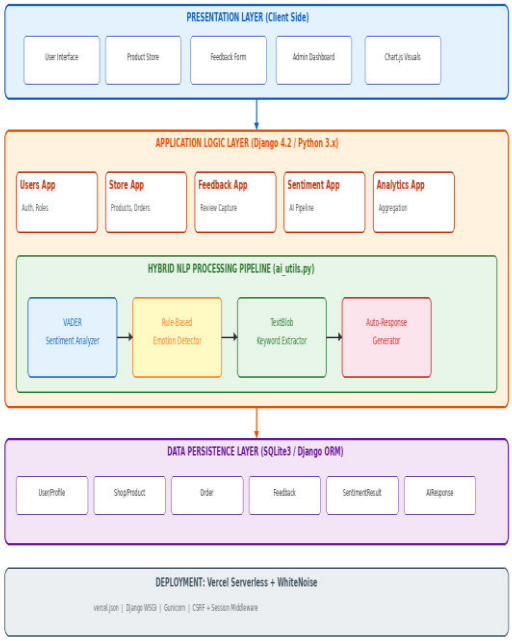

Figure 1: System Architecture of SentimentAI

3.2 Proposed System

SentimentAI provides an integrated platform combining real-time NLP analysis, automated response generation, persistent storage, and visual analytics within a single Django application. Five Django apps handle Users (authentication and roles), Store (product catalogue simulation), Feedback (review capture), Sentiment (AI pipeline and results storage), and Analytics (data aggregation and reporting). The VADER analyzer classifies compound polarity scores using ±0.05 thresholds, while a five-category rule-based emotion detector, TextBlob keyword extractor, and template-based response generator complete the hybrid pipeline.

3.3 System Architecture

The system follows a layered three-tier architecture separating presentation, application logic, and data persistence. The Presentation Layer uses Django's Jinja2 templates styled with Bootstrap 5 and Chart.js visualizations. The Application Logic Layer implements five Django apps with the AI pipeline in ai_utils.py invoked synchronously during feedback submission. The Data Persistence Layer uses SQLite3 through Django's ORM. Fig. 1 illustrates the complete architecture.

3.4 System Flow

The feedback submission workflow begins when a registered user places an order and submits a review with star rating. The browser sends an HTTP POST to /feedback/api/submit/. Django validates fields, retrieves the Order, creates a Feedback record, then invokes the AI pipeline: VADER returns polarity label and compound score, emotion detection identifies the dominant emotional state, TextBlob extracts noun phrase keywords, and the response generator produces a contextual reply. Results are persisted as SentimentResult and AIResponse records. The administrator analytics workflow aggregates data through /analytics/data/, serving JSON consumed by Chart.js for dashboard rendering. Fig. 2 depicts both workflows.

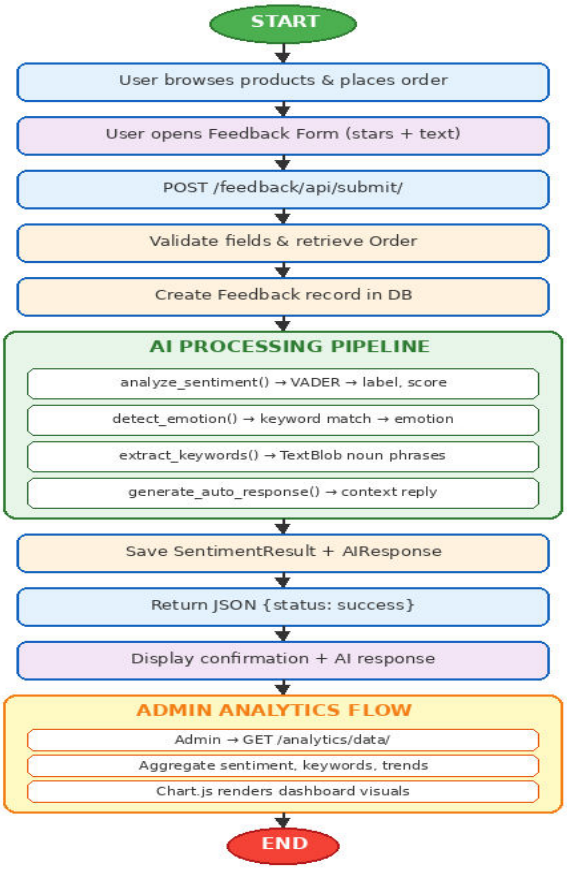

Figure 2: System Flow Diagram of SentimentAI

3.5 UML Diagrams

3.5.1 Use Case Diagram

Two primary actors interact with the system: Customers (register, browse, order, submit feedback, view AI responses) and Administrators (dashboard access, sentiment charts, keyword analysis, alerts, feedback management, AI responses, insights, and settings). The AI Pipeline acts as a system actor invoked during feedback submission. Fig. 3 presents the use case model.

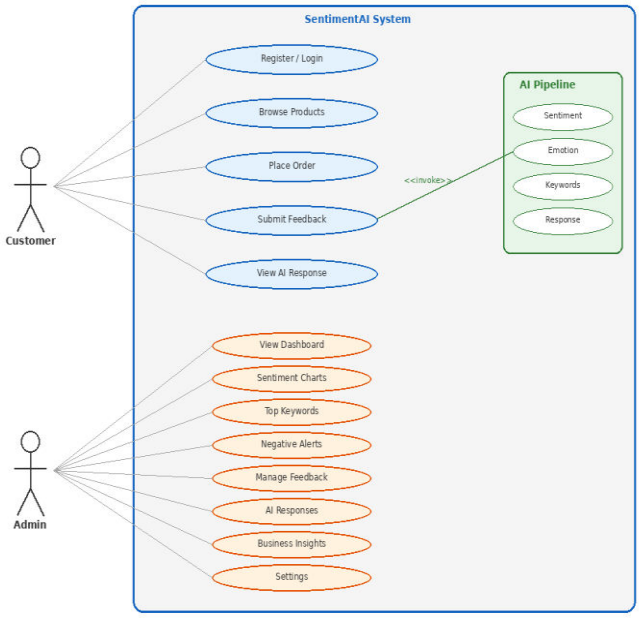

Figure 3: Use Case Diagram of SentimentAI

3.5.2 Sequence Diagram

The sequence diagram traces interactions between Browser, URL Dispatcher, FeedbackView, OrderModel, AI Pipeline, and Database during feedback submission. The flow covers POST request routing, order retrieval, feedback creation, sequential AI pipeline invocations (sentiment, emotion, keywords, response generation), result persistence, and JSON response delivery. Fig. 4 illustrates the complete interaction sequence.

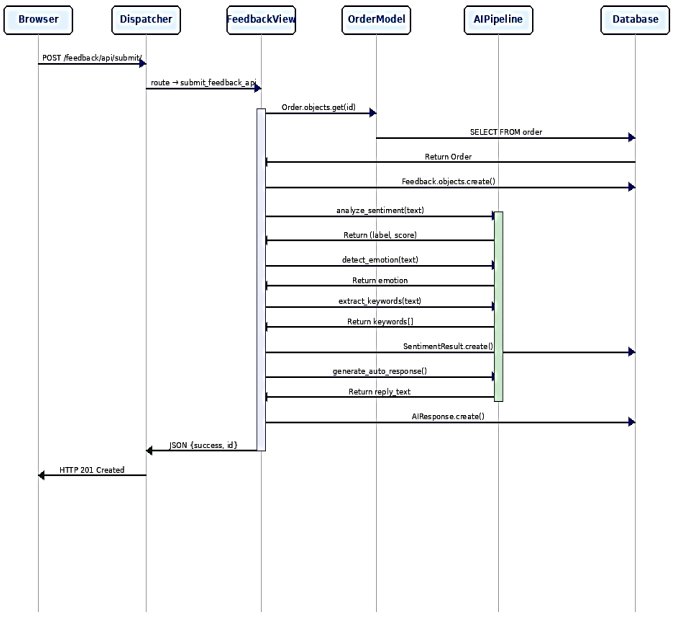

Figure 4: Sequence Diagram of SentimentAI

4. IMPLEMENTATION

4.1 Sentiment Analysis Pipeline

The analyze_sentiment() function instantiates a module-level cached VADER SentimentIntensityAnalyzer, which processes text by matching words against its 7,500-entry lexicon, applying heuristic adjustments for ALL CAPS amplification, exclamation punctuation, and valence shifters ("very," "not"). The compound score (−1 to +1) is thresholded at ±0.05 for three-way classification.

4.2 Emotion Detection Module

The detect_emotion() function implements keyword-based classification mapping five categories (Happy, Satisfied, Angry, Frustrated, Confused) to lexical trigger sets. The function iterates categories in priority order, returning the first match or Neutral as default. Each category includes emoji indicators for visual differentiation in the interface.

4.3 Keyword Extraction Module

The extract_keywords() function uses TextBlob's noun phrase chunker, which applies pattern-based part-of-speech tagging to identify multi-word noun phrase boundaries. Extracted phrases are deduplicated via Python's set and limited to five unique phrases, stored as JSON arrays in SentimentResult records for aggregation in the analytics dashboard.

4.4 Automated Response Generator

The generate_auto_response() function accepts sentiment label, emotion, rating, and review text, selecting from sentiment-categorized template libraries with three variants per category. Templates embed emotion and rating via f-string interpolation. Context-specific additions are appended when delivery or quality issues are detected in the text, following customer service best practices of acknowledgment, empathy, and commitment to action.

4.5 Analytics Dashboard

The dashboard_data_api view aggregates seven metric categories: total feedback count, sentiment distribution via Django ORM annotations, keyword frequency using Python's Counter, recent negative alerts filtered by severity threshold, daily volume trends, and AI-generated business insights. All data is serialized as JSON and consumed by Chart.js for asynchronous dashboard rendering without page reloads.

5. RESULTS AND DISCUSSION

5.1 Sentiment Classification

VADER's compound scoring mechanism demonstrates strong performance on informal, emotionally expressive product review text. Clear positive reviews ("fantastic product, exceeded expectations") consistently receive compound scores above +0.5, while clear negative reviews ("terrible quality, waste of money") score below −0.5. Ambiguous or mixed-sentiment reviews are correctly classified as Neutral when compound scores fall within the ±0.05 threshold range. Fig. 5 visualizes the classification thresholds and emotion detection categories.

Figure 5: VADER Classification Thresholds and Emotion Detection Framework

5.2 Response Generation Quality

The template-based response generator produces contextually relevant replies that reference the customer's specific rating, emotional state, and identified issues. Negative reviews mentioning delivery problems receive apology templates with delivery improvement commitments. While less linguistically rich than generative AI responses, the template approach ensures consistency, predictability, and zero-latency generation suitable for real-time deployment.

5.3 System Performance

The complete AI pipeline executes in sub-second time on standard hardware, with VADER analysis completing in under 10 milliseconds. The lightweight NLP stack (VADER lexicon ~50MB, TextBlob models ~100MB) enables deployment on commodity hardware without GPU requirements. Cross-browser testing confirms correct rendering across Chrome, Firefox, Edge, and Safari. The Vercel deployment validates production readiness in a real network environment.

6. CONCLUSION

SentimentAI demonstrates the successful integration of hybrid NLP techniques within a full-stack web application to create a production-ready business intelligence tool. The combination of VADER sentiment analysis, rule-based emotion detection, TextBlob keyword extraction, and automated response generation provides multi-dimensional analysis of customer reviews in real time. The administrative dashboard transforms unstructured feedback into structured, visual intelligence through sentiment distribution charts, keyword frequency analysis, and negative feedback alerts. The system's deployment on Vercel validates technical completeness, while its open-source technology stack eliminates licensing costs and ensures full transparency. SentimentAI represents a practical, maintainable solution for small and medium-sized e-commerce operators seeking to automate customer feedback analysis and response generation.

7. FUTURE ENHANCEMENTS

Key enhancement directions include integration of transformer-based models (BERT, RoBERTa) for superior sentiment classification accuracy on complex reviews; aspect-based sentiment analysis to decompose reviews into attribute-level sentiment scores for targeted improvement; multilingual support via XLM-RoBERTa or translation APIs; generative AI response engines using large language model APIs for richer, more personalized responses; migration from SQLite3 to PostgreSQL for concurrent production deployments; and real-time streaming analytics via Django Channels WebSocket support for continuous dashboard updates.

8. REFERENCES

[1] C. J. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," *Proc. ICWSM*, AAAI Press, pp. 216–225, 2014.
[2] B. Liu, "Sentiment Analysis and Opinion Mining," *Synthesis Lectures on HLT*, vol. 5, no. 1, pp. 1–167, 2012.
[3] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in IR*, vol. 2, pp. 1–135, 2008.
[4] Django Software Foundation, "Django Documentation v4.2," *docs.djangoproject.com*, 2023.
[5] S. Loria, "TextBlob Documentation," *textblob.readthedocs.io*, 2020.
[6] A. Vaswani et al., "Attention Is All You Need," *Advances in NIPS*, vol. 30, pp. 5998–6008, 2017.
[7] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers," *Proc. NAACL*, pp. 4171–4186, 2019.
[8] T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997.
[9] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, O'Reilly Media, 2009.

[10] Bootstrap Team, "Bootstrap 5 Documentation," *getbootstrap.com*, 2023.
[11] T. Nasukawa and J. Yi, "Sentiment Analysis: Capturing Favorability Using NLP," *Proc. K-CAP*, pp. 70–77, 2003.
[12] Vercel Inc., "Vercel Documentation: Python Applications," *vercel.com/docs*, 2023.
[13] M. Pontiki et al., "SemEval-2014 Task 4: Aspect Based Sentiment Analysis," *Proc. SemEval*, pp. 27–35, 2014.