**Sentiment Analysis Dashboard Documentation**



|  |  |
| --- | --- |
| **Name** | **Surname** |
| **Tebelelo** | **Lekoana** |
| **Hartton** | **Malau** |
| **Kagiso** | **Monene** |
| **Babyface** | **Mokoena** |
| **Morokolo** | **Chueu** |
| **Nelisiwe** | **Kaziwa** |

**Documentation for Sentiment Analysis Dashboard**

**Introduction**

This documentation outlines the development and functionality of the Sentiment Analysis Dashboard, a web-based application designed to analyze textual data for sentiment insights. Built using modern web technologies and integrated with advanced AI APIs, the dashboard enables users to upload documents in various formats (.txt, .docx, .pdf), input raw text, and receive detailed sentiment analysis outputs, including sentiment classification, confidence scores, keyword highlights, and exportable reports. The project aims to streamline sentiment analysis for applications such as customer feedback review, market research, and content moderation. This document justifies the selected APIs and libraries, discusses implementation challenges, provides a user guide with practical examples, and illustrates system use cases to demonstrate real-world applicability.

**1. API Selection Justification**

The project integrates several external APIs and libraries:  
  
1.1 Gemini API (Google Generative AI)  
 • Provides access to Google’s Generative AI APIs for natural language tasks.  
 • Chosen because it offers state-of-the-art sentiment and text analysis without needing to  
 build ML models from scratch.  
 • Justification: Ensures accurate and scalable NLP, faster prototyping, and reduces the need  
 for manual training pipelines.  
  
1.2 Jspdf & jspdf-autotable  
 • Used to export analysis results (charts, tables, summaries) into PDF format.  
 • Justification: Enables easy report generation for users who need downloadable insights.  
  
1.3 Mammoth  
 • Converts .docx files into clean HTML.  
 • Justification: Allows importing of textual data from Word documents for analysis.  
  
1.4 Pdfjs-dist  
 • Parses PDF files so their text can be extracted and analyzed.  
 • Justification: Many organizations store reports in PDF; this ensures compatibility.  
  
1.5 React + Tailwind CSS  
 • React provides modular, component-based UI development.  
 • Tailwind ensures responsive, modern styling with minimal custom CSS.  
 • Justification: Enables fast development, scalability, and maintainability.

**2. Implementation Challenges**

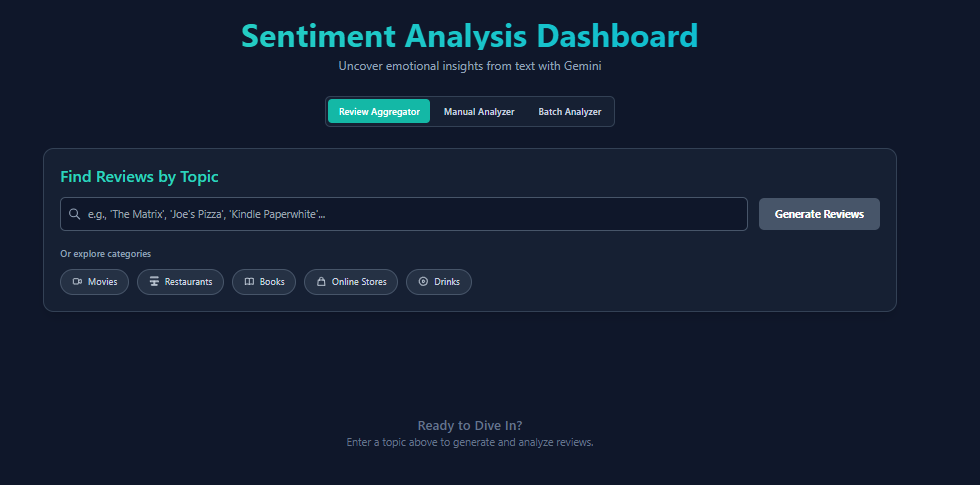
Several challenges were encountered during the implementation of the dashboard:  
  
2.1. API Integration Complexity  
 The Gemini API (Google Generative AI) API requires proper authentication and handling of async requests.   
 *Challenge:* Managing rate limits and ensuring error handling for failed API calls.  
  
2.2. File Handling  
 Extracting clean text from .docx and .pdf files was non-trivial:  
 *Challenge*: Preserving formatting and avoiding noisy characters.  
 *Solution:* Used mammoth for .docx and pdfjs-dist for .pdf.

2.3. UI Performance  
 Rendering large documents and long sentiment outputs slowed the dashboard.  
 *Solution*: Implemented lazy rendering and optimized state management in React.  
  
2.4. Export Functionality  
 Generating structured PDFs with jspdf-autotable required careful data formatting.  
 *Challenge*: Ensuring charts/tables fit neatly into exports.  
  
2.5. Styling Consistency  
 Tailwind utilities are powerful but can lead to cluttered code.  
 *Solutio*n: Introduced custom classes and reusable components for readability.

**3. User Guide with Examples:**

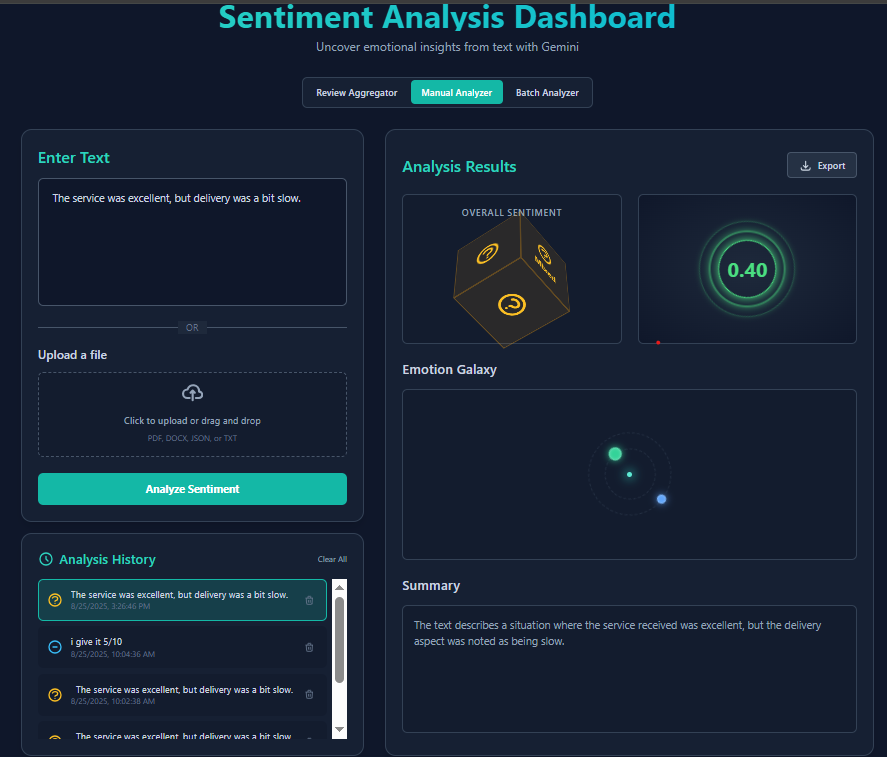
**Example 1: Getting Started**

3.1. Open the dashboard in your browser

  
3.2. Upload a file (.txt, .docx, or .pdf) or paste raw text into the input field.  
3.3. Click 'Analyze Sentiment'.

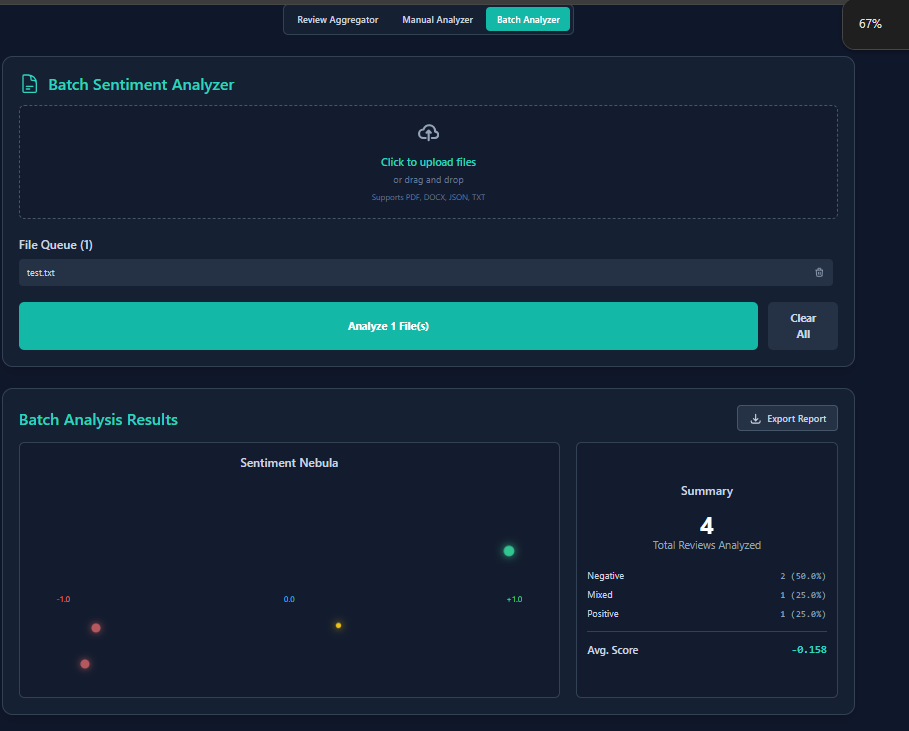
**Example 2: Paste Text**

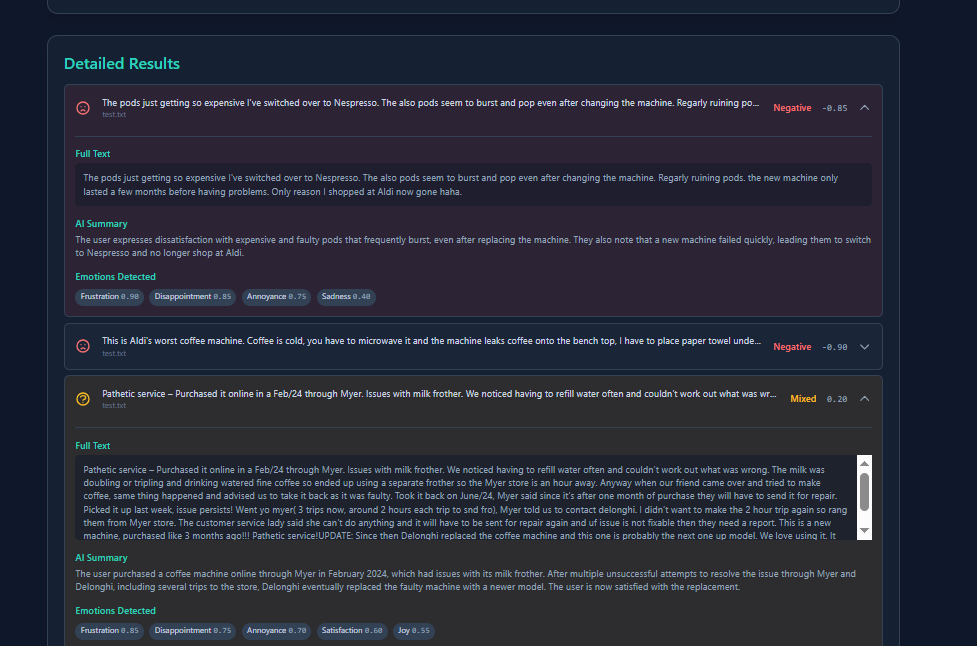
Input:  
 The service was excellent, but delivery was a bit slow.  
Output:  
 - Sentiment: Mixed (Positive service, Negative delivery)  
 - Confidence: 0.40



**Example 3: Upload a PDF Report**

1. Upload a customer feedback PDF.

  
2. Dashboard extracts text runs it through Gemini API (Google Generative AI) sentiment model, and displays:  
 - Overall sentiment distribution (positive/negative/neutral).  
 - Keyword highlights.  
 - Exportable summary in PDF.

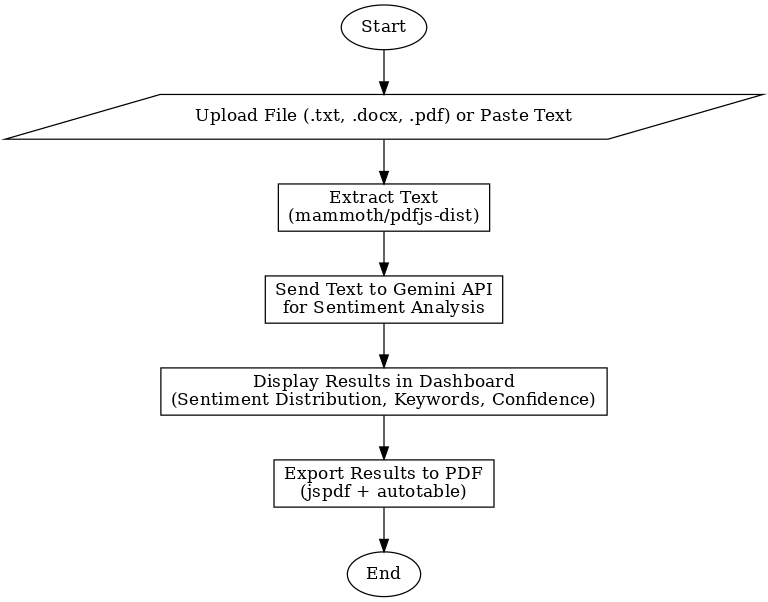


**Example 4: Export Results**

After analysis, click 'Export → PDF'.  
A formatted PDF with tables and charts is generated (via jspdf + jspdf-autotable).



**System User case**



**Conclusion**

In conclusion, the Sentiment Analysis Dashboard represents a robust solution for efficient text-based sentiment analysis, leveraging cutting-edge APIs like Gemini for accurate NLP processing and user-friendly libraries for seamless file handling and reporting. While challenges in API integration, file parsing, and performance were addressed through targeted solutions, the final application delivers scalable, maintainable, and intuitive functionality. This project not only facilitates quick insights from diverse data sources but also sets a foundation for future enhancements, such as real-time analysis, multi-language support, or integration with additional AI models. Overall, it empowers users to make data-driven decisions in sentiment-sensitive domains.