14 – T1 – Sentiment Analysis

CS 4850 – Spring 2025 – 04/28/2025

Sharon Perry

Website Link: *https://heartspeakai.netlify.app/*
GitHub Link: *https://github.com/KSU-Sentiment-Analysis/sentiment-analysis-Integration*

Stats & Status:

LOC	4,112
Components/Tools	React, Flask, Azure, OpenAI API, Python, GitHub, Discord
Hours Estimate	353
Hours Actual	233

Aravind Iyer, Nabeel Faridi, Ethan Barnes, Isha Minocha, Shammah Charles

Table of Contents

1. Introduction

This project presents a Sentiment Analysis AI designed for in-depth text review analysis. Leveraging fine-tuned BERT-based models, the system can classify overall sentiment, detect emotions, identify sarcasm, and extract aspect-level opinions. Evaluation results demonstrate strong performance across these tasks, with sentiment classification achieving approximately 69% accuracy. By integrating with OpenAI, the analysis pipeline delivers actionable insights that enable businesses to enhance their products and improve customer satisfaction.

1.1: Goals & Objectives

The project is driven by three core goals, focusing on solving specific challenges in customer interaction management.

1.1.1: Efficient Sentiment Analysis:

- Automate the process of analyzing customer feedback to determine sentiment (positive, negative, neutral) and identify emotional tones.
- Enable advanced capabilities such as sarcasm detection, emotion detection, and aspect-based sentiment analysis for deeper insights.

1.1.2: Context-Aware Response Generation:

- Generate empathetic, personalized responses based on sentimental insights, ensuring timely communication with customers.
- Maintain brand consistency by aligning responses with the company's tone and voice guidelines.

1.1.3: Enhanced Operational Efficiency:

- Reduce the time and resources required for manual sentiment analysis and response generation.
- Provide visual insights and demonstrate use cases through example outputs, such as mock emails, chat transcripts, or social media replies.

2. Requirements/ Design Constrains

2.1: Requirements

2.1.1: Usability Requirements:

- The CMS should provide an intuitive UI for modifying response guidelines.

2.1.2: Operational Requirements

- The system should handle at least 100 concurrent API requests.

2.1.3: Performance Requirements

- Response Generation should take less than 20 seconds for each entry
- Sentiment Analysis should be completed within 10 seconds for each entry.

2.1.4: Security Requirements

- Only authenticated users should access CMS features.
- API calls must use OAuth2 authentication.

2.1.5: Other Requirements

- The system should have **99.9% uptime**.
- All responses must be **auditable** for compliance.

2.2: Design Constraints

2.2.1: Design Constraints

- **Hosting:** Must be deployed on Azure App Services.
- **Back-End:** Must use Azure SQL Database or Cosmos DB.
- **Front-**End: Must be developed using React or Angular.
- **AI Integration:** Uses OpenAI API through Azure OpenAI Services.
- **Security:** Must implement OAuth2 authentication.
- **System:** Will support csv file uploads with a specific format.

2.2.2: User Constraints

- **Customer Service Representatives**: Need AI-generated suggestions for responding to customers.
- **Developers**: Will use the API to integrate with existing systems.
- **Data Analyst**: Will analyze dataset

3. Design

3.1: System Architecture

The system will be hosted entirely on **Azure**, utilizing:

- **Azure App Services** for API deployment.
- **Azure Functions** for AI-based response processing.
- **Azure SQL Database** for CMS storage.

3.1.1: System Software Architecture

Component	Technology Stack
Frontend	React, Power BI
Backend	Python (Flask Framework)
Database	Azure SQL Database (via SQLAlchemy ORM)
AI Services	OpenAI API, Azure Cognitive Services
Deployment	Azure App Services
Version Control	GitHub with CI/CD pipelines via GitHub Actions
Security	Environment variables for credentials

3.1.2: Internal Communications Architecture

- **API to CMS Communication:** RESTful API endpoints.
- **CMS to Database Communication:** Secure SQL queries.
- **Frontend to Backend Communication:** HTTPS with OAuth2 authentication.

3.2: Inputs

- **Customer Feedback:** .csv file containing feedback.
- **User Preferences:** Customization options via CMS dashboard.

3.3: Outputs

- **Generated Responses:** AI-crafted text responses.
- **Sentiment Reports:** Visual sentiment breakdown.
- **Admin Dashboard:** CMS interface for customization.

3.4: Hardware Detailed Design

- **Servers:** Azure cloud infrastructure.

3.5: Software Detailed Design

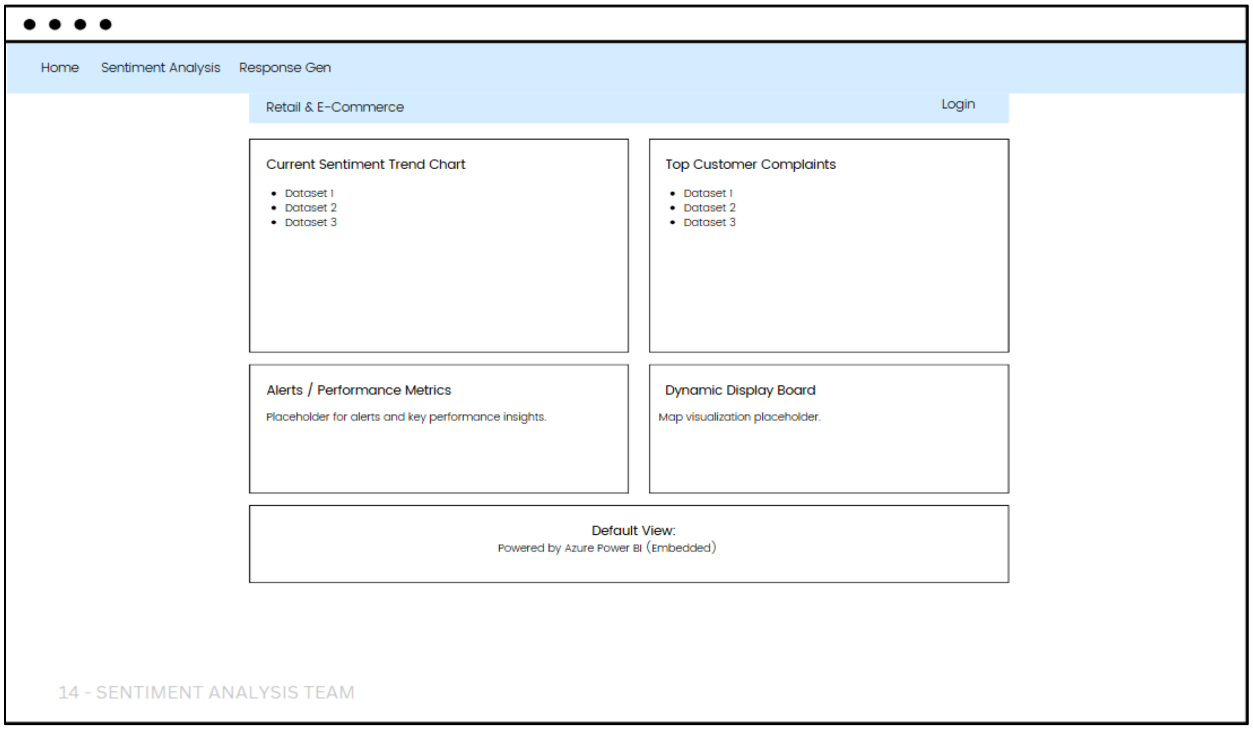
- **AI Model Integration:** Calls OpenAI API for response generation.
- **Authentication:** OAuth2-based token system.
- **Logging & Monitoring:** Azure Log Analytics.

3.6: Design Mock-ups

3.6.1: Homepage

- **Current Sentiment Trend Chart**
  - Displays sentiment trends across multiple datasets.
- **Top Customer Complaints**
  - Highlights common customer concerns categorized by dataset.
- **Alerts & Performance Metrics**
  - Placeholder for alerts and key performance insights.
- **Dynamic Display Board**

  Reserved space for interactive map-based visualizations.

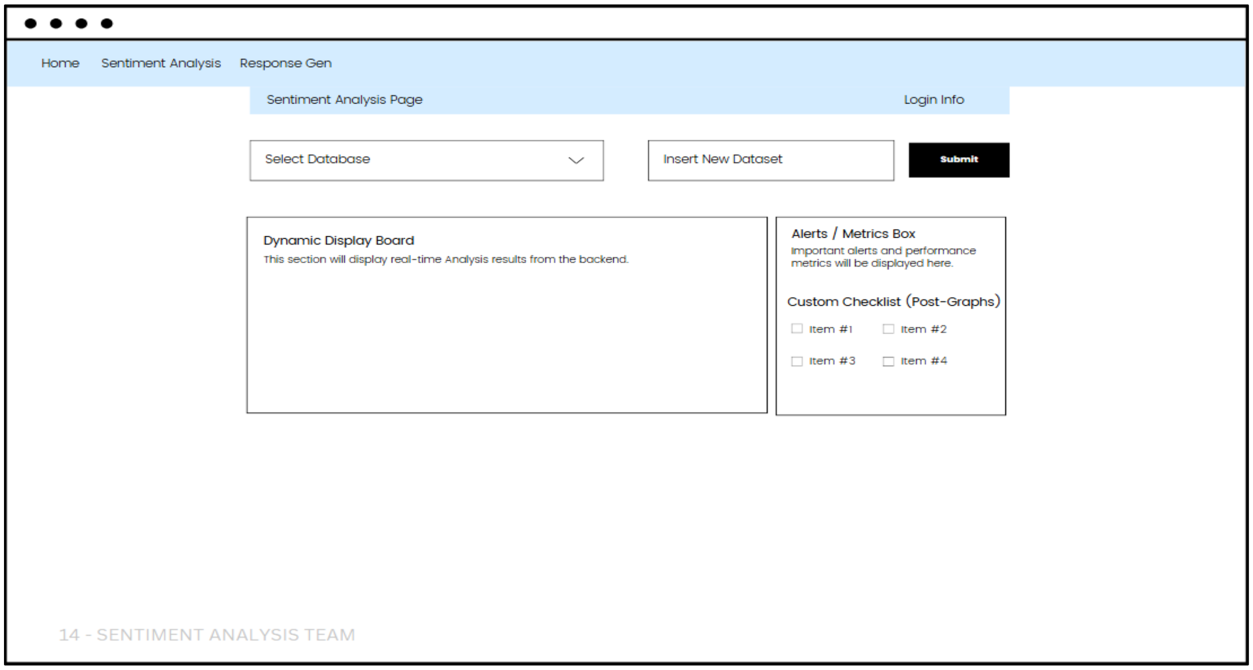

Home
Sentiment Analysis
Response Gen
Retail & E-Commerce
Login
Current Sentiment Trend Chart
Dataset 1
Dataset 2
Dataset 3
Top Customer Complaints
Dataset 1
Dataset 2
Dataset 3
Alerts / Performance Metrics
Placeholder for alerts and key performance insights.
Dynamic Display Board
Map visualization placeholder.
Default View:
Powered by Azure Power BI (Embedded)
14 - SENTIMENT ANALYSIS TEAM

3.6.2: Sentiment Analysis

- **Database Selection & Input**
  - Dropdown to select an existing database.
  - Input field to insert a new dataset with a Submit button.
- **Dynamic Display Board**
  - Real-time analysis results fetched from the backend.
- **Alerts & Metrics Box**
  - Displays important alerts and performance metrics.

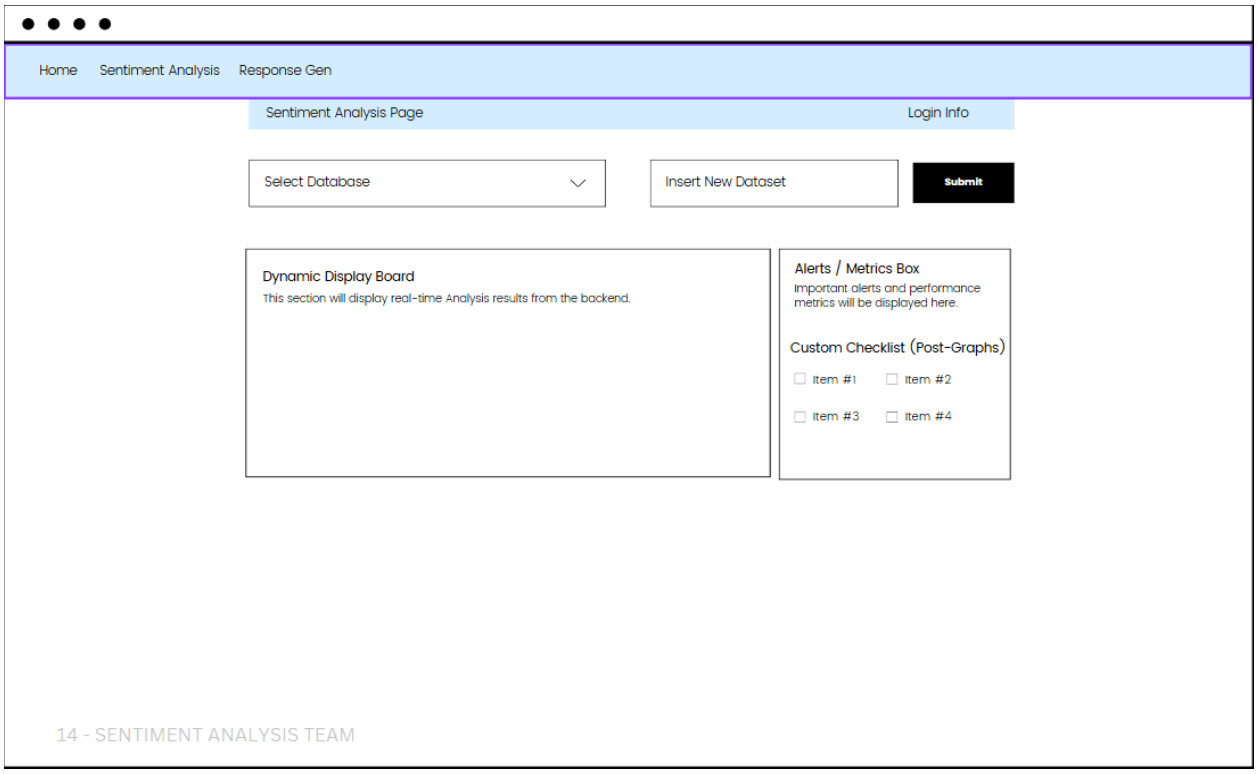
- **Custom Checklist (Post-Graphs)**

  Checklist for validation and task completion after analysis.


Home
Sentiment Analysis
Response Gen
Sentiment Analysis Page
Login Info
Select Database
Insert New Dataset
Submit
Dynamic Display Board
This section will display real-time Analysis results from the backend.
Alerts / Metrics Box
Important alerts and performance metrics will be displayed here.
Custom Checklist (Post-Graphs)
Item #1
Item #2
Item #3
Item #4
14 - SENTIMENT ANALYSIS TEAM

3.6.3: Response Generation

- **User-Friendly Design**
  - Simplified interface designed for ease of use.
- **Database Interaction**
  - Easily select and manage customer interaction databases.
- **AI-Generated Responses**
  - Quickly generate empathetic AI-driven responses to customer inquiries.
- **Metrics & Alerts**
  - Metrics and alerts box for immediate insights into performance.
- **Response Checklist**
  - Integrated checklist to verify response quality and completion.


Home
Sentiment Analysis
Response Gen
Sentiment Analysis Page
Login Info
Select Database
Insert New Dataset
Submit
Dynamic Display Board
This section will display real-time Analysis results from the backend.
Alerts / Metrics Box
Important alerts and performance metrics will be displayed here.
Custom Checklist (Post-Graphs)
Item #1
Item #2
Item #3
Item #4
14 - SENTIMENT ANALYSIS TEAM

4. Development

4.1 Database Configuration

- **Cloud Provider:** Microsoft Azure
- **Subscription Plan:** Azure for Students
- **Resource Group:** SentimentAnalysisRG
- **Region:** East US 2
- **Database Name:** sentiment-analysis-db
- **Server:** sentiment-analysis-sqlserver
- **Authentication Method:** SQL Authentication
- **Server Admin Login:** sentimentadmin
- **Computer & Storage:**
  - Tier: General Purpose - Serverless
  - Series: Standard series (Gen5)
  - vCores: 2
  - Storage Capacity: 32 GB
  - Overage Billing: Disabled

5. Test Plan and Test Report

5.1 Test Plan

The testing strategy focused on validating core functionalities of the HeartSpeak AI platform, including sentiment analysis, AI response generation, and user interface rendering. Both frontend and backend components were tested using manual, unit, and automated techniques.

Frontend Testing (React)

Component	Test Focus
HomePage	Chart rendering, real-time sentiment updates, testimonial rotation
SentimentAnalysis	File upload, prediction output, chart display
ResponseGeneration	Dataset selection, backend integration, output validation
Team Page	Static rendering of team cards

Tests were implemented using **Jest**, **React Testing Library**, and **Cypress** for selected UI flows.

Backend Testing (Flask)

Endpoint	Test Cases
`/upload`	Validates CSV uploads and file handling
`/process`	Confirms correct branching for different model types
`/predict_single`	Returns valid sentiment, emotion, sarcasm, and aspect analysis
`/generate-responses`	Returns structured responses, prompts used, and evaluation scores

5.2 Test Cases

Test Case Description	Pass/ Fail	Severity
Backend		

CSV Upload Endpoint (`/upload`) accepts a valid CSV and returns the file path	P	High
CSV Processing Endpoint (`/process`) correctly processes data and triggers analysis	P	High
Pretrained Sentiment Analysis returns correct predictions	F	Low
Custom/Train Sentiment Analysis trains model and predicts without error	P	Medi um
Advanced Analysis returns outputs for sentiment, emotion, sarcasm, and aspects	P	High
Single Review Prediction (`/predict`) returns a complete JSON response	P	Medi um
File Download Endpoint (`/download`) serves output files correctly	P	High
Error Handling displays clear error messages for missing/incorrect inputs	F	Minor
Frontend		
Home page review slider auto-rotates and buttons work	P	Medi um
Login page validates inputs and submits credentials	F	Medi um
SentimentAnalysis file upload triggers backend and displays preview	P	High
Processed CSV file is downloadable from frontend	P	Medi um
ResponseGeneration adds emails and displays responses in a table	P	Medi um
Dataset dropdown and checklist elements are functional	F	Low
NavBar correctly routes and highlights active tab	P	Medi um
Logo and styling render correctly across all pages	P	Low

5.3 Test Report

Frontend Test Cases Report

Test Case ID	Functionality Tested	Test Steps	Expected Result	Actual Result	Status	Severity
TC-FE-001	Home Page Review Slider	Load page and observe slider behavior	Reviews auto-rotate	All animations and controls worked	Pass	Medium
TC-FE-002	Login Page Input Validation	Submit with/without valid inputs	Fields validated; no submission without input	Fields validated; no submission without input	Fail	Medium
TC-FE-003	SentimentAnalysis File Upload	Upload CSV and click “Upload & Process”	Table preview after upload and backend triggers processing	File uploads and preview table shows	Pass	High
TC-FE-004	Download Processed File	Wait for processing, then click download	File downloads successfully	Download button worked	Pass	Medium
TC-FE-005	ResponseGeneration Dataset Table	Select dataset and submit	Dataset is processed and table displays placeholder	Table rendered without email input functionality	Pass	Medi

						um
TC-FE-006	Checklist and Alerts on Response Page	Scroll on Response page	Checklist and alerts display properly	All items clickable and rendered	Pass	Low
TC-FE-007	NavBar Navigation	Click tabs (Home, Sentiment, etc.)	Active tab highlights and correct page renders	Routing worked as expected	Pass	Medium
TC-FE-008	Logo and Style Verification	Inspect logo and layout	Logo renders cleanly; UI styled correctly	All elements styled as described	Pass	Low

Backend Test Cases Report

Test Case ID	**Functionality Tested**	**Expected Result**	**Actual Result**	**Status**	**Severity**
TC-001	CSV Upload Endpoint (`/upload`)	JSON response with file path	Returned valid JSON with path	Pass	High
TC-002	Pretrained Mode Processing	Output CSV with predicted sentiments	Output CSV generated as expected	Pass	High
TC-003	Custom Mode	CSV with `custom_sentiment_prediction` column	Custom predictions present in output	Pa	Low

				ss	
TC-004	Advanced Mode	CSV with all advanced analysis columns	Output CSV populated with all expected columns	Pass	High
TC-005	Single Review Prediction (`/predict`)	JSON with sentiment, emotion, sarcasm, and aspect fields	JSON returned with all fields	Pass	Medium
TC-006	File Download Endpoint (`/download`)	File downloaded successfully	File downloaded without error	Pass	High
TC-007	`dataset_handler.load_csv` Function	DataFrame returned with expected structure	Columns printed as expected	Pass	High
TC-008	`fuzzy_match_column` for custom headers	Returns best-matched column	Correct fuzzy match returned	Pass	High
TC-009	`process_data` cleans DataFrame	Cleaned and valid review/rating columns returned	Data cleaned as expected	Pass	High
TC-010	`run_deep_sentiment_analysis`	Training completes, output CSV + metrics generated	Training and file generation successful	Pass	High
TC-011	`run_inference` (sentiment)	Output with predicted sentiment	Correct predictions returned	Pass	High

TC-012	`run_deep_emotion_analysis`	Emotion training completes, output files generated	Output generated as expected	Pass	High
TC-013	`run_inference` (emotion)	Predictions for emotions returned	Emotion predictions accurate	Pass	High
TC-014	`run_sarcasm_analysis`	Sarcasm training complete, flags generated	Sarcasm flags correctly output	Pass	High
TC-015	`run_inference` (sarcasm)	Sarcasm flags returned	Output correct	Pass	High
TC-016	`run_aspect_analysis`	Aspect training complete, CSV with analysis column	Aspect JSON strings generated	Pass	High
TC-017	`run_inference` (aspect)	Inference returns aspect sentiment	Aspect predictions accurate	Pass	High
TC-018	`run_full_deep_analysis`	Pipeline trains all, output merged	Full pipeline completed	Pass	High
TC-019	`predict_single_review`	JSON with all 4 outputs (sentiment, emotion, etc.)	Complete JSON returned	Pass	High

6. Version Control

Our team utilized both **GitHub** and **Discord** throughout the development process for version control and coordination.

GitHub

- Primary source control system for all frontend and backend code
- Used branches and pull requests for collaborative development
- GitHub Actions used for CI/CD deployment tests
- Repositories maintained under: https://github.com/KSU-Sentiment-Analysis

Discord

- Served as the primary communication tool for daily collaboration
- Used for stand-ups, planning, and immediate debugging discussions
- Facilitated screen sharing and pair programming when needed

These tools ensured effective version tracking, issue management, and team communication throughout the project lifecycle.

7. Conclusion

The HeartSpeak AI project successfully delivered a comprehensive sentiment analysis and AI-powered response generation platform tailored for enhancing customer communication. By combining advanced natural language processing techniques with a user-friendly interface, the application provides deep emotional insights and context-aware responses that streamline customer service workflows.

Throughout the development process, the team demonstrated strong collaboration, adaptability, and technical proficiency—especially in integrating a React frontend with a Flask backend, processing real-time data, and visualizing complex emotional metrics. While certain features like login authentication were not implemented due to infrastructure limitations, all core functionalities were fully developed, tested, and deployed.

This project not only meets its intended objectives but also lays the groundwork for future enhancements such as user authentication, data export features, and real-time sentiment alerts. With a clean and scalable architecture, HeartSpeak AI is well-positioned for continued development and real-world application in enterprise customer engagement systems.

8. Appendix

- **Project Plan**: Initial project planning documents including deadlines and feature goals.
- **Design Mock-ups**: Homepage, Sentiment Analysis, and Response Generation layout sketches.
- **Key Screenshots**:
    - Sentiment chart output
    - Emotion pie chart
    - Aspect-based bar chart
    - Response generation output table
- **Team Roster & Roles**:
    - Aravind Iyer – Lead Backend AI Developer/Tester
    - Nabeel Faridi – Front-End Developer/Tester
    - Ethan Barnes – OpenAI & Cloud Developer
    - Isha Minocha – UI/UX Designer
    - Shammah Charles – UI/UX Designer
- **Technologies Used**: React, Flask, Azure, OpenAI API, Python, GitHub, Discord
- **Glossary**:
    - *Sentiment Analysis*: Detecting emotional tone in text.
    - *Aspect-based Sentiment*: Identifying opinion on specific attributes.
    - *Sarcasm Detection*: Determining sarcastic language presence.