SentimentSense: Automated Sentiment Analysis Assistant using Gemini Model

Project Description

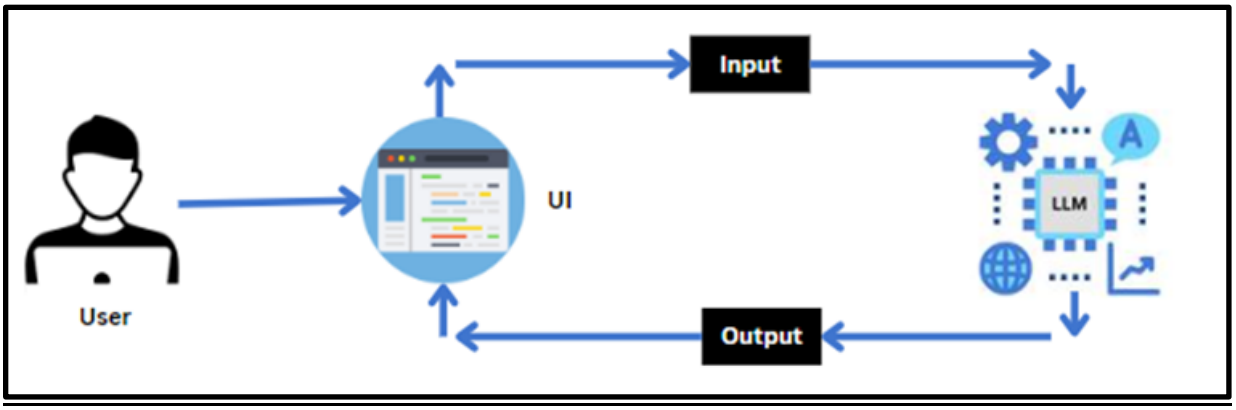
SentimentSense is an advanced project powered by a Large Language Model (LLM) designed to provide detailed and accurate sentiment analysis of text. This innovative system simplifies the process of understanding emotions and opinions in text by providing comprehensive insights, including overall sentiment, sentiment score, key phrases, emotion categories, and more. SentimentSense can be utilized across various scenarios, offering robust solutions tailored to different user needs.

Scenario 1: Customer Feedback Analysis
SentimentSense allows businesses to analyze customer feedback effortlessly. For instance, a company can input customer reviews, and SentimentSense will provide an overall sentiment (Positive, Negative, Neutral), a sentiment score, and highlight key phrases that contributed to the sentiment. This feature enables businesses to understand customer satisfaction levels, identify areas for improvement, and enhance their products or services.

Scenario 2: Social Media Monitoring
With SentimentSense, social media managers can quickly gauge public sentiment about their brand or campaigns. By analyzing social media posts, comments, and tweets, SentimentSense offers an overall sentiment, emotion categories such as happiness or anger, and a confidence score. This helps managers identify trends, respond to customer concerns promptly, and improve their social media strategies based on real-time sentiment analysis.

Scenario 3: Educational Tools
SentimentSense can be integrated into educational platforms to assist students in understanding the emotional tone of various texts. Students can input essays or articles, and SentimentSense will provide insights into the overall sentiment, polarity, and subjectivity of the text. Additionally, it can offer suggestions for improvement if the sentiment is negative, and highlight specific parts of the text that are particularly positive or negative. This helps students learn to write more balanced and positive content, improving their writing skills and emotional intelligence.

Technical Architecture



Project Flow:

- User interacts with the UI to enter the input.
- User input is collected from the UI and transmitted to the backend using the Google API key.
- The input is then forwarded to the Gemini Pro pre-trained model via an API call.
- The Gemini Pro pre-trained model processes the input and generates the output.
- The results are returned to the frontend for formatting and display.

To accomplish this, we have to complete all the activities listed below:

- **Requirements Specification**
  - Create a requirements.txt file to list the required libraries.
  - Install the required libraries
- **Initialization of Google API Key**
  - Generate Google API Key
  - Initialize Google API Key
- **Interfacing with Pre-trained Model**
  - Load the Gemini Pro pre-trained model
  - Implement a function to get gemini response
- **Model Deployment**
  - Integrate with Web Framework
  - Host the Application

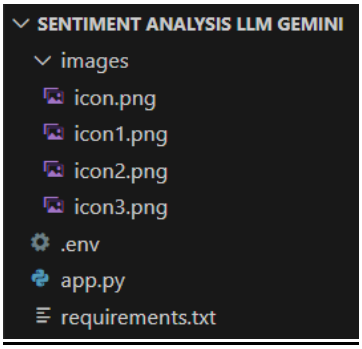
Prior Knowledge:

You must have the prior knowledge of the following topics to complete this project.

- Generative AI Concepts
- NLP: https://www.tutorialspoint.com/natural_language_processing/index.htm
- Generative AI: https://en.wikipedia.org/wiki/Generative_artificial_intelligence
- About Gemini: https://deepmind.google/technologies/gemini/#introduction
- Gemini API: https://ai.google.dev/gemini-api/docs/get-started/python
- Gemini Demo: https://colab.research.google.com/github/google/generative-ai-docs/blob/main/site/en/gemini-api/docs/get-started/python.ipynb
- Streamlit: https://www.geeksforgeeks.org/a-beginners-guide-to-streamlit/

Project Structure

Create the Project folder which contains files as shown below:



- images folder: It is established to store the images utilized in the user interface.
- .env file: It securely stores the Google API key.
- app.py: It serves as the primary application file housing both the model and Streamlit UI code.
- requirements.txt: It enumerates the libraries necessary for installation to ensure proper functioning.
- Additionally, ensure proper file organization and adhere to best practices for version control.

Milestone 1: Requirements Specification

Specifying the required libraries in the requirements.txt file ensures seamless setup and reproducibility of the project environment, making it easier for others to replicate the development environment.

Activity 1: Create a requirements.txt file to list the required libraries.

```
# libraries need to be installed
streamlit
streamlit_extras
google-generativeai
python-dotenv
Pillow
```

- streamlit: Streamlit is a powerful framework for building interactive web applications with Python.
- streamlit_extras: Additional utilities and enhancements for Streamlit applications.
- google-generativeai: Python client library for accessing the GenerativeAI API, facilitating interactions with pre-trained language models like Gemini Pro.
- python-dotenv: Python-dotenv allows you to manage environment variables stored in a .env file for your Python projects.
- Pillow: Pillow is a Python Imaging Library (PIL) fork that adds support for opening, manipulating, and saving many different image file formats.

Activity 2: Install the required libraries

- Open the terminal.
- Run the command: pip install -r requirements.txt
- This command installs all the libraries listed in the requirements.txt file.

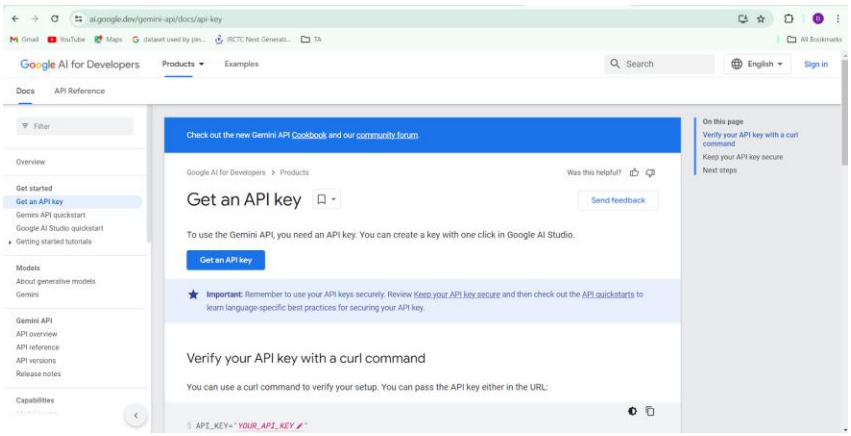
Milestone 2: Initialization of Google API Key

The Google API key is a secure access token provided by Google, enabling developers to authenticate and interact with various Google APIs. It acts as a form of identification, allowing users to access specific Google services and resources. This key plays a crucial role in authorizing and securing API requests, ensuring that only authorized users can access and utilize Google's services.

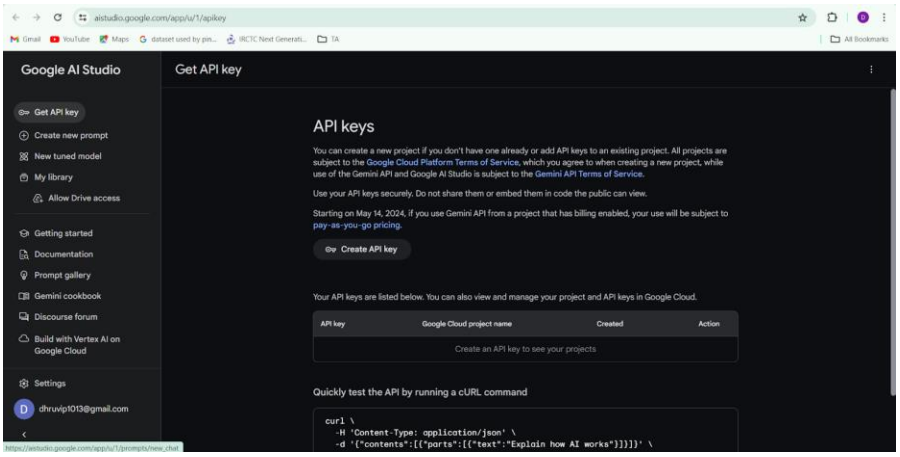
Activity 1: Generate Google API Key

Click the provided link to access the following webpage.

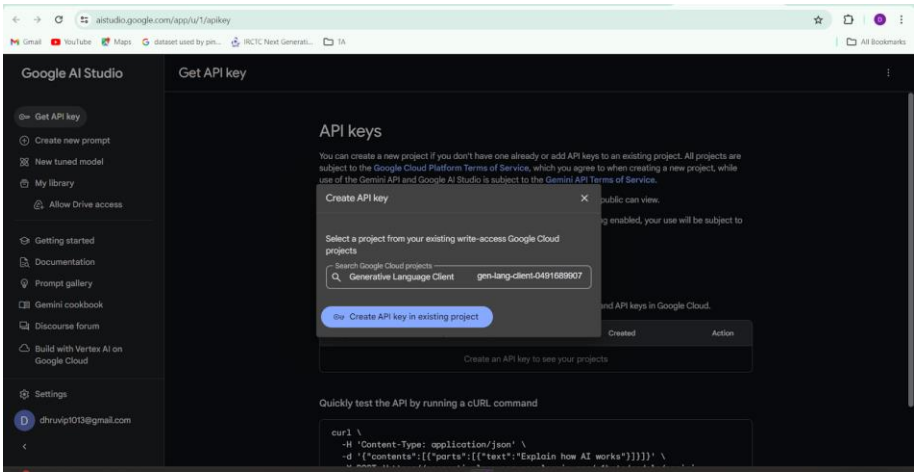
Link: https://ai.google.dev/gemini-api/docs/api-key



After signing in to your account, navigate to the 'Get an API Key' option. Clicking on this option will redirect you to another webpage as shown below.



Next, click on 'Create API Key' and choose the generative language client as the project. Then, select 'Create API key in existing project'.



Copy the newly generated API key as it is required for loading the Gemini Pro pre-trained model.

Activity 2: Initialize Google API Key

```
GOOGLE_API_KEY = "<Enter the copied Google API Key>"
```

- Create a .env file and define a variable named GOOGLE_API_KEY.
- Assign the copied Google API key to this variable.
- Paste the API key obtained from the previous steps here.

Milestone 3: Interfacing with Pre-trained Model

To interface with the pre-trained model, we'll start by creating an app.py file, which will contain both the model and Streamlit UI code.

Activity 1: Load the Gemini Pro pre-trained model

```
from dotenv import load_dotenv
import streamlit as st
from streamlit_extras import add_vertical_space as avs
import google.generativeai as genai
import os
from PIL import Image

load_dotenv()

genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))

model = genai.GenerativeModel('gemini-pro')
```

- The code begins by importing necessary libraries and modules, including dotenv, Streamlit, os, GenerativeAI from Google, PIL (Python Imaging Library), and a custom module for adding vertical space in Streamlit.
- It loads environment variables from the .env file using the load_dotenv() function.
- The GenerativeAI module is configured with the Google API key stored in the environment variable GOOGLE_API_KEY.
- A GenerativeModel object named "model" is created using the Gemini Pro pre-trained model from Google.

- The code is essentially setting up the environment, configuring the GenerativeAI module with the API key, and loading the Gemini Pro model for generating responses to user inputs in the Streamlit app.

Activity 2: Implement a function to get gemini response

```
def get_sentiment_analysis(text):
    prompt = f"Analyze the sentiment of the following text and provide a summary with the following
               parametersin a tabular format strictly:\n" \
             f"- Overall Sentiment\n" \
             f"- Sentiment Score\n" \
             f"- Key Phrases\n" \
             f"- Emotion Categories\n" \
             f"- Confidence Score\n" \
             f"- Polarity\n" \
             f"- Subjectivity\n" \
             f"- Suggestions for Improvement\n" \
             f"- Highlights of Positive/Negative Aspects\n\n" \
             f"Text: {text}"

    response = genai.GenerativeModel('gemini-pro').generate_content([prompt])
    return response.text
```

- The function get_gemini_response takes an input text as a parameter.
- A string prompt is created that instructs the AI model to analyze the sentiment of the provided text.
- The function calls the generate_content method of the model object to generate a response.
- The generated response is returned as text.

Milestone 4: Model Deployment

We deploy our model using the Streamlit framework, a powerful tool for building and sharing data applications quickly and easily. With Streamlit, we can create interactive web applications that allow users to interact with our models in real-time, providing an intuitive and seamless experience.

Activity 1: Integrate with Web Framework

The webpage is organized into four main sections to provide users with a comprehensive experience:

- Introduction:

```python
st.set_page_config(page_title="Sentiment Analysis", layout="wide")

avs.add_vertical_space(5)

col1, col2 = st.columns([3, 2])

with col1:
    st.title("SentimentSense")
    st.header("Uncovering Emotions with Precision")
    st.markdown("""<p style='text-align: justify;'>
                    Introducing SentimentSense - your advanced sentiment analysis assistant powered by
                    cutting-edge AI technology. Designed to provide detailed and accurate insights into
                    the emotions and opinions expressed in text, SentimentSense simplifies the process of
                    understanding textual sentiment. Whether you're a business looking to analyze customer
                    feedback, a social media manager monitoring brand sentiment, or an educator seeking to
                    enhance students' understanding of textual emotion, SentimentSense offers tailored
                    solutions to meet your needs. With its ability to analyze sentiment, identify key phrases,
                    and offer actionable suggestions for improvement, SentimentSense revolutionizes the way
                    you interpret and leverage textual data. Experience seamless sentiment analysis with
                    SentimentSense and unlock new insights into the emotions conveyed in text.
                    </p>""", unsafe_allow_html=True)


with col2:
    avs.add_vertical_space(5)
    img = Image.open("images/icon.png")
    st.image(img, use_column_width=True)


avs.add_vertical_space(10)
```

- This code sets up a web page for the SentimentSense application using Streamlit.
- It configures the page with a title "Sentiment Analysis" and a wide layout.
- It then creates two columns, placing a title, header, and descriptive text about SentimentSense in the first column.
- In the second column, it adds vertical space and displays an image.
- The text uses Markdown with HTML for justified alignment to enhance readability.

- Offering:

```python
col1, col2 = st.columns([3, 2])

with col1:
    img1 = Image.open("images/icon1.png")
    st.image(img1, use_column_width=True)

with col2:
    avs.add_vertical_space(4)
    st.header("Wide Range of Offerings")
    st.write("- Detailed Sentiment Analysis")
    st.write("- Effortless Customer Feedback Analysis")
    st.write("- Real-time Social Media Monitoring")
    st.write("- Educational Assistance")
    st.write("- Tailored Solutions for Different User Needs")

avs.add_vertical_space(10)
```

- This code creates a layout for a section of the SentimentSense application page using Streamlit.
- It divides the page into two columns, displaying an image in the first column using the full column width.
- In the second column, it presents a header titled "Wide Range of Offerings" followed by a list of five key features.

- Sentiment Analysis Application:

```
col1, col2 = st.columns([3, 2])

with col1:
    st.title("Let's Engage")
    st.header("Share your content for a sentiment analysis synopsis:")
    input_text = st.text_area("Input Text Here", height=100)

    if st.button("Analyze Sentiment"):
        if input_text:
            with st.spinner("Analyzing sentiment..."):
                sentiment_summary = get_sentiment_analysis(input_text)
                st.subheader("Sentiment Analysis Result")
                st.write(sentiment_summary)
        else:
            st.error("Please enter text to analyze the sentiment.")

with col2:
    img2 = Image.open("images/icon2.png")
    st.image(img2, use_column_width=True)

avs.add_vertical_space(10)
```

  - This code creates a user interface for the "Let's Engage" section of the SentimentSense application using Streamlit.
  - The page is divided into two columns: the first column includes a title, a header, and a text area for users to input text for sentiment analysis.
  - A button labeled "Analyze Sentiment" triggers the sentiment analysis process using the get_sentiment_analysis function.
  - If text is provided, a spinner indicates the analysis in progress, and the results are displayed under the "Sentiment Analysis Result" subheader.
  - The second column displays an image using the full column width.

- FAQ:

```
col1, col2 = st.columns([2, 3])

with col1:
    avs.add_vertical_space(5)
    img3 = Image.open("images/icon3.png")
    st.image(img3, use_column_width=True)

with col2:
    st.write("Q: What is SentimentSense?")
    st.write("""A: SentimentSense is an advanced project powered by a Large Language Model (LLM)
             designed to provide detailed and accurate sentiment analysis of text. It simplifies
             the process of understanding emotions and opinions in text by offering comprehensive
             insights.""")
    avs.add_vertical_space(3)
    st.write("Q: What insights does SentimentSense provide?")
    st.write("""A: SentimentSense offers insights such as overall sentiment (Positive, Negative,
             Neutral), sentiment score, key phrases, emotion categories, confidence score, polarity,
             subjectivity, and suggestions for improvement.""")
    avs.add_vertical_space(3)
    st.write("Q: How can SentimentSense be used in Customer Feedback Analysis?")
    st.write("""A: Businesses can input customer reviews into SentimentSense to obtain an overall
             sentiment, sentiment score, and highlighted key phrases. This helps in understanding
             customer satisfaction levels and identifying areas for improvement.""")
```

- This code creates a FAQ section for the SentimentSense application using Streamlit, structured into two columns.
- The first column displays an image with added vertical space for layout balance.
- The second column presents a series of questions and answers about SentimentSense.
- Each Q&A section is separated by additional vertical space for better readability.

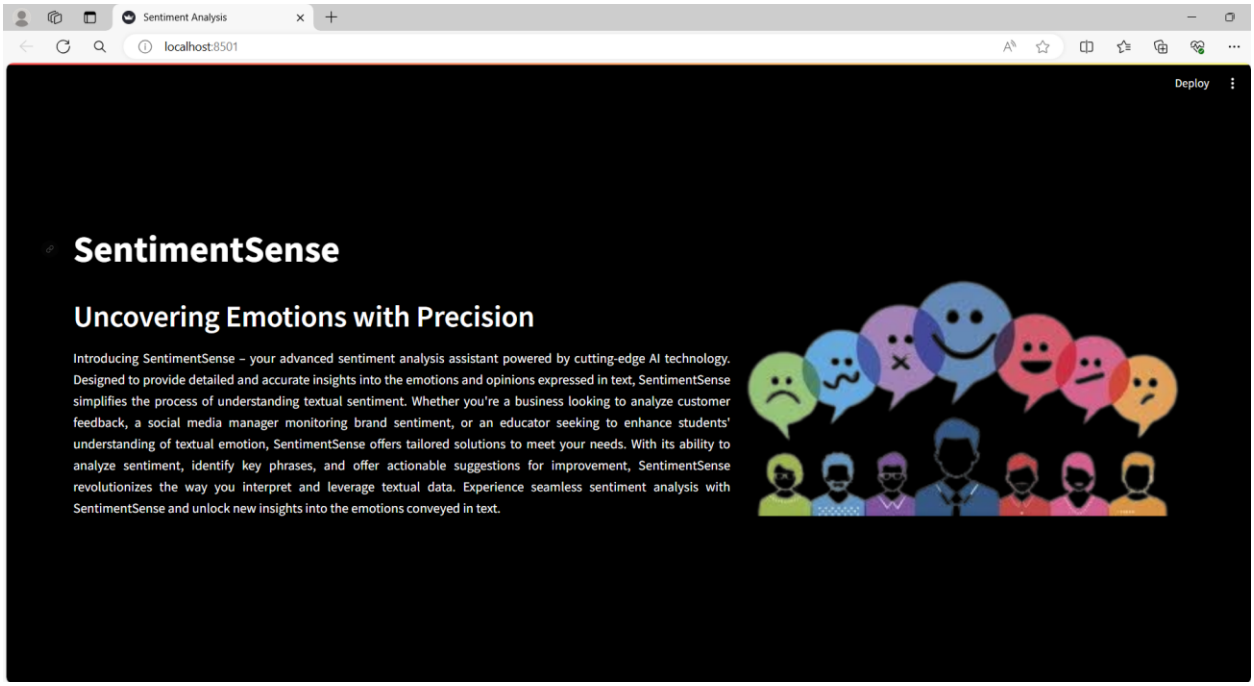
Activity 2: Host the Application

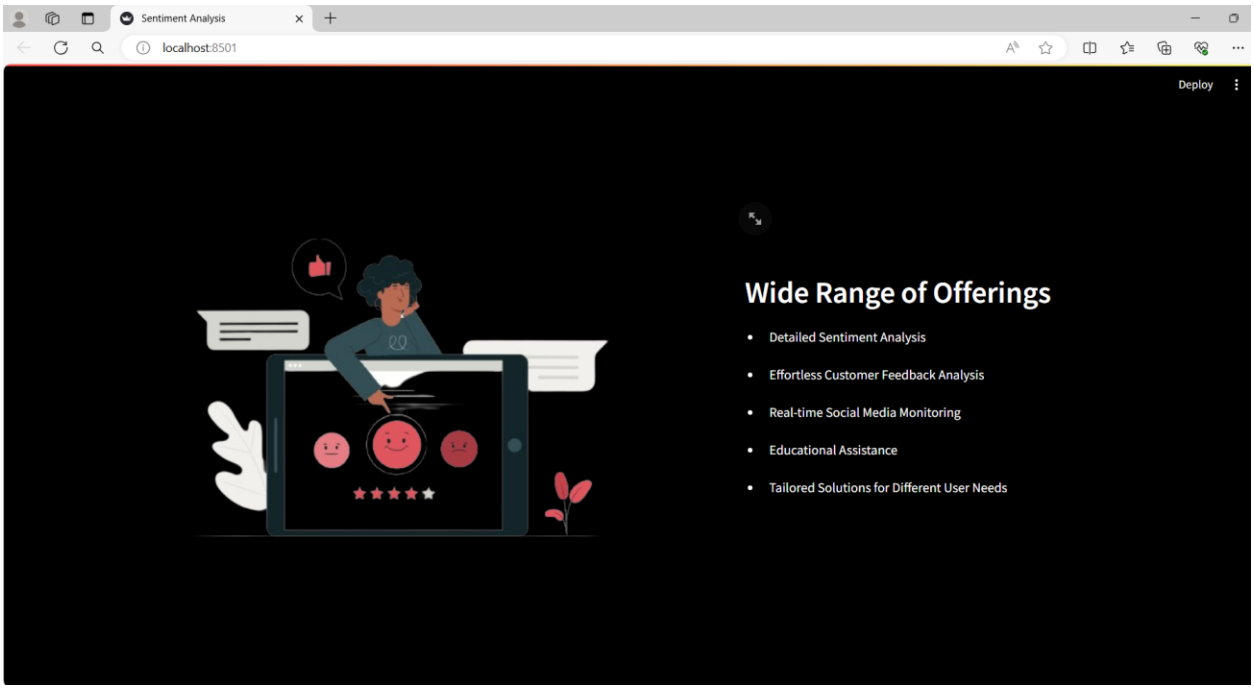
Launching the Application:

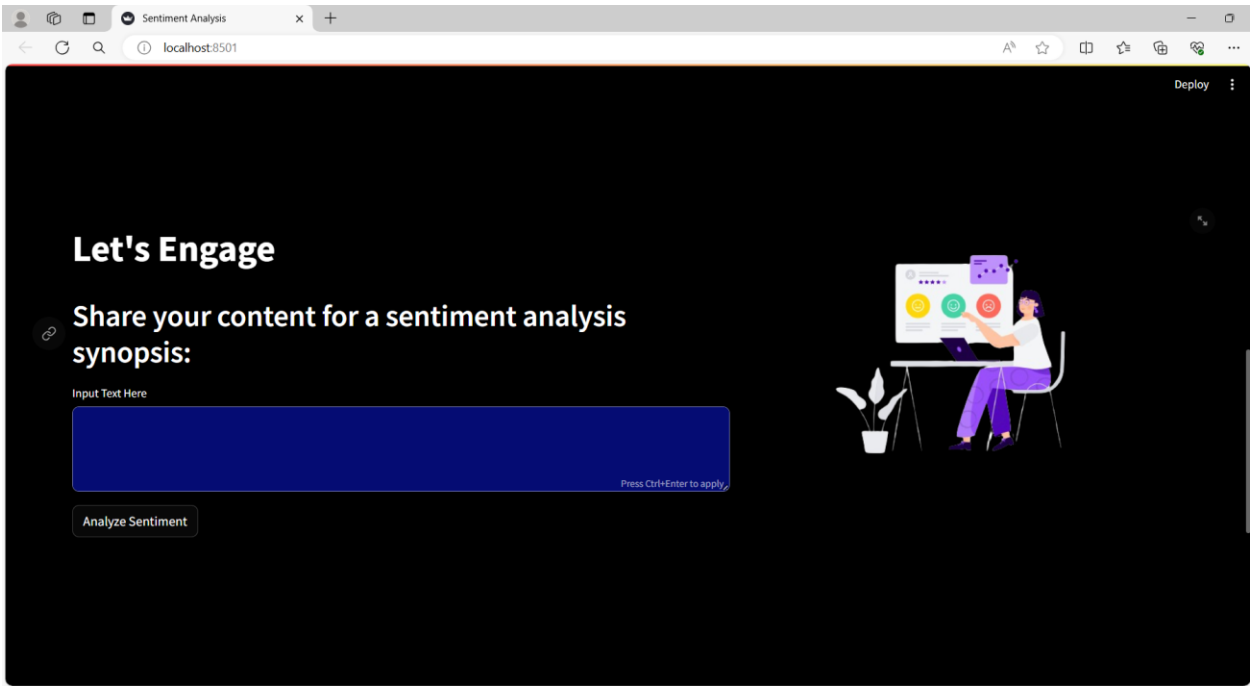
- To host the application, go to the terminal, type - streamlit run app.py
- Here app.py refers to a python script.

```
PS C:\Users\dhurvi patel\Desktop\SCIT academic\Smart_Bridge_Internship\SB_Company work\May_Month\Sentiment Analysis LLM Gemini> streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.29.80:8501
```

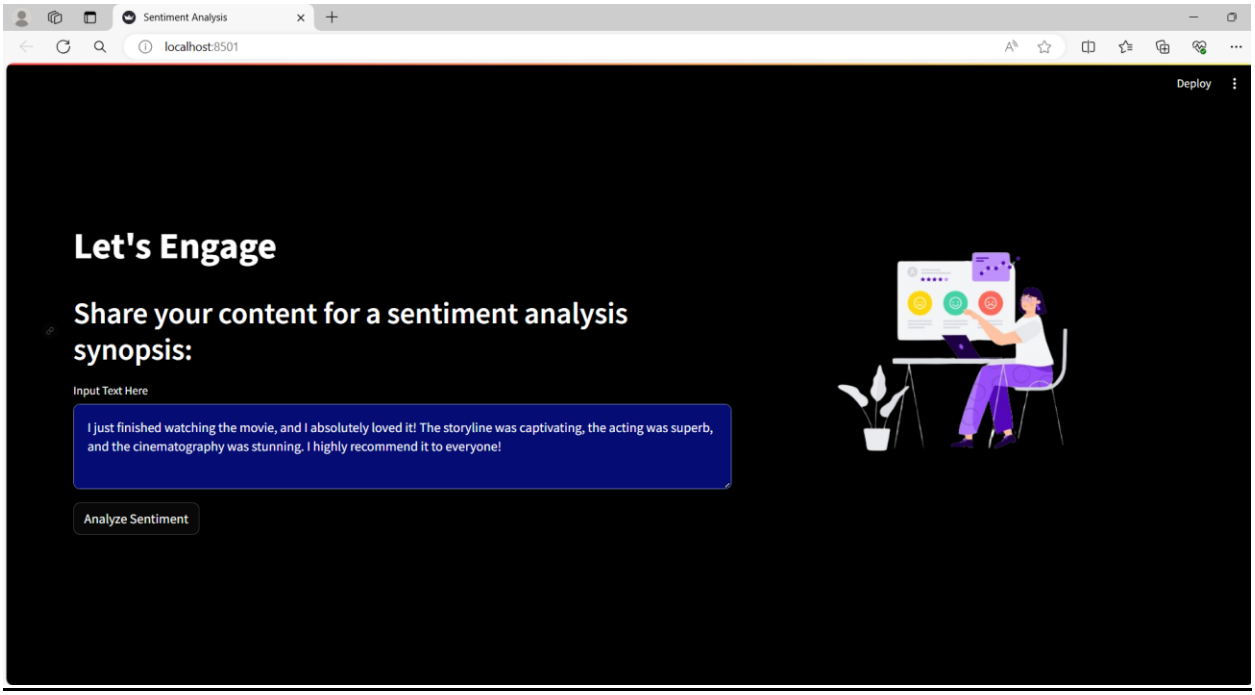
Run the command to get the below results



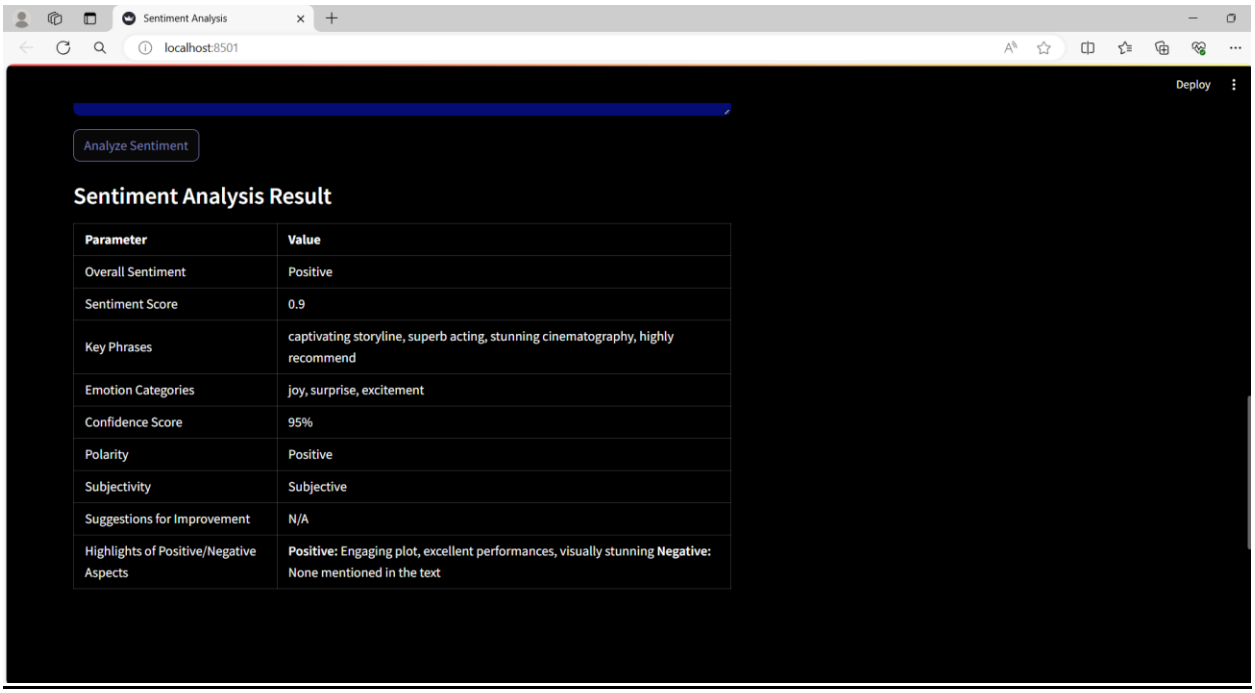

Sentiment Analysis
localhost:8501
Deploy
Wide Range of Offerings
Detailed Sentiment Analysis
Effortless Customer Feedback Analysis
Real-time Social Media Monitoring
Educational Assistance
Tailored Solutions for Different User Needs


Sentiment Analysis
localhost:8501
Deploy
Let's Engage
Share your content for a sentiment analysis synopsis:
Input Text Here
Press Ctrl+Enter to apply
Analyze Sentiment

INPUT 1

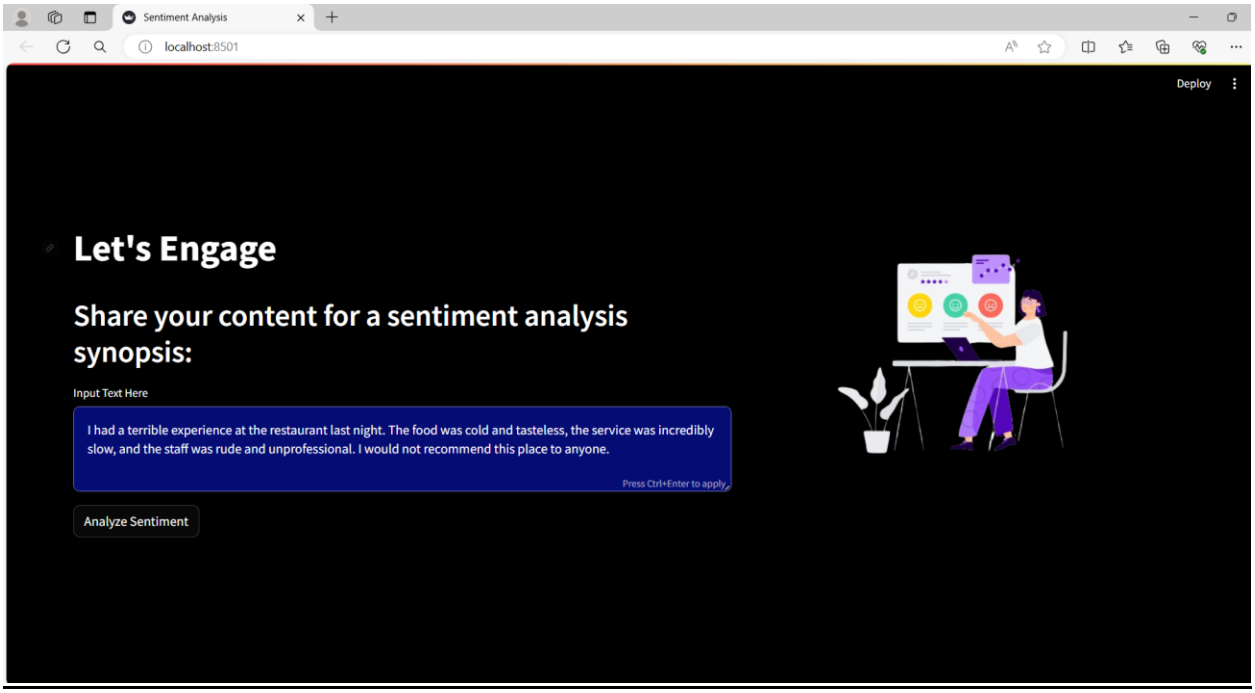


OUTPUT 1

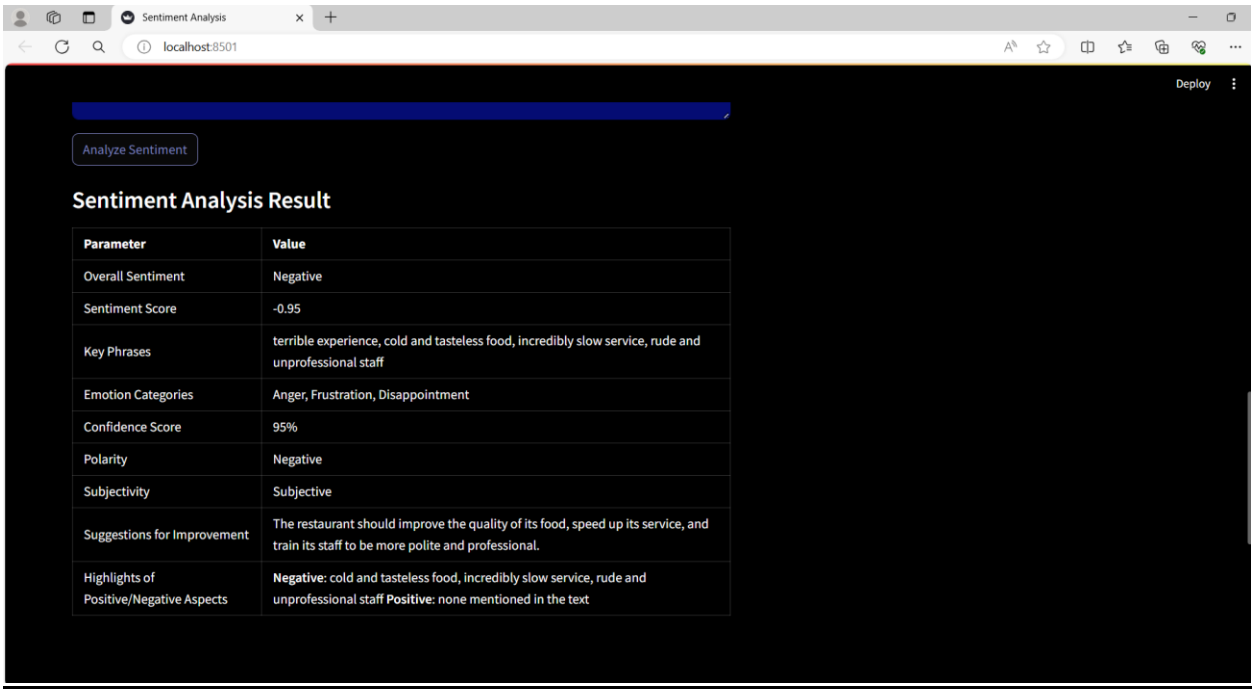


Parameter	Value
Overall Sentiment	Positive
Sentiment Score	0.9
Key Phrases	captivating storyline, superb acting, stunning cinematography, highly recommend
Emotion Categories	joy, surprise, excitement
Confidence Score	95%
Polarity	Positive
Subjectivity	Subjective
Suggestions for Improvement	N/A
Highlights of Positive/Negative Aspects	**Positive:** Engaging plot, excellent performances, visually stunning **Negative:** None mentioned in the text

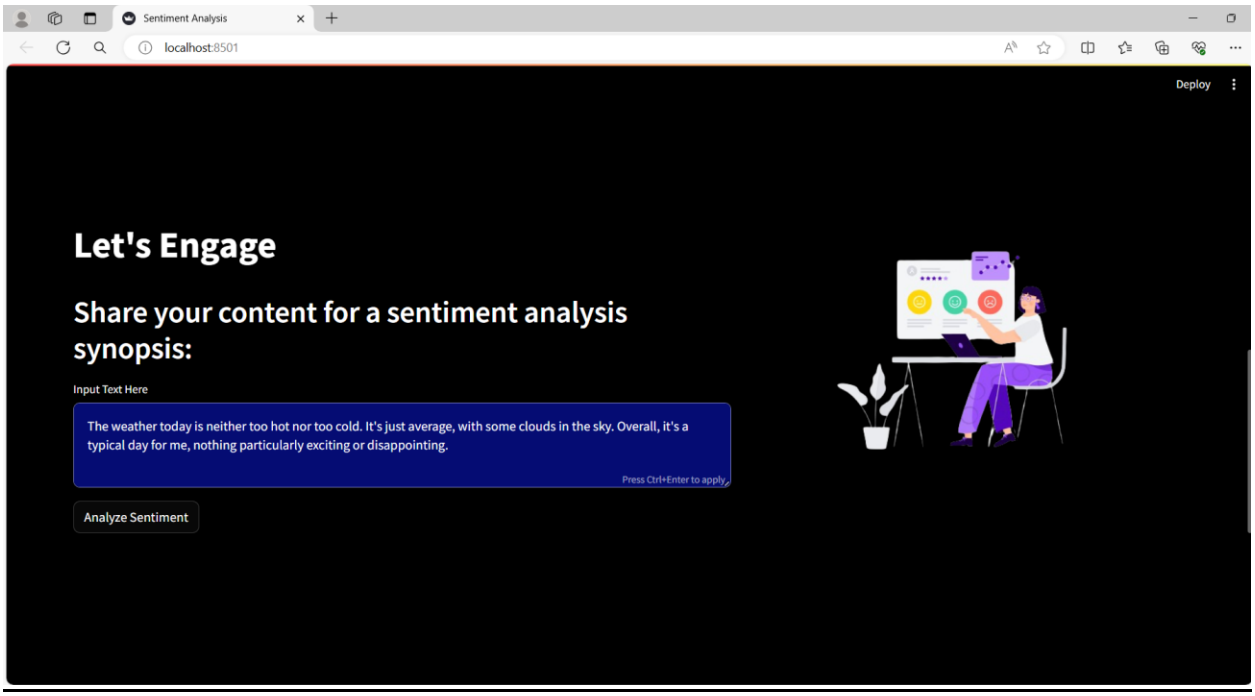
INPUT 2



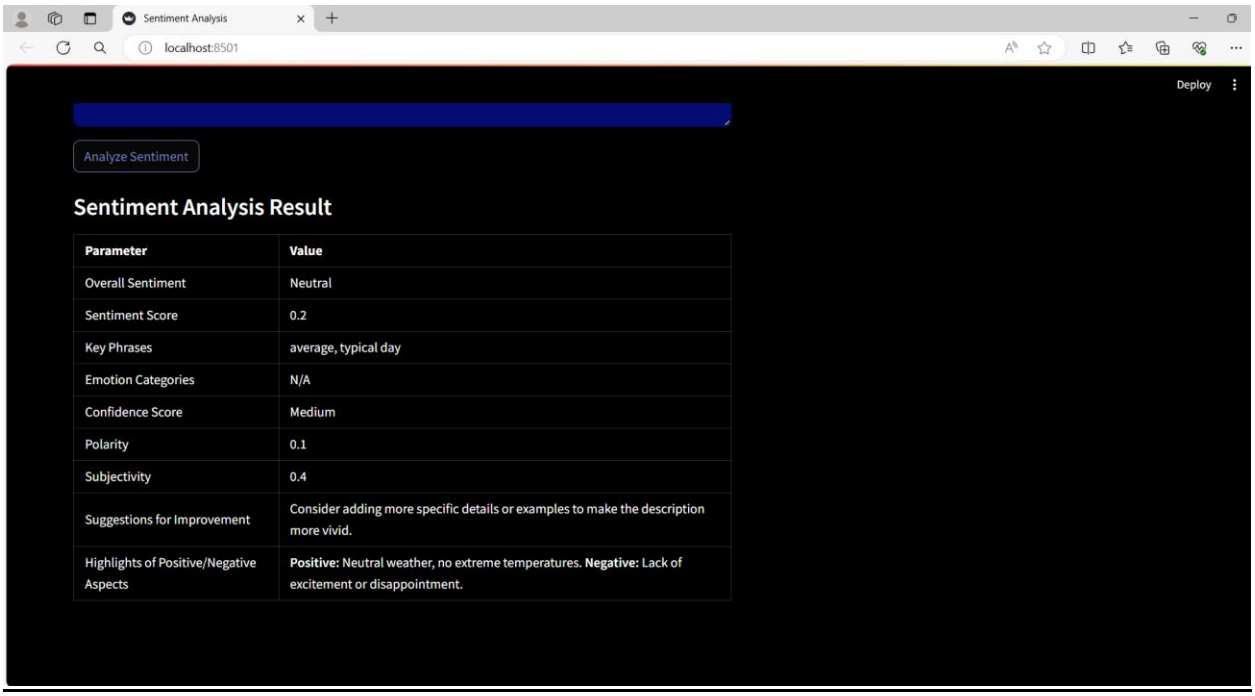
OUTPUT 2

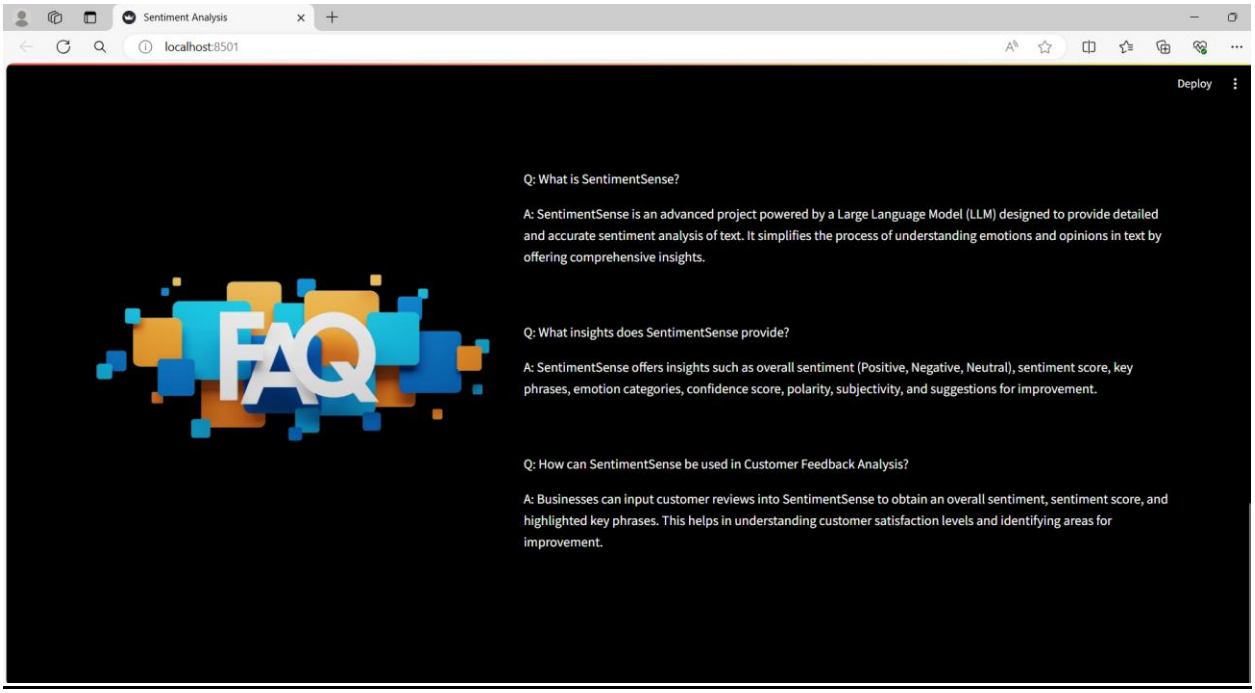


INPUT 3


Sentiment Analysis
localhost:8501
Deploy
Let's Engage
Share your content for a sentiment analysis synopsis:
Input Text Here
The weather today is neither too hot nor too cold. It's just average, with some clouds in the sky. Overall, it's a typical day for me, nothing particularly exciting or disappointing.
Press Ctrl+Enter to apply
Analyze Sentiment

OUTPUT 3


Sentiment Analysis
localhost:8501
Deploy
Analyze Sentiment
Sentiment Analysis Result
Parameter
Value
Overall Sentiment
Neutral
Sentiment Score
0.2
Key Phrases
average, typical day
Emotion Categories
N/A
Confidence Score
Medium
Polarity
0.1
Subjectivity
0.4
Suggestions for Improvement
Consider adding more specific details or examples to make the description more vivid.
Highlights of Positive/Negative Aspects
Positive: Neutral weather, no extreme temperatures. Negative: Lack of excitement or disappointment.


Sentiment Analysis
localhost:8501
Deploy
FAQ
Q: What is SentimentSense?
A: SentimentSense is an advanced project powered by a Large Language Model (LLM) designed to provide detailed and accurate sentiment analysis of text. It simplifies the process of understanding emotions and opinions in text by offering comprehensive insights.
Q: What insights does SentimentSense provide?
A: SentimentSense offers insights such as overall sentiment (Positive, Negative, Neutral), sentiment score, key phrases, emotion categories, confidence score, polarity, subjectivity, and suggestions for improvement.
Q: How can SentimentSense be used in Customer Feedback Analysis?
A: Businesses can input customer reviews into SentimentSense to obtain an overall sentiment, sentiment score, and highlighted key phrases. This helps in understanding customer satisfaction levels and identifying areas for improvement.