An
Industry Oriented Mini Project Report On

SMART SENTIMENTAL ANALYSIS ON YOUTUBE COMMENTS

Submitted in partial fulfilment of the requirements for the award of degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING
(DATA SCIENCE)

Submitted By

P. HARSHA VARDHINI	**217Z1A6744**
T. VENKTA NAGA TEJA	**217Z1A6757**
G. KARTHIK REDDY	**217Z1A6722**

Under the Guidance of
Mr. K. Anil Kumar
Assistant Professor

SCHOOL OF ENGINEERING
Department of Computer Science and Engineering

NALLA NARASIMHA REDDY
EDUCATION SOCIETY'S GROUP OF INSTITUTION
(AN AUTONOMOUS INSTITUTION)

Approved by AICTE, New Delhi, Chowdariguda (V) Korremula 'x' Roads,
Via Narapally, Ghatkesar (Mandal)Medchal (Dist), Telangana-500088

2024-2025

NALLA NARASIMHA REDDY
Education Society's Group of Institutions - Integrated Campus
(UGC AUTONOMOUS INSTITUTION)

SCHOOL OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project report titled **"SMART SENTIMENTAL ANALYSIS ON YOUTUBE COMMENTS"** is being submitted by **P. Harsha Vardhini (217Z1A6744), T. Venkata Naga Teja (217Z1A6757)** and **G. Karthik Reddy (217Z1A6722)** in Partial fulfillment for the award of **Bachelor of technology in Computer Science and Engineering (Data Science)** is a record bonafide work carried out by them. The results embodied in this report have not been submitted to any other University for the award of any degree.

Internal Guide
(Mr. K. Anil Kumar)

Head of Department
(Dr. K. Rameshwaraiah)

Submitted for Viva voce Examination held on…….…….

External Examiner

DECLARATION

We P. Harsha Vardhini, T. Venkata Naga Teja and G. Karthik Reddy are students of **Bachelor of Technology in Computer Science and Engineering(Data Science), Nalla Narasimha Reddy Education Society's Group Of Institutions**, Hyderabad, Telangana, here by declare that the work presented in this project work entitled **SMART SENTIMENTAL ANALYSIS ON YOUTUBE COMMENTS** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of engineering ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning.

By

P. Harsha Vardhini	**217Z1A6744**
T. Venkata Naga Teja	**217Z1A6757**
G. Karthik Reddy	**217Z1A6722**

Date :

Signature :

ACKNOWLEDGEMENT

We express our sincere gratitude to our guide **Mr. K. Anil Kumar,** Assistant Professor in Computer Science and Engineering Department, NNRESGI, who motivated throughout the period of the project and also for his valuable and intellectual suggestions apart from his adequate guidance, constant encouragement right throughout our work.

We would like to express our profound gratitude to our mini project In-charge **Mrs. B. Sriveni,** Assistant Professor in Computer Science and Engineering Department, NNRESGI, for her support and guidance in completing our project and for giving us this opportunity to present the project work.

We profoundly express thanks to **Dr. K. Rameshwaraiah,** Head of Computer Science and Engineering Department, NNRESGI, for his cooperation and encouragement in completing the project successfully.

We wish to express our sincere thanks to **Dr. G. Janardhana Raju,** Dean School of Engineering, NNRESGI, for providing the facilities for completion of the project.

We wish to express our sincere thanks to **Dr**. **C. V. Krishna Reddy,** Director NNRESGI, for providing the facilities for completion of the project.

Finally, we would like to thank overall mini-project coordinator, **members of Project Review Committee (PRC),** all the faculty members and supporting staff of the Department of Computer Science and Engineering, NNRESGI, for extending their help in all circumstances.

By

P. Harsha Vardhini **217Z1A6744**

T. Venkata Naga Teja **217Z1A6757**

G. Karthik Reddy **217Z1A6722**

ABSTRACT

The primary objective of this project is to evaluate and summarize viewer's sentiments, providing insights into overall reactions, common themes, and viewer concerns. This helps content creators understand audience emotions and attitudes, providing feedback for content improvement, ultimately enabling them to refine their content and enhance viewer engagement. To achieve the project objectives, we applied a combination of data collection, preprocessing, and sentiment analysis techniques. Data was gathered using the YouTube API to extract comments, including both textual content and emojis, from selected videos. The system performs NLP techniques such as text cleaning, removal of stop words, and filters the comments by understanding the emotion using sentiment analysis tools like VADER sentiment. Comments categorized as positive express favorable opinions, joy, praise, or overall satisfaction. Negative comments reflect dissatisfaction, technical issues, criticism, or unfavorable opinions. Neutral comments do not convey strong positive or negative emotions. The system provides feedback to the video based on the analysis of the comments and emojis. Furthermore, the project offers an additional feature that allows content creators to download the report generated by the system, which provides valuable insights into viewer sentiments. Using libraries like Python and NLTK (Natural Language Toolkit), the machine is able to understand and analyze the emotions expressed through both text and emojis. This project has important implications for content creators and platform managers. By understanding the sentiment trends in YouTube comments, stakeholders can tailor their content and community management strategies to enhance viewer satisfaction and engagement. Future work could explore the integration of sentiment analysis with other metadata, such as video views and likes, to provide a more comprehensive understanding of user engagement. Additionally, extending the analysis to multilingual comments could further enhance the applicability of the sentiment analysis model.

Keywords:

Sentiment Analysis, YouTube Comments, Natural Language Processing, User reviews.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION

The motivation behind this project comes from the growing importance of the audience feedback for content creators, businesses, and marketers. With millions of YouTube videos being uploaded and consumed daily, creators often struggle to understand the exact emotion of their audience due to the vast number of comments. It would be better if there's tool that can automatically analyze and classifies user sentiments (positive, negative, or neutral) to save time and offer valuable feedback for content improvement.

Additionally, brands and marketers often rely on social media platforms like YouTube to promote their products. By understanding the sentiment of comments, companies can gauge their success to the top, understand customer interests, and optimize their strategies.

1.2 PROBLEM STATEMENT

Manually reviewing the YouTube comments to access the sentiment is inefficient and time consuming, especially when there are large number of comments for a video. Traditional sentiment is useful for structure text only.

1.3 PURPOSE

The main purpose of this project is to create a tool for YouTube content creators. This tool will help them understand the emotions of their subscribers, as expressed through comments. By gaining insight into their subscribers' emotions, content creators can produce positive content that resonates with their audience. Ultimately, this will benefit the audience, as they will receive content they enjoy, and the creators, who will receive encouragement, recognition, and financial support.

1.4 SCOPE

To build a we application where users can give the URLs as an input and receive the sentiment analysis of the video as results in real-time. Providing insights of a video by representing the comments as positive, negative and neutral.

Integrating visual elements like pie chart and bar graph to display the sentiment distribution. Offering additional features, such as downloading a PDF report for the analysis.

1.5 PROJECT OBJECTIVE

To develop a system which can extract the comments from YouTube using an Api and analyze the sentiment of comments by classifying them into positive, negative and neutral. This also can analyze the emojis to improve the sentiment classification,

Additionally, it provides graphical representations of the sentiment distribution across comments, allowing users to easily interpret the results and highlight the positive, neutral and negative comments to give content creators actionable insights about the audience opinions.

1.6 LIMITATIONS

Sarcasm detection: The system is not capable of accurately detecting sarcasm, which can lead to misinterpretation of certain comments.

API rate limits: The project relies on the YouTube API to fetch comments, which may restrict the number of comments that can be analyzed due to rate limits imposed by the API.

Limited language support: The system mainly supports English comments. Analyzing comments in other languages may require additional resources and models for accurate interpretation.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

In this digital age, YouTube has emerged as one of the most influential platforms and also became a life source for many people, with billions of users and millions of hours of content uploaded daily. Producing content that meets audience interest and current trends became a challenging task for the content creators. Also, it provides a vast and diverse repository of user interactions in the form of comments. These comments offer valuable insights into viewer's opinions, emotions, and engagement with the content, which helps the content creator understand what to produce and what not to. This also plays a crucial role in marketing, as it helps the creator understand their clients' expectations and the sentiment behind a video.

But manually analyzing these comments is a challenging task and time-consuming. This project aims to analyze the sentiments of comments using the NLP (Natural Language Processing) techniques.

2.2 EXISTING SYSTEM

- **Social Blade:** It is an analytics platform focused on the social media applications such as YouTube. It provides insights on subscriber growth, Video Views and Likes but doesn't offer text analysis or detailed sentiment analysis of comments.
- **YouTube Studio Analytics:** It's a YouTube native platform provides some insights into audience engagement such as Likes, Dislikes, and Comments. However, it lacks advanced sentiment analysis like categorizing comments into positive, negative and neutral.
- **Rapid API for YouTube Comments Sentiment Analysis:** Some APIs available on rapid API are designed specifically for YouTube comments sentiment analysis but requires a third-party service, often comes with a usage limitation

Disadvantages of Existing system:

1. Many tools offer sentiment analysis, only are few produce the report directly to YouTube creators that results actionable insights to improve their content**.**
2. Provides real-time feedback for content creators
3. No platforms offer the ability to generate a PDF report summarizing sentiment analysis.
4. Many existing systems struggle to analyze emojis effectively.

2.3 PROPOSED SYSTEM

To overcome the limitations of existing system we propose Smart Sentiment Analysis on YouTube Comments system This system fetches the comments from the YouTube in real-time using the API. The system accepts the input as an URL and generates the report according to that respective video. The generated report can be downloaded in the pdf format which helps the content creators to improve the content and engage the audience according to their interests.

Advantages of proposed system.

1. By YouTube's Data API, the system allows for real-time sentiment analysis of video comments**.**
2. The system filters the mechanism to remove irrelevant comments containing the hyperlinks accepts only one comments from an individual.
3. Unlike the traditional models this system also accepts the emoji for better accuracy.
4. Top comments are identified for the video.
5. Allows the users to download the generated report for the video.

CHAPTER 3

SYSTEM ANALYSIS

3.1 FUNCTIONAL REQUIREMENTS

- **YouTube Video URL Input:**

  The system allows the users to enter a valid input URL. This evaluates the provided URL to conform this is a YouTube video. It extracts the video Id from the URL passed.

- **Fetching YouTube Comments:**

  Using the YouTube Data API the comments are fetched up to 800 from the YouTube video and the system filters the comments which contain hyperlinks, avoiding the spam or irrelevant comments.

- **Sentiment Analysis and Emoji Interpretation:**

  The system will have to perform a sentiment analysis on each relevant comment so as to classify it as either positive, negative or neutral. Emojis, slang, and casual language are taken into consideration in the sentiment analysis Sentiment scores should be determined by algorithms of natural language processing such as VADER and others or computer intelligent. The system must be able to analyze the common emoji's used in comments and alter sentiment scores based on their meaning. Positive and negative emoji's used in the comments should have weight, whether it is a positive comment or a negative comment.

- **Sentiment Visualization:**

  The system should generate the output in visual representations for the sentiment distribution using pie charts and bar graphs.

  The visualization shows the proportion of positive, negative and neutral comments.

- **Downloadable PDF Report:**

  The system allows the users to download the Pdf report of the sentimental analysis which includes the sentiment percentages, visualization, and top comments.

3.2 NON-FUNCTIONAL REQUIREMENTS

- **Performance:**

  The performance of the system is calculated by the processing speed of a system. The system must fetch and analyze the comments efficiently by processing up to 800 comments under 30 seconds. Large-scale videos with high user participation should run smoothly and without any delay.

- **Security:**

  Using an API key, the system should be able to securely communicate with the YouTube Data API. User information, including the video URL and all other details needs to be kept private so that they are not disclosed to outside parties.

- **Scalability:**

  The system needs to have the capacity to manage several YouTube videos at once, particularly those that receive a lot of comments. In order to avoid exceeding the rate limitations set by the YouTube API, the system needs to effectively handle API requests.

- **Usability:**

  Without requiring any technical knowledge, users should be able to the input YouTube URLs and review the results and an interactive user interface.

  The visualizations should be simple to understand, especially for those who are not familiar with data analysis.

3.3 INTERFACE REQUIREMENTS

3.3.1 User requirements

- Basic knowledge on YouTube URLs
- Internet Access

3.3.2 System Requirements

3.3.2.1 Software Requirements

Operating system : Windows , Linux or MacOS

Programing language : Python 3.7

Front end languages : HTML and JavaScript

Cross Platform : VS code , PyCharm (community) or terminal

Browser : Google chrome, Mozilla Firefox or Microsoft Edge

Python Libraries : Vader sentiment, emoji, re, Jspdf, html2canvas, Googleapiclient and matplotlib/chart.js

API Key : YouTube Data API

3.3.2.1 Hardware Requirements

Server Side Requirements:

CPU : 2 core processor(Intel Core i3 or equivalent)

RAM : At least 4GB RAM

Storage : 10 GB to store files.

Internet Connection

Client Side Requirements

Device : Desktop, smartphone, laptop or tablet.

Processor : minimum of 2 GHZ dual core processor.

RAM : 2 GB

Display : At least 1024x768

3.3.3 Performance requirements

It works fine with moderate internet speed. The connection is secured and the API is stored in secured manner. Sentimental analysis is accurate over 80 % particularly dealing with the comments which consist informal language, emojis and spam comments. The input URL is taken from the user correctly and respond is recorded quickly.

CHAPTER 4

SYSTEM DESIGN

4.1 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general- purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts. Be independent of particular programming languages and development process. Provide a formal basis for understanding the modeling language.
- Encourage the growth of Object-Oriented tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.

4.1.1 Data Flow Diagrams

A data flow diagram maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. There are 2 levels in the below diagram.

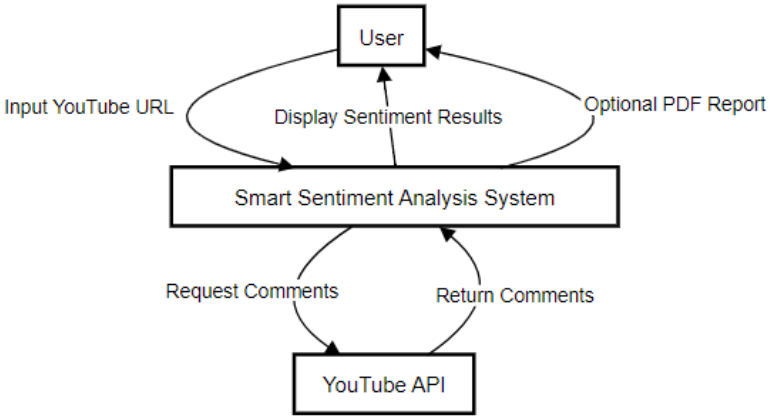

Fig : 4.1.1.1 Data Flow Diagram :level 0

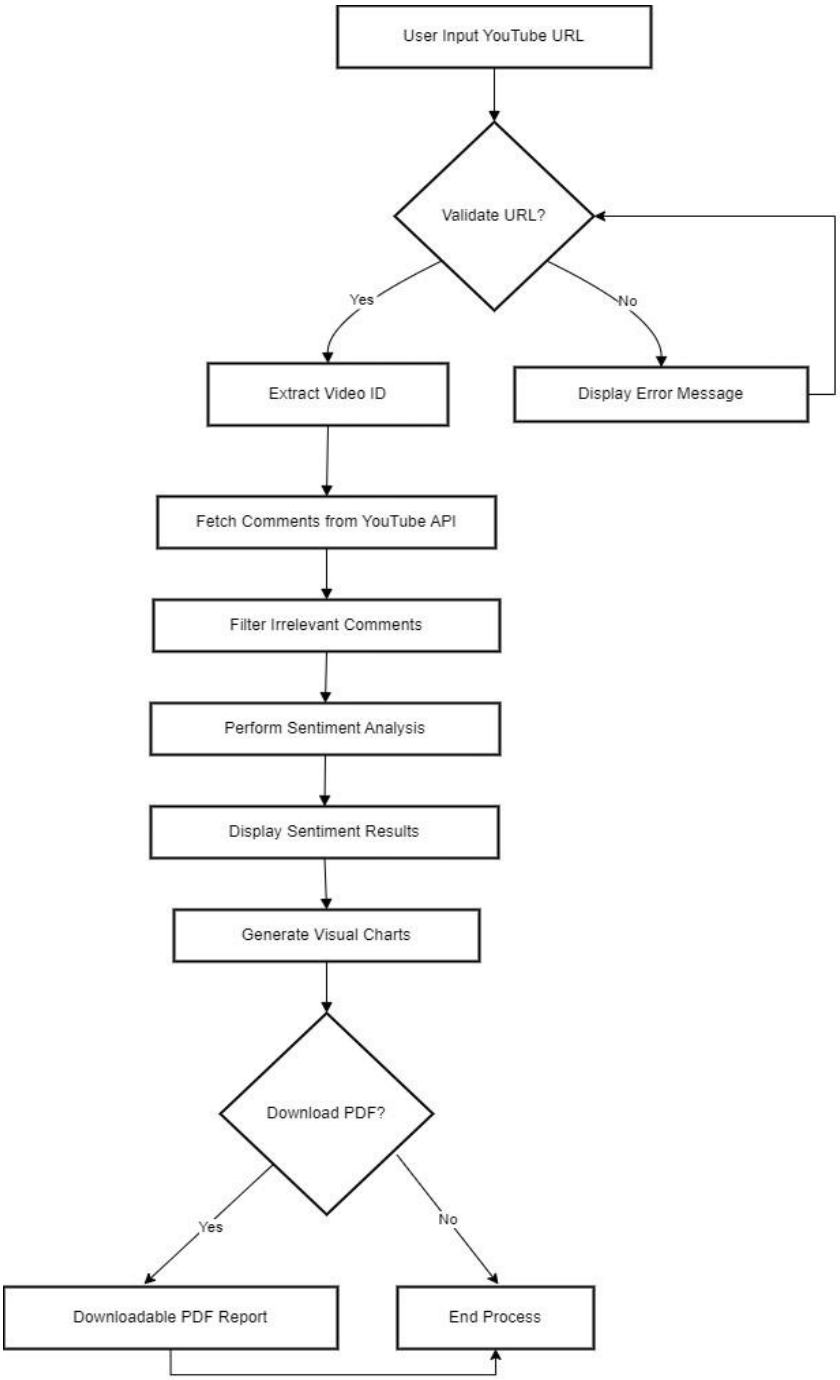

Fig : 4.1.1.2 Data Flow Diagram :Level 1

4.1.2 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is topresent a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use case, and any dependencies between those usecases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

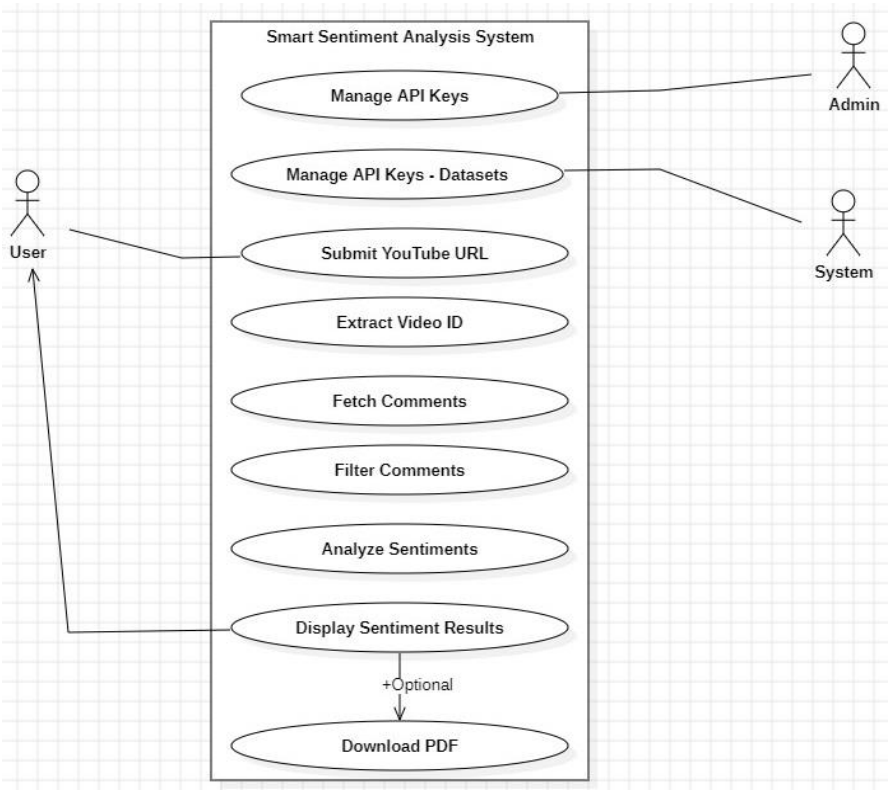

Fig : 4.1.2.1 Use case Diagram

4.1.3 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML)is a type of static structure diagram that describes the structure of a system by showingthe system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

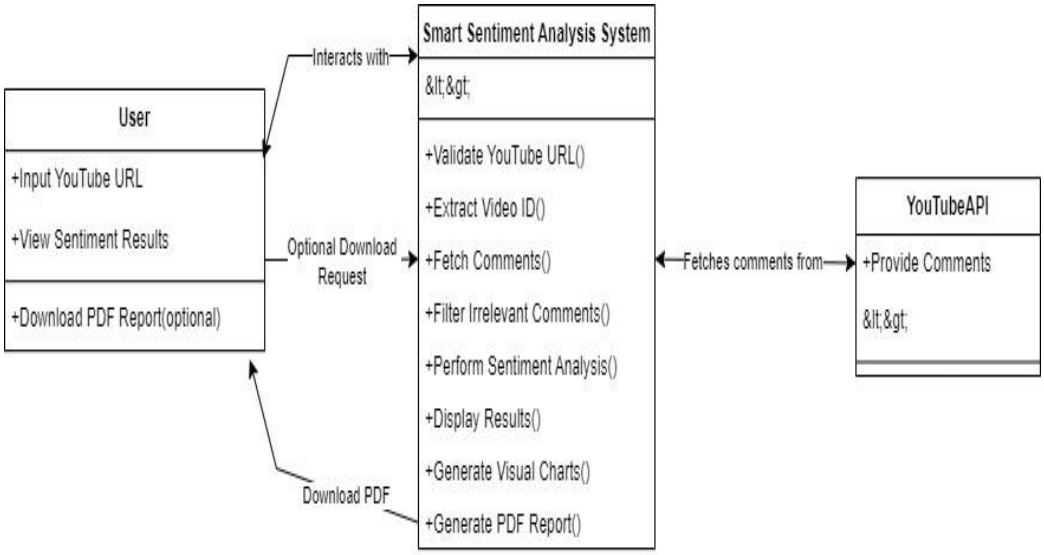

Fig : 4.1.3.1 Class Diagram

4.1.4 Sequence Diagram

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

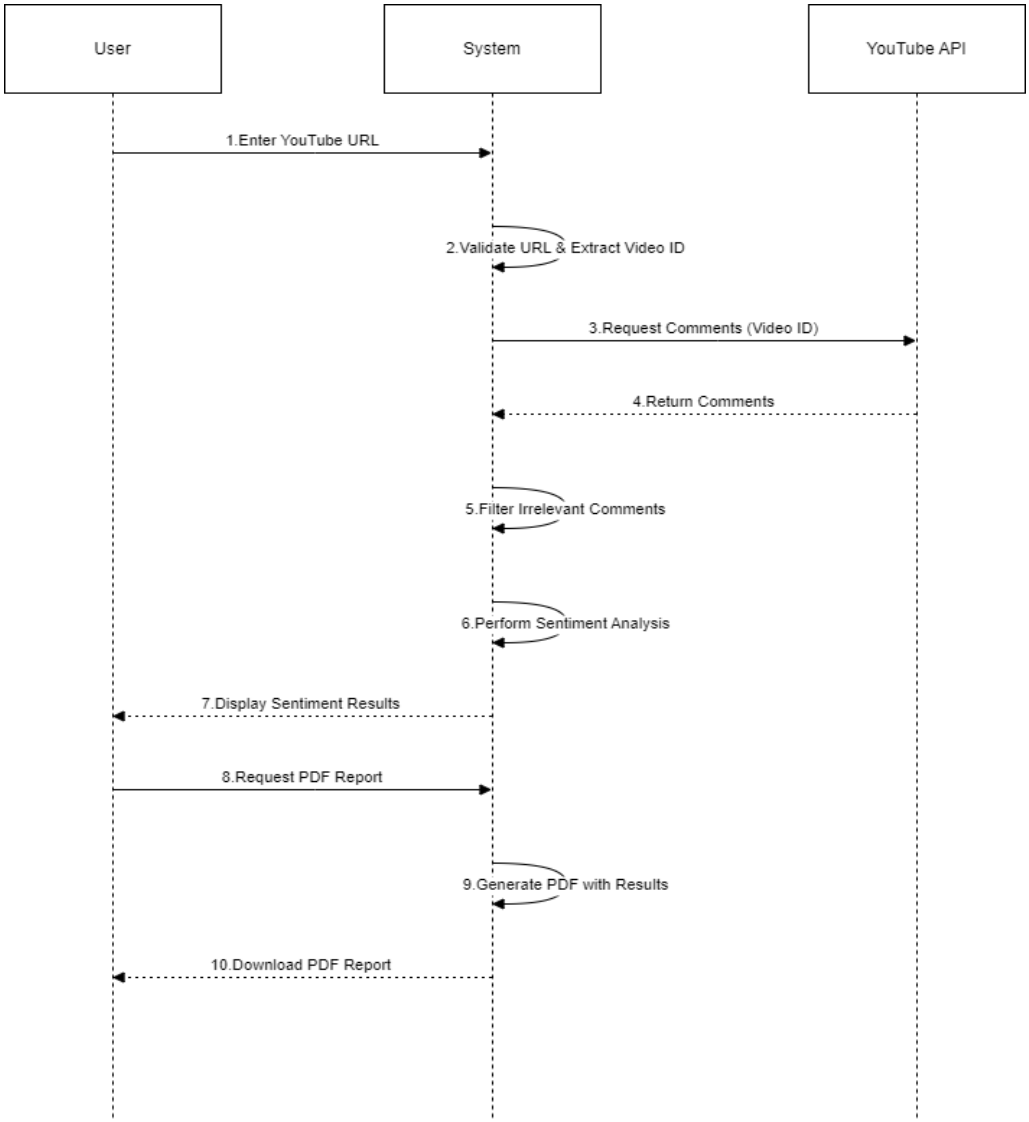

Fig : 4.1.4.1 Sequence Diagram

4.2 MODULES

YouTube Sentiment Analysis Functional Requirements:

1. **Input Video URL:** The user enters a YouTube URL in the User interface along with selecting the no. of comments
2. **Video ID Extraction:** The system extracts the video ID from the provided URL.
3. **Fetch Comments:** The system uses the YouTube API to fetch the comments of the video.
4. **Filter Comments:** The system filters out irrelevant comments (such as those with hyperlinks or excessive emojis, non-English comments, and comments which are posted by the same user multiple times. This is increase accurate results)
5. **Sentiment Analysis:** The system runs sentiment analysis on the filtered comments using VADER sentiment and categorizes them into 3 sections Positive, Neutral and Negative. And then performs Calculations.
6. **Display Sentiment Results:** The system shows whether the sentiment for the video is positive, negative, or neutral, along with some top comments and counts.
7. **Download:** This system allows the user to download the generated report in the pdf format. This is an optional step if the user wants an report then he/she will download else only the generated report is displayed.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 METHOD OF IMPLEMENTATION

5.1.1 PYTHON

Python is currently the most widely used multi-purpose, high-level programming language, Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code.

5.1.2 HTML

HTML stands for Hyper Text Markup Language. It was invented by Tim Berners Lee in 1991. Many say html is the backbone of Webpages. HTML is used in frond end work where developers focus on the design, structure and to create a meaningful web content. The HTML elements are the building block of a web page, web content contain elements like headings, tags, images, paragraphs, links. HTML provides data in form of text or multimedia which is written in scripting language such as the javaScript that effects the behavior and content of the page.

5.1.3 JAVASCRIPT

JavaScript is a high-level, versatile programming language primarily used for creating dynamic and interactive content on websites. It enables real-time manipulation of HTML and CSS, allowing developers to modify web pages based on user interactions like clicks and form submissions. JavaScript supports event-driven and asynchronous programming, making it ideal for tasks like fetching data from servers without refreshing the page. While traditionally used for client-side scripting, it can also run on

the server side using platforms like Node.js, making it a popular choice for full-stack development. Its flexibility, object-oriented capabilities, and wide browser support have made it essential for modern web development**.**

MODULES USED IN PROJECT

Vader Sentiment

Vader (Valence Aware Dictionary and Sentiment Reasoner) is a rule-based sentiment analysis tool that is specifically designed for analyzing the social media texts. Vader is an open source module used to calculate the sentiment of the text. Vader is a pre-trained sentiment analysis model that provides a sentiment score for a given text of data. Vader uses a dictionary of words and rules for determining the sentiment for a piece of text data. It uses a valence score for each word to determine whether it's a positive, negative or a neutral.

Emoji

The emoji module in Python is a library used for handling and manipulating emojis in strings. It allows developers to work with emojis in a variety of ways, such as adding, removing, or counting emojis within text. This is especially useful when dealing with user-generated content, like social media posts or comments, where emojis are frequently used to convey emotions or reactions.

Re module

The re module in Python provides support for working with regular expressions, which are powerful tools used for pattern matching and searching in strings. Regular expressions allow for sophisticated searching, matching, and manipulation of text by defining specific patterns to identify particular sequences of characters.

Googleapiclient

The Google API Client library for Python is a tool that simplifies interaction with Google APIs by providing a high-level interface to communicate with various Google services. It helps developers access and manage data from Google services like YouTube, Google Drive, Google Calendar, and others through their APIs.

Flask

Flask is a lightweight and flexible Python web framework designed for building web applications and APIs with minimal setup. It follows a minimalist approach, providing essential tools for routing, request handling, and template rendering while allowing developers the freedom to add features through extensions. Flask's simplicity makes it easy to create and maintain applications, offering a built-in development server for testing and debugging. Its straightforward design and support for RESTful API development have made it a popular choice for creating custom web solutions efficiently.

Jspdf and html2canvas

jsPDF and html2canvas are JavaScript libraries often used together to generate PDF documents from HTML content in web applications. html2canvas captures a screenshot of the HTML content, rendering it as a canvas image, while jsPDF takes this image and integrates it into a PDF file. This combination allows developers to convert web pages or specific elements into downloadable PDFs, preserving the visual layout and content formatting. Together, they enable seamless client-side PDF generation directly from HTML and CSS.

Chart.js

Chart.js is a popular open-source JavaScript library for creating interactive and customizable charts and graphs. It provides an easy-to-use API for rendering various types of visualizations, such as line, bar, pie, and radar charts, using HTML5 canvas elements. Chart.js supports features like animations, tooltips, and responsive design, making it ideal for incorporating data visualizations into web applications. Its simplicity and flexibility allow developers to create visually appealing charts with minimal configuration, enhancing the presentation and analysis of data on websites and dashboards

5.2 EXPLANATION OF KEY FUNCTIONS

1. Extract_video_id(url)

- Improve Video ID Extraction: Enhance the regex to handle different URL formats or use the urllib.parse library to parse URL parameters more robustly.
- Handle Errors: Add error handling to manage cases where the video ID cannot be extracted due to malformed URLs or other issues.

2. Fetch_comments(video_id, max_comments)

- Pagination and Rate Limiting: Implement better pagination handling and manage API rate limits more effectively.
- Filter Comments by Date: Add an option to filter comments based on the date they were posted.
- Store Comments: Store fetched comments in a database for later analysis or querying.

3. Filter_comments(comments)

- Advanced Text Analysis: Incorporate more advanced text analysis techniques to filter comments based on sentiment, keywords, or topics.
- Language Detection Enhancement: Improve language detection accuracy or support additional languages.

4. Analyze_sentiments(comments)

- Sentiment Analysis Customization: Adjust sentiment analysis based on specific use cases, such as recognizing domain-specific terms or slang.
- Visualize Sentiment Data: Integrate visualization libraries like Chart.js to present sentiment distribution in charts.

5. Index ()

- Dynamic Content: Add dynamic content to the index.html template, such as displaying recent analysis results or user statistics.
- User Authentication: Implement user authentication to restrict access to certain features or data.

6. Analyze ()

- Error Handling: Improve error handling for various scenarios, such as invalid video URLs or API errors.
- User Feedback: Provide more detailed feedback to users about the analysis process or results.
- Export Results: Add functionality to export analysis results as CSV or Excel files.

5.3 SOURCE CODE

5.3.1 BACK END CODE

```
from flask import Flask, request, render_template, jsonify
from googleapiclient.discovery import build
import re
import emoji
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from langdetect import detect, LangDetectException
import config
app = Flask(__name__)

def extract_video_id(url):
    video_id = None
    regex = r'(?:v=|\/)([0-9A-Za-z_-]{11}).*'
    match = re.search(regex, url)
    if match:
        video_id = match.group(1)
    return video_id

def fetch_comments(video_id, max_comments):
    youtube = build('youtube', 'v3', developerKey=config.key)
    comments = []
    nextPageToken = None

    while len(comments) < max_comments:
        request = youtube.commentThreads().list(
            part='snippet',
            videoId=video_id,
            maxResults=min(200, max_comments - len(comments)),  # Adjust maxResults according to
remaining comments
            pageToken=nextPageToken
        )
        response = request.execute()

        for item in response['items']:
            comment = item['snippet']['topLevelComment']['snippet']
            if 'authorChannelId' in comment:
                comments.append(comment['textDisplay'])
        nextPageToken = response.get('nextPageToken')
        if not nextPageToken:
            break
    return comments

def filter_comments(comments):
    hyperlink_pattern = re.compile(
        r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\\(\\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+')
    threshold_ratio = 0.65
    relevant_comments = []

    for comment_text in comments:
        comment_text = comment_text.lower().strip()
        emojis = emoji.emoji_count(comment_text)
        text_characters = len(re.sub(r'\s', '', comment_text))
        try:
            if detect(comment_text) != 'en':  # Filter out non-English comments
                continue
        except LangDetectException:
            continue  # If language detection fails, skip the comment
```

```
        if (any(char.isalnum() for char in comment_text)) and not
hyperlink_pattern.search(comment_text):
            if emojis == 0 or (text_characters / (text_characters + emojis)) > threshold_ratio:
                relevant_comments.append(comment_text)
    return relevant_comments

def analyze_sentiments(comments):
    sentiment_object = SentimentIntensityAnalyzer()
    emoji_positive = ['🔥', '😍', '🥳', '😃', '😊', '👍', '💯', '🤡', 'uffffff']
    emoji_negative = ['👺', '😡', '🤌', '😰', '😞', '💔', 'mad', 'really mad', 'killing', 'no spiderman']
    polarity = []
    positive_comments = []
    negative_comments = []
    neutral_comments = []

    for comment in comments:
        sentiment_dict = sentiment_object.polarity_scores(comment)
        score = sentiment_dict['compound']

        # Adjust score based on presence of positive/negative emojis
        for emj in emoji_positive:
            if emj in comment:
                score += 0.2
        for emj in emoji_negative:
            if emj in comment:
                score -= 0.2

        polarity.append(score)
        if score > 0.05:
            positive_comments.append(comment)
        elif score < -0.05:
            negative_comments.append(comment)
        else:
            neutral_comments.append(comment)

    return polarity, positive_comments, negative_comments, neutral_comments

@app.route('/', methods=['GET', 'POST'])
def index():
    return render_template('index.html')

@app.route('/analyze', methods=['POST'])
def analyze():
    video_url = request.form['video_url']
    video_id = extract_video_id(video_url)
    if not video_id:
        return jsonify({"error": "Invalid YouTube URL"})

    # Get number of comments from the form input
    max_comments = int(request.form.get('comment_count', 800))

    comments = fetch_comments(video_id, max_comments)
    relevant_comments = filter_comments(comments)
    polarity, positive_comments, negative_comments, neutral_comments =
analyze_sentiments(relevant_comments)

    avg_polarity = sum(polarity) / len(polarity)
    sentiment = "neutral"
    top_comments = neutral_comments[:5]
```

```
    if avg_polarity > 0.05:
        sentiment = "positive"
        top_comments = positive_comments[:5]
    elif avg_polarity < -0.05:
        sentiment = "negative"
        top_comments = negative_comments[:5]

    result = {
        "sentiment": sentiment,
        "top_comments": top_comments,
        "positive_count": len(positive_comments),
        "negative_count": len(negative_comments),
        "neutral_count": len(neutral_comments)
    }

    return jsonify(result)

if __name__ == '__main__':
    app.run(debug=True)
```

5.3.2 FRONT END CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>YouTube Comments Sentiment Analysis</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 20px;
            background-color: #f4f4f4;
        }
        h1, h2 {
            color: #333;
        }
        form {
            margin-bottom: 20px;
        }

        #progress-bar {
            width: 100%;
            background-color: #ddd;
            border-radius: 5px;
            display: none;
            margin-top: 10px;
        }
        #progress {
            width: 0%;
            height: 30px;
            background-color: #4caf50;
            text-align: center;
            line-height: 30px;
            color: white;
            border-radius: 5px;
        }

        .charts-container {
```

```
            display: flex;
            flex-wrap: wrap;
            justify-content: space-between;
        }

        .chart-wrapper {
            flex: 1;
            min-width: 300px;
            margin: 20px;
        }

        #top-comments {
            list-style-type: none;
            padding: 0;
        }

        #top-comments li {
            background: #fff;
            padding: 10px;
            margin-bottom: 5px;
            border-radius: 5px;
            box-shadow: 0 0 5px rgba(0,0,0,0.1);
        }

        #sentiment-summary {
            margin-top: 20px;
            font-size: 18px;
            font-weight: bold;
        }

        #download-button {
            margin-top: 20px;
            padding: 10px 20px;
            background-color: #4caf50;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            display: none; /* Hide initially */
        }

    </style>
</head>
<body>
    <h1>YouTube Comments Sentiment Analysis</h1>
    <form id="analyze-form">
        <label for="video_url">YouTube Video URL:</label>
        <input type="text" id="video_url" name="video_url" required>

        <label for="comment_count">Number of Comments to Fetch (Max 800):</label>
        <input type="number" id="comment_count" name="comment_count" min="1" max="800"
value="800">

        <input type="submit" value="Analyze">
    </form>

    <div id="progress-bar">
        <div id="progress">0%</div>
    </div>
    <div id="sentiment-summary"></div>
```

```
<h2 id="sentiment-title"></h2>
<ul id="top-comments"></ul>

<div class="charts-container">
  <div class="chart-wrapper">
    <h2>Sentiment Distribution:</h2>
    <canvas id="sentimentPieChart"></canvas>
  </div>

  <div class="chart-wrapper">
    <h2>Sentiment Count:</h2>
    <canvas id="sentimentBarChart"></canvas>
  </div>
</div>

<button id="download-button">Download PDF</button>
<!-- External JS libraries -->
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.4.0/jspdf.umd.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/html2canvas.min.js"></script>
<script>
  $('#analyze-form').on('submit', function(e) {
    e.preventDefault();
    $('#progress-bar').show();
    $('#progress').css('width', '0%').text('0%');
    $('#sentiment-summary').text('');
    $('#download-button').hide(); // Hide the download button until analysis is done
    var video_url = $('#video_url').val();
    var comment_count = $('#comment_count').val();
    $.ajax({
      url: '/analyze',
      type: 'POST',
      data: { video_url: video_url, comment_count: comment_count },
      xhr: function() {
        var xhr = new window.XMLHttpRequest();
        xhr.upload.addEventListener('progress', function(evt) {
          if (evt.lengthComputable) {
            var percentComplete = evt.loaded / evt.total;
            percentComplete = parseInt(percentComplete * 100);
            $('#progress').css('width', percentComplete + '%').text(percentComplete + '%');
          }
        }, false);
        return xhr;
      },
      success: function(data) {
        if (data.error) {
          alert(data.error);
        } else {
          var totalComments = data.positive_count + data.negative_count + data.neutral_count;
          var positivePercentage = ((data.positive_count / totalComments) * 100).toFixed(2);
          var negativePercentage = ((data.negative_count / totalComments) * 100).toFixed(2);
          var neutralPercentage = ((data.neutral_count / totalComments) * 100).toFixed(2);
          $('#sentiment-summary').text(
            'This video has received a ' + data.sentiment + ' response (' +
            positivePercentage + '% positive, ' +
            negativePercentage + '% negative, ' +
            neutralPercentage + '% neutral).'
          );
```

```
$('#sentiment-title').text('Top ' + data.sentiment.charAt(0).toUpperCase() +
data.sentiment.slice(1) + ' Comments:');
$('#top-comments').empty();
data.top_comments.forEach(function(comment, index) {
    $('#top-comments').append('<li>' + (index + 1) + '. ' + comment + '</li>');
});
$('#progress').css('width', '100%').text('100%');
// Show the download button after the analysis is done
$('#download-button').show();
// Pie chart
var ctxPie = document.getElementById('sentimentPieChart').getContext('2d');
var pieChart = new Chart(ctxPie, {
    type: 'pie',
    data: {
        labels: ['Positive', 'Negative', 'Neutral'],
        datasets: [{
            data: [data.positive_count, data.negative_count, data.neutral_count],
            backgroundColor: ['#4caf50', '#f44336', '#9e9e9e']
        }]
    }
});
// Bar chart
var ctxBar = document.getElementById('sentimentBarChart').getContext('2d');
var barChart = new Chart(ctxBar, {
    type: 'bar',
    data: {
        labels: ['Positive', 'Negative', 'Neutral'],
        datasets: [{
            label: 'Sentiment Count',
            data: [data.positive_count, data.negative_count, data.neutral_count],
            backgroundColor: ['#4caf50', '#f44336', '#9e9e9e']
        }]
    },
    options: {
        scales: {
            y: {
                beginAtZero: true
            }
        }
    }
});
// Download PDF on button click
$('#download-button').off('click').on('click', function() {
    var doc = new jspdf.jsPDF();
    // Capture the sentiment summary
    doc.setFontSize(12);
    doc.text(20, 20, $('#sentiment-summary').text());
    // Capture the top comments
    doc.setFontSize(18);
    doc.text(20, 40, 'Top Comments:');
    doc.setFontSize(14);
    $('#top-comments li').each(function(index, comment) {
        doc.text(20, 50 + (index * 10), (index + 1) + '. ' + $(comment).text());
    });
    // Capture the pie chart
    html2canvas(document.getElementById('sentimentPieChart')).then(function(canvas) {
        var imgData = canvas.toDataURL('image/png');
        doc.addImage(imgData, 'PNG', 20, 100, 100, 100);
        // Capture the bar chart
```

```
                html2canvas(document.getElementById('sentimentBarChart')).then(function(canvas) {
                    var imgData = canvas.toDataURL('image/png');
                    doc.addImage(imgData, 'PNG', 20, 200, 100, 100);
                    // Save the PDF
                    doc.save('sentiment_analysis.pdf');
                });
            });
        });
    }
},
error: function() {
    alert('An error occurred while processing the request.');
}
});
});
</script>
</body>
</html>
```

5.4 OUTPUT SCREENS

The below screen is displayed when you run the program in Pycharm and open the link.

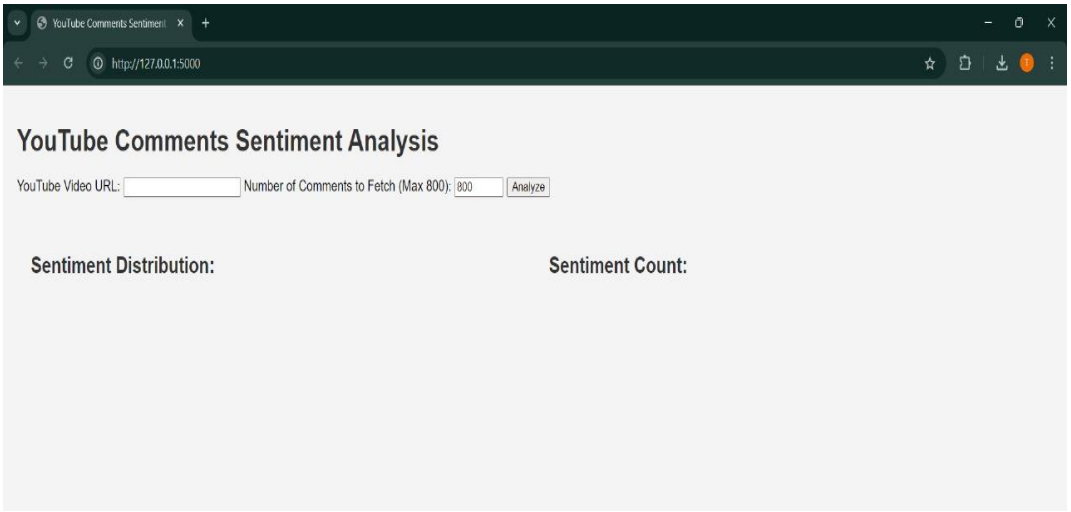

Fig 5.4.1 : Open the link in browser

The below screen is displayed when we pass the URL and the number of comments to system.

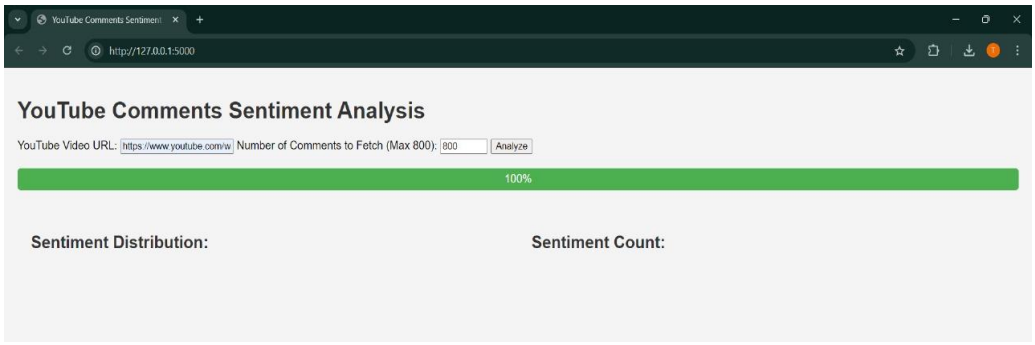

Fig 5.4.2 : Analyzing and Fetching comments

The below screen Displays the Top comments.

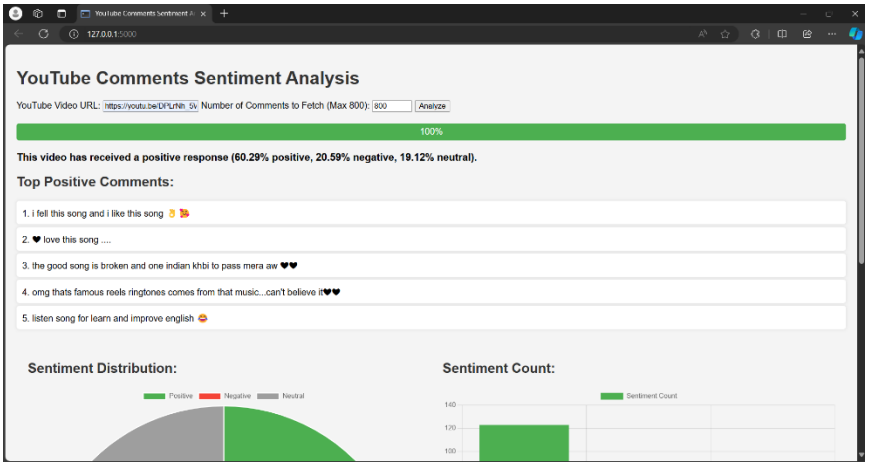

Fig 5.4.3: Top comments are Displayed

The below screen, Displays the generated report for the passed URL.

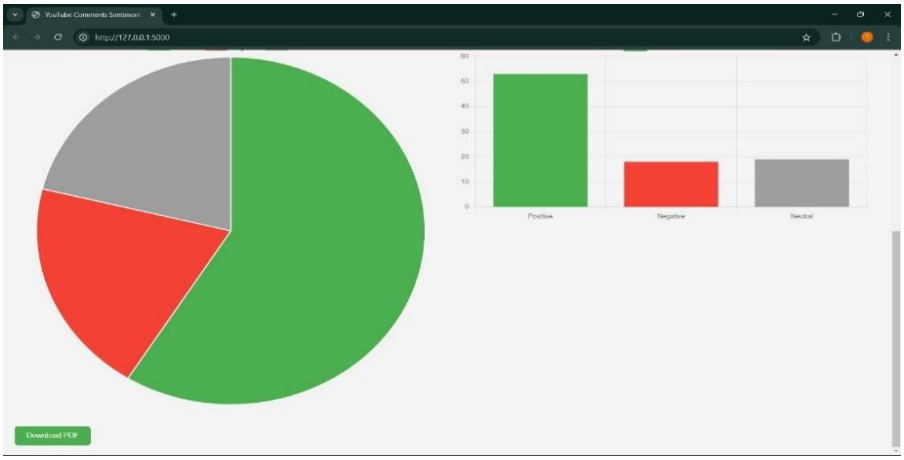

Fig 5.4.4: Report for the Video

The below Screen is shown when be click on Download button.

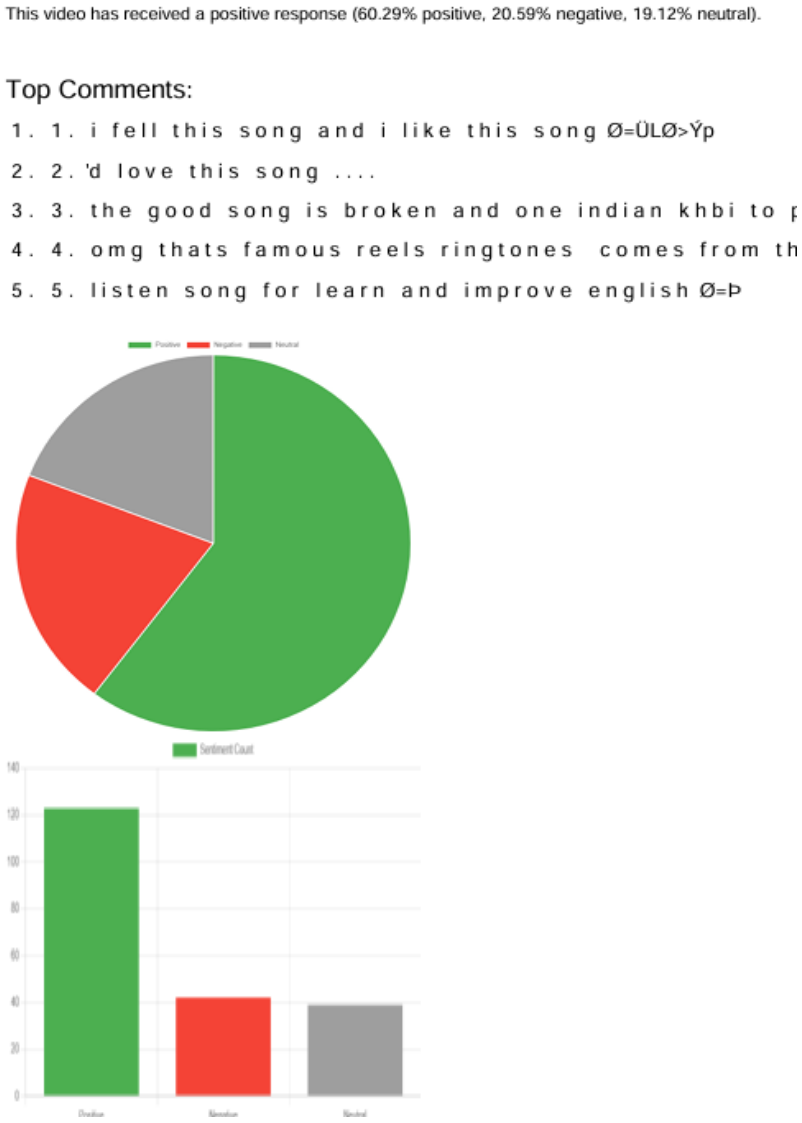

Fig 5.4.5 : Downloaded report

CHAPTER 6

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.1 TYPES OF TESTS

Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional Test

Functional tests provide systematic demonstrations that functions tested are available asspecified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs mustbe exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must beconsidered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. Ittests a configuration to ensure known and predictable results. An example of system testingis the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, It is purpose.It is used to test areas that cannot be reached from a black box level

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or

requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.2 VARIOUS TESTCASE SCENARIOS

Test case ID	**Test case Descrip-tion**	**Pre Condition**	**Test Case steps**	**Expected Result**	**Actual Result**	**Status**
TC01	https://youtu.be/cL4-YWStl3M?si=xRhf-vVmM9EwvPBo	User should be on the input page	1. Enter a valid YouTube URL 2. Click "Submit"	The system extracts the video ID, fetches comments, and displays sentiment results.	Fetches the video-id, comments, and display report.	Pass
TC02	https://youtu.be/0-p5EbAsxUM?si=4MZJyP6JA7DHlUQM	A valid YouTube video ID is extracted	1.The system sends a request to YouTube API 2. YouTube API returns comments.	The system successfully fetches up to 800 comments for the video.	Fetches the comments less than or equal to 800	Pass

TC03	Download the report	Sentiment result is displayed	1. User clicks "Download PDF" 2. PDF is generated	A PDF report containing sentiment results, charts, and top comments is downloaded.	It downloads the report	Pass
TC04	Emoji interpretation during sentiment analysis	Comments contain emojis	1. Comments with emojis are analyzed 2. System interprets emojis' emotional impact during sentiment analysis	Emojis are correctly factored into sentiment scores, enhancing accuracy.	Evaluates the emojis correctly	Pass
TC05	Selecting the number of comments to be fetched (eg: 800)	User submits The URL	1.User passes URL 2.Select the comments to 800	The system should evaluate for less than or equal to 800 comments	It fetches the comments correctly	Pass

Fig : 6.2.1 Testcases

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 PROJECT CONCLUSION

The project has successfully developed a Flask-based web application for performing sentiment analysis on YouTube video comments. The application leverages key Python libraries such as googleapiclient for accessing YouTube data, Vader Sentiment for sentiment analysis, and re for regular expression-based text processing. This allows the application to efficiently extract, process, and analyze comments. The core functionality includes extracting video IDs from URLs, fetching comments from the YouTube Data API, filtering out irrelevant or non-English comments, and analyzing sentiment to categorize comments as positive, negative, or neutral. The results are then presented to users in a clear JSON format, including the overall sentiment of the video, counts of positive, negative, and neutral comments, and a list of top comments from each sentiment category. This project demonstrates the effective integration of various technologies to create a user-friendly tool that provides valuable insights into viewer sentiment on YouTube. It showcases practical applications of web development, data processing, and sentiment analysis, offering a robust solution for understanding audience feedback and engagement on video content.

7.2 FUTURE ENHANCEMENT

In the next phase of this project, we are planning significant enhancements. These include the integration of advanced sentiment analysis techniques and machine learning models to gain deeper insights. We also aim to implement user authentication to personalize experiences and secure data, and to develop a backend database to store historical comment data for trend analysis. Additionally, we intend to enhance visualizations with interactive charts, expand language support, and improve error handling and application resilience to elevate overall functionality. Furthermore, we plan to optimize API requests to manage rate limits and enhance performance, as well as to improve data management and user feedback mechanisms to ensure a smoother and more scalable user experience.

CHAPTER 8

REFERENCES

8.1 WEBSITES

- www.google.com
- www.tutorialspoint.com
- www.geeksforgeeks.org
- ieeexplore.ieee.org
- en.wikipedia.org
- www.app.diagrams.net
- www.data-flair.com

8.2 PAPER REFERENCES

[1]Sainath Pichad, Sunit Kamble, Rohan Kalamb, Sumit Chavan, "Analysing Sentiments for YouTube Comments using Machine Learning", International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol 11 Issue-May 2023.

[2] Rawan Fahad Alhujaili, Wael M.S. Yafooz , "Sentiment Analysis for YouTube Videos with User Comments : Review'', International Conference on Artificial Intelligence and Smart Systems (ICAIS),Issue-2021.

[3] Byungkyu Yoo, Julia M Rayz , "Understanding Emojis for Sentiment Analysis'', The International FLAIRS Conference Proceedings , Vol.34 ,Issue-2021.

[4] Alexandre Ashade Lassance Cunha(B) ,Melissa Carvalho Costa, and Marco Aur´elio C. Pacheco, "Sentiment Analysis of YouTube Video Comments Using Deep Neural Networks", International Conference on Artificial Intelligence and Soft Computing, Issue-may 2019.

[5] Lambodara Parabhoi, Payel Saha, "Sentiment Analysis of YouTube Comments on Koha Open Source Software Videos'', International Journal of Library and Information Studies, Vol.8(4) Issue-Oct-Dec, 2018.