TABLE OF CONTENTS

CHAPTER 1:

INTRODUCTION

The rapid growth of social media sites like YouTube has led to an explosion of user-generated content, especially comments. Such comments are rich in information, trends and sentiments, which can be analyzed to derive insights into the opinions, feedback, and emotions of the users. This project, **YT Sentiment Explorer,** seeks to automate the process of extracting, processing, and analyzing comments from YouTube videos for meaningful conclusions.

The system uses the YouTube Data API to obtain comments and applies NLP-based preprocessing and sentiment analysis. It further uses libraries like Pandas and NLTK for data handling and analysis. The sentiments are classified into positive, negative and neutral categories to better understand user opinions. Visualization tools have also been included to display comment statistics and sentiment distribution graphically for better interpretation.

This project gives a scalable and user-friendly solution to analyze vast amounts of YouTube video comments, which can be helpful for content creators, marketers, and researchers to measure audience engagement, sentiment trends, and feedback. It saves time and reduces manual effort in the process, and it is accurate in the insights of the nature of user interactions on YouTube.

With the exponential growth of online content platforms, **YouTube** has emerged as one of the most prominent video-sharing platforms, hosting billions of videos and generating millions of user interactions daily in the form of comments. These comments reflect the thoughts, opinions, and emotions of viewers, making them a valuable resource for understanding audience sentiment and behavior. However, analyzing such vast amounts of unstructured textual data manually is impractical and time-consuming.

The **YT Sentiment Explorer** is a system designed to address this challenge by automating the extraction and analysis of sentiments expressed in user comments, providing valuable insights into viewer opinions and engagement.

This project leverages Natural Language Processing (NLP) technique to analyse YouTube comments. Using the Valence Aware Dictionary and sEntiment Reasoner (VADER) sentiment analysis tool from the Natural Language Toolkit (NLTK) library, the analyser classifies comments into positive, negative, or neutral categories based on their textual content.

The preprocessing pipeline involves converting text to lowercase, removing punctuation, special characters, stopwords, and applying lemmatization to normalize the text. The process begins by gathering comments from a specified YouTube video using the YouTube API.

These comments are then processed through the sentiment analysis pipeline. Each comment is assigned sentiment scores for positive, negative, and neutral sentiments, along with a compound score that indicates the overall sentiment and further categorizes each comment based on its compound score.

The YT Sentiment Explorer provides a user-friendly interface for visualizing the distribution of sentiments, identifying trends, and generating detailed reports. This tool is invaluable for content creators seeking to gauge audience reactions, marketers aiming to measure brand perception, and researchers studying online communication dynamics.

CHAPTER 2:

LITERATURE REVIEW

2.1 EXISTING PROJECT

The existing project for analysing YouTube comments are limited in functionality and efficiency. This project can only fetch comments from a single YouTube video at a time, which restricts their usability when working with multiple videos or larger datasets. Moreover, it is constrained by the following issues:

1. Limited Comment Extraction: The current project is only capable of fetching a maximum of **100 comments** per video. This limitation makes it ineffective for analysing videos with a large volume of comments, which are often required for meaningful sentiment analysis and trend identification.

2. Lack of Scalability: The inability to fetch comments from multiple videos prevents users from performing a comparative analysis or gaining insights across a group of related videos. For creators, businesses, or researchers who require large-scale analysis, this limitation proves to be a significant drawback.

3. Inability to Save Results: The existing project does not allow users to save the extracted comment data in a structured file format (e.g., CSV, Excel). This restricts users from storing, sharing, or further analysing the data using other tools or platforms.

4. User Experience Constraints: The lack of a user-friendly interface and flexible options for specifying the desired number of comments or video URLs further diminishes the usability of the existing system. Users cannot customize their input or output according to their requirements.

2.2 COMPARISON

To address the limitations of the existing project, our proposed project introduces a robust and scalable **YT Sentiment Explorer** that provides enhanced functionality for comment extraction, storage, and analysis. The proposed project overcomes the existing shortcomings with the following features:

1. Fetch Comments from any YouTube Video: Users can provide the **URL** of any YouTube video to fetch its comments. This ensures flexibility and allows the system to analyse data from any video on the platform.

2. Fetch Required Number of Comments: Unlike the existing project that limits comment extraction to only **100 comments**, the proposed project allows users to specify the **desired number of comments** to fetch. This feature ensures better coverage of comments, especially for videos with a large number of interactions.

3. Save Comments to CSV File: The extracted comments are **saved as a CSV file**, enabling users to store, share, and perform further analysis. This structured storage format ensures compatibility with other tools such as Excel, Google Sheets, and Python-based data analysis libraries.

4. Sentiment Analysis and Reports: The project performs **sentiment analysis** on the extracted comments to categorize them into three types: **Positive, Negative and Neutral Comments**

The project provides the following **overall results**:

- Count of Total comments of a video, Positive, Negative, and Neutral comments.
- Accuracy of the sentiment analysis model.
- Comprehensive reports summarizing the sentiment distribution.

5. User-Friendly Implementation: The project is implemented in a way that is easy to use, where the user simply inputs the YouTube Video URL, the desired number of comments and the file name to be saved as. The tool then automates the process of fetching comments, analysing them, and saving the results.

CHAPTER 3:

PROPOSED PROJECT

3.1 PROJECT SCOPE

The scope of this project is defined to ensure that the system provides a comprehensive solution for fetching, analyzing, and reporting YouTube video comments. The project aims to deliver a tool that is scalable, efficient, and user-friendly for users seeking insights into audience sentiments from YouTube comments.

In-Scope Features:

1. YouTube Comment Extraction:

- ➢ Fetch comments from any YouTube video using its URL.
- ➢ Allow users to define the number of comments to be extracted.

2. Flexible Comment Limit:

- ➢ Users can fetch up to thousands of comments (based on API limitations and availability).

3. Data Storage:

- ➢ Save the fetched comments into a CSV file.
- ➢ Ensure the stored file is well-structured and compatible with data analysis tools like Excel, Pandas, and other data visualization libraries.

4. Sentiment Analysis:

- ➢ Analyse the extracted comments using sentiment analysis techniques.
- ➢ Categorize the comments into three sentiment types:
    - ▪ Positive
    - ▪ Negative
    - ▪ Neutral

5. Summary Reports:

- Generate overall sentiment distribution reports, including:
  - Total number of comments analysed.
  - Count of positive, negative, and neutral comments.
  - Sentiment analysis accuracy.
- Provide insights for understanding user engagement and audience reactions.

6. User-Friendly Interface:

- Allow users to interact with the system by providing simple inputs (like YouTube Video URL, desired comment count and file name).
- Automate the process of fetching, analysing, and saving results.

3.2 HIGH-LEVEL ARCHITECTURE

The High-Level Architecture (HLA) of the project provides a conceptual view of the components, their interactions, and how the system achieves its functionalities. Below is the detailed High-Level Architecture:

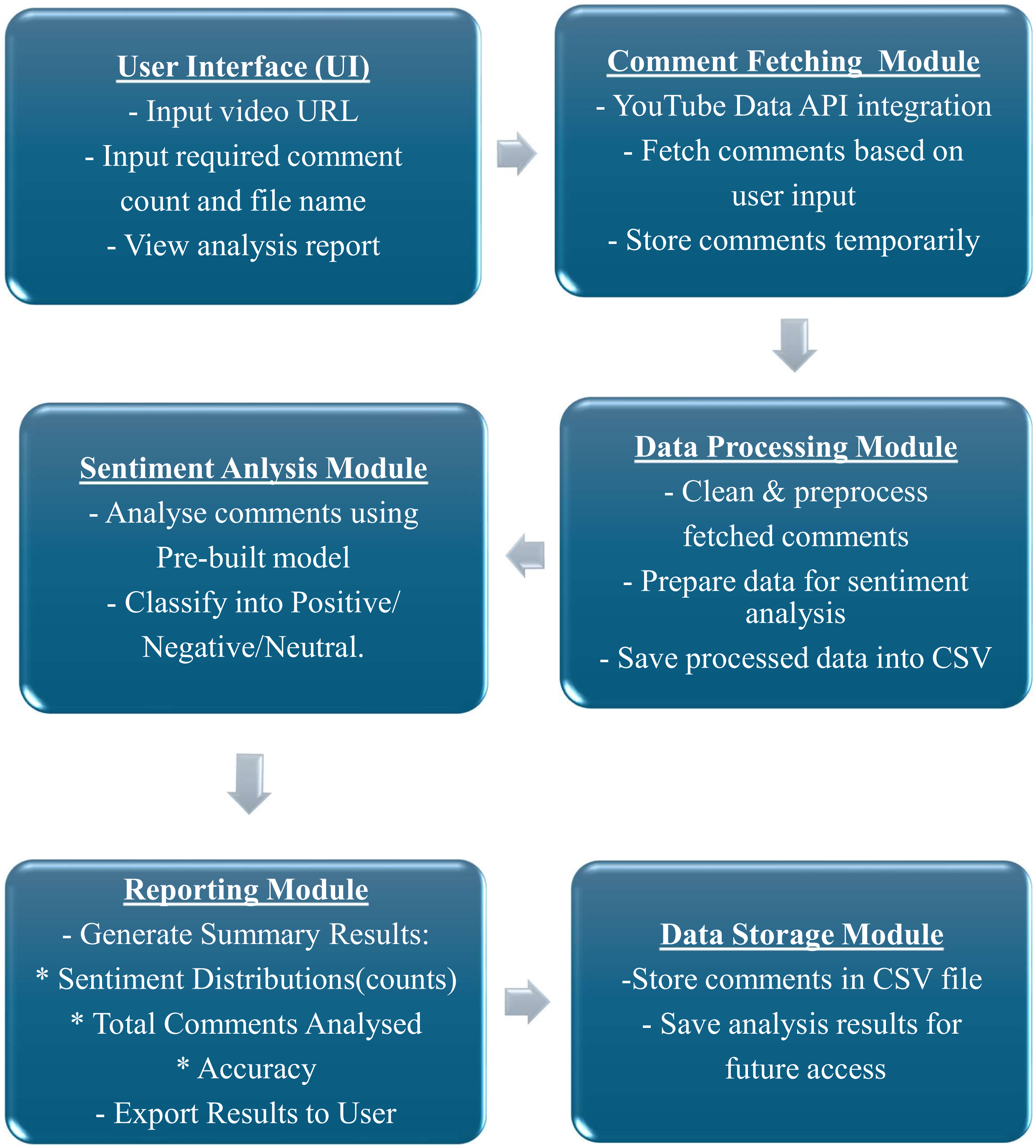

User Interface (UI)
- Input video URL
- Input required comment count and file name
- View analysis report
Comment Fetching Module
- YouTube Data API integration
- Fetch comments based on user input
- Store comments temporarily
Data Processing Module
- Clean & preprocess fetched comments
- Prepare data for sentiment analysis
- Save processed data into CSV
Sentiment Anlysis Module
- Analyse comments using Pre-built model
- Classify into Positive/ Negative/Neutral.
Reporting Module
- Generate Summary Results:
* Sentiment Distributions(counts)
* Total Comments Analysed
* Accuracy
- Export Results to User
Data Storage Module
-Store comments in CSV file
- Save analysis results for future access

Fig: High-Level Architecture

3.3 OBJECTIVE OF THE STUDY

The primary objective of this project is to develop an automated system for fetching, analysing, and reporting sentiment from YouTube video comments. This project aims to address the limitations of existing solutions by improving the accuracy, scalability, and usability of comment analysis.

The specific objectives of the study are as follows:

1. **Automated Comment Retrieval**
   - ➢ To retrieve comments from any YouTube video using its URL.
   - ➢ To allow users to specify the desired number of comments to fetch.

2. **Efficient Data Processing**
   - ➢ To clean and preprocess the fetched comments by removing unwanted characters, symbols, and stop words for better analysis.
   - ➢ To store the comments in a structured CSV file for further use or archival purposes.

3. **Sentiment Analysis**
   - ➢ To perform sentiment classification on the fetched comments using Natural Language Processing (NLP) techniques.
   - ➢ To classify comments into positive, negative, and neutral categories.
   - ➢ To compute summary metrics, such as sentiment counts and overall accuracy.

4. **Result Reporting**
   - ➢ To generate a report summarizing the sentiment analysis results.
   - ➢ To present key metrics such as:
     - ▪ Total number of comments analysed.
     - ▪ Number of positive, negative, and neutral comments.
     - ▪ Overall accuracy of the analysis.

5. **Enhanced User Usability**
   - ➢ To design a simple and user-friendly interface that allows easy input of video URL and comment count.
   - ➢ To facilitate seamless export of the analysis results into a CSV file for future reference.

CHAPTER 4:

PROJECT ANALYSIS

4.1 SYSTEM ANALYSIS

System analysis is a critical phase of project development that involves understanding the functional and non-functional requirements, limitations of existing solutions, and the enhancements introduced in the proposed system. It ensures a comprehensive understanding of the system's architecture and functionalities.

4.2 THE EXISTING PROJECT

The existing project has the following limitations:

1. **Restricted Data Fetching:**
   - Only fetches comments from a single video at a time.
   - Limited to retrieving 100 comments per video, which is insufficient for large-scale sentiment analysis.

2. **Limited File Handling:**
   - The Comments are not saved in user-specified formats or filenames

3. **Basic Sentiment Analysis**:
   - Provides only a minimal categorization of sentiments without detailed insights or metrics such as sentiment distribution or accuracy.

4. **Lack of Flexibility**:
   - Input parameters like the number of comments to be fetched or output customization are not supported.

4.3 THE PROPOSED PROJECT

The proposed project addresses these challenges by introducing significant enhancements:

1. **Dynamic Comment Fetching:**
   - Allows fetching comments from any YouTube video by providing its Video URL.
   - Supports a user-defined number of comments to be retrieved, ensuring scalability.

2. **Enhanced Data Management:**
   - Saves fetched comments into a CSV file with user-defined filenames for better organization and accessibility.

3. **Advanced Sentiment Analysis:**
   - Categorizes comments into positive, negative, and neutral sentiments.
   - Provides detailed metrics such as:
     - Sentiment distribution (positive, negative, neutral counts).
     - Accuracy of the sentiment analysis.

4. **User Interaction:**
   - Incorporates a user-friendly interface for input parameters like video URL and the number of comments.
   - Displays comprehensive results, including sentiment counts, accuracy, and distribution reports.

4.4 HARDWARE AND SOFTWARE SELECTION

The proper selection of appropriate hardware and software components determines the efficient development and smooth operation of the project. Below is a detailed overview of the hardware and software requirements for implementing this project.

Hardware Requirements:

Component	Specification
Processor	Intel Core i3 or higher / AMD Ryzen 5 or higher
RAM	Minimum 8 GB (16 GB recommended for large datasets)
Storage	Minimum 250 GB SSD (500 GB recommended for better performance)
Graphics	Integrated graphics sufficient (dedicated GPU optional)
Network	Stable internet connection for accessing YouTube API

Software Requirements:

Component	Details
Operating System	Windows 8,10/11, macOS
Programming Environment	Visual Studio Code (or any preferred IDE)
Programming Language	Python 3.9 or later
APIs	YouTube Data API v3 for fetching comments
Libraries and Frameworks	Required Python libraries: - googleapiclient for API integration - pandas for data manipulation - nltk for sentiment analysis - matplotlib for data visualization - numpy for numerical computations
Browser	Google Chrome, Mozilla Firefox, or Microsoft Edge
Package Manager	pip (Python's package installer)

4.5 FUNCTIONAL REQUIREMENTS

The functional requirements define the core functionalities that the system must fulfil to meet the project objectives, they are:

1. **Comment Fetching**
   - Allow the user to input a YouTube video URL to fetch comments.
   - Enable the user to specify the number of comments to fetch.

2. **Data Storage**
   - Save the fetched comments as a CSV file with a user-defined name.

3. **Sentiment Analysis**
   - Analyse the sentiment of each comment using sentiment analysis techniques.
   - Categorize comments into Positive, Negative, and Neutral.

4. **Data Visualization and Reporting**
   - Provide the overall sentiment distribution as a report, including:
     - Count of Positive, Negative, and Neutral comments.
     - Accuracy of the sentiment analysis model.

5. **Frontend Interaction**
   - Provide an easy-to-use graphical interface for input and output operations.

4.6 EXTERNAL INTERFACE REQUIREMENTS

The external interface requirements describe how the system interacts with users, APIs, and external systems.

1. **User Interface (UI)**

   - A simple input form for entering:
     - YouTube video URL.
     - The desired number of comments to fetch.
     - File name for saving the comments as a CSV file.
   - Output reports displayed in a graphical format.

2. Hardware Interface

- The system requires a stable internet connection to communicate with the YouTube Data API.

3. Software Interface

- YouTube Data API v3:
  - Fetch comments using the API.
  - Ensure proper authentication through an API key.

4. Communication Interfaces

- The system uses HTTP protocols to interact with YouTube Data API for fetching data.

4.7 USER INTERFACE

The user interface should be designed to ensure ease of use and efficiency. Below are the proposed components of the UI:

1.Input Section

- **Fields**:
  - Textbox to input the YouTube video URL.
  - Numeric input for the number of comments to fetch.
  - Textbox for specifying the file name to save the CSV.
- **Buttons**:
  - An "Analyse" button to start the process.

2. Output Section

- **Sentiment Analysis Report**:
  - Pie chart to represent the sentiment distribution.
  - A summary showing counts for each sentiment category.
  - Display the model accuracy percentage.
- **Option for downloading the CSV file.**
- **Option for analysing another video.**

CHAPTER 5:

DESIGN

5.1 DATA DESIGN

The data design defines how the system organizes and stores data during the fetching, analysis, and reporting processes.

1. **Input Data:**
   - Video URL, number of comments, file name.

2. **Raw Data:**
   - Comments fetched from the YouTube Data API stored temporarily for processing.

3. **Processed Data:**
   - Sentiment scores and classifications (Positive, Negative, Neutral).
   - Count of each sentiment category.

4. **Output Data:**
   - CSV file containing comments and sentiment labels.
   - Sentiment distribution summary.

5.2 DATA DEFINITION

Input Fields

Field	**Type**	**Description**
Video URL	String	URL of the YouTube video
No. of Comments	Integer	Max. number of comments to fetch
File Name	String	Name of the CSV file to save comments

Output Fields

Field	Type	Description
Video URL	String	URL of the given YouTube Video
Sentiment	String	Sentiment label: Positive, Negative, Neutral

5.3 ARCHITECTURAL DESIGN

The system follows a three-tier architecture:

1. **Presentation Layer (Frontend):**
   - User interface for input, visualization, and reports.

2. **Application Layer (Backend):**
   - Python code to fetch comments, analyze sentiment, and process data.

3. **Data Layer:**
   - CSV files for temporary data storage.
   - Option for future integration with relational databases.

5.4 INTERFACE DESIGN

➢ **YouTube Data API Interface**

- **Purpose:** Fetch comments from YouTube.

- **KeyMethod:** commentThreads().list() for retrieving comments.

➢ **System Interfaces**

- **Input Interface:** Takes video URL, number of comments, and file name from the user.
- **Output Interface:** Saves results as a CSV file and generates graphical sentiment reports.

5.5 USER INTERFACE DESIGN

- **User Inputs**
  - **Form Components:**
    - Textbox for video URL.
    - Number input for the count of comments.
    - Textbox for file name.

- **User Outputs**
  - Displays the given URL
  - Accuracy of Sentiment Analysis
  - Sentiment Distribution (Positive, Negative & Neutral)
  - **Graphical Reports:**
    - **Pie chart showing sentiment distribution.**

5.6 PROCEDURAL DESIGN

The procedural design describes the step-by-step process for each functionality.

- **Fetch Comments**
  - User inputs video URL, number of comments, and file name.
  - System fetches comments using YouTube Data API.
  - Comments are stored temporarily in memory.
- **Sentiment Analysis**
  - Apply sentiment analysis to each comment.
  - Assign sentiment labels based on sentiment scores.

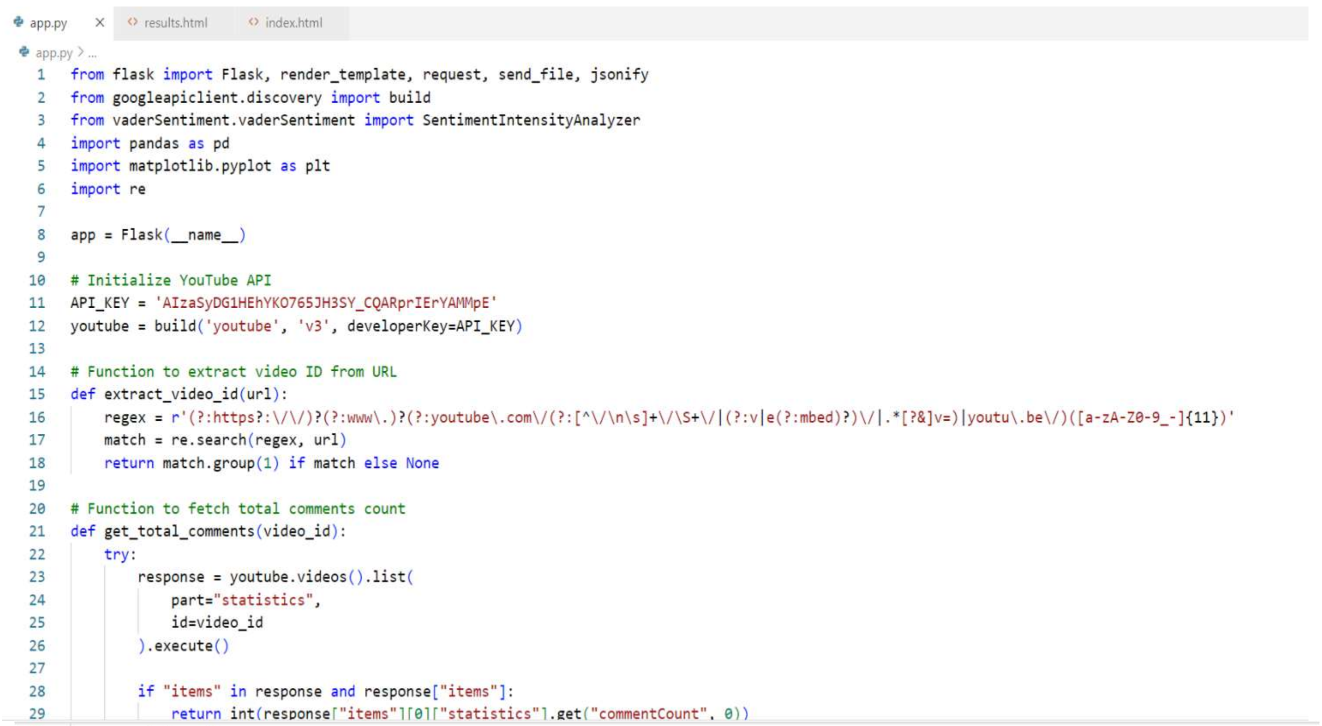
- **Save and Report**
  - Save the comments and sentiments in a CSV file.
  - Generate a summary report with sentiment counts and accuracy.
- **Visualization**
  - Plot sentiment distribution using graphs.

This design ensures clarity, modularity, and ease of implementation for each system component.

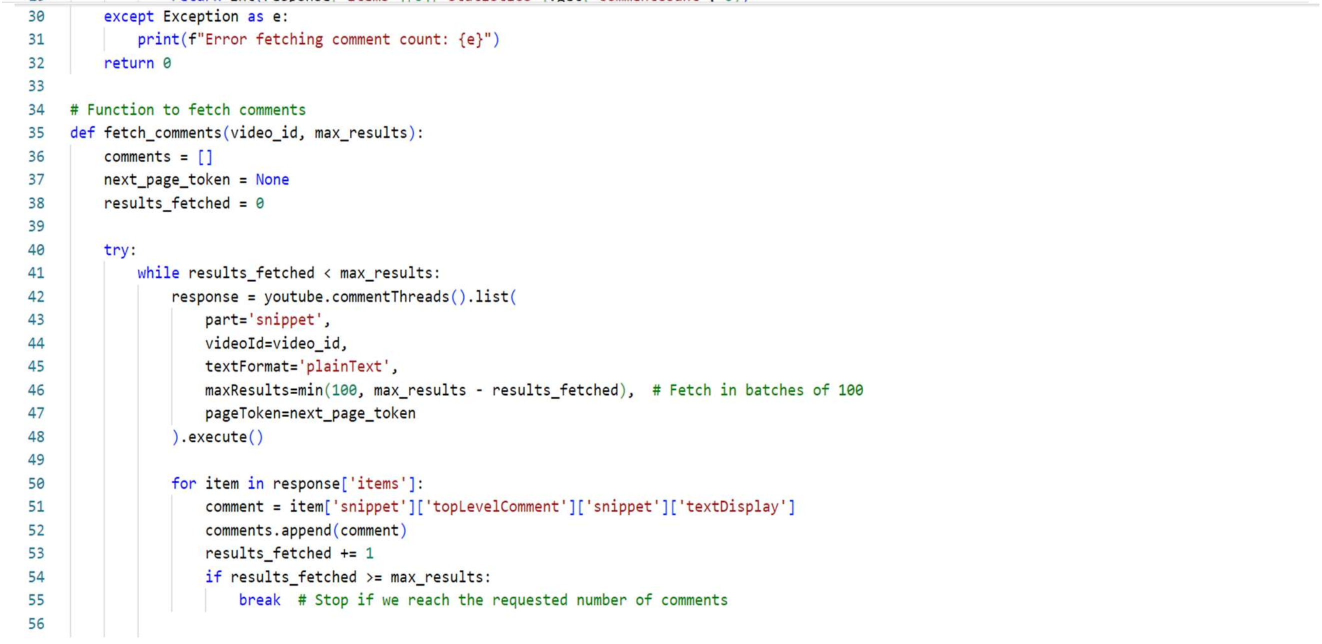
CHAPTER 6:

PSEUDO CODE

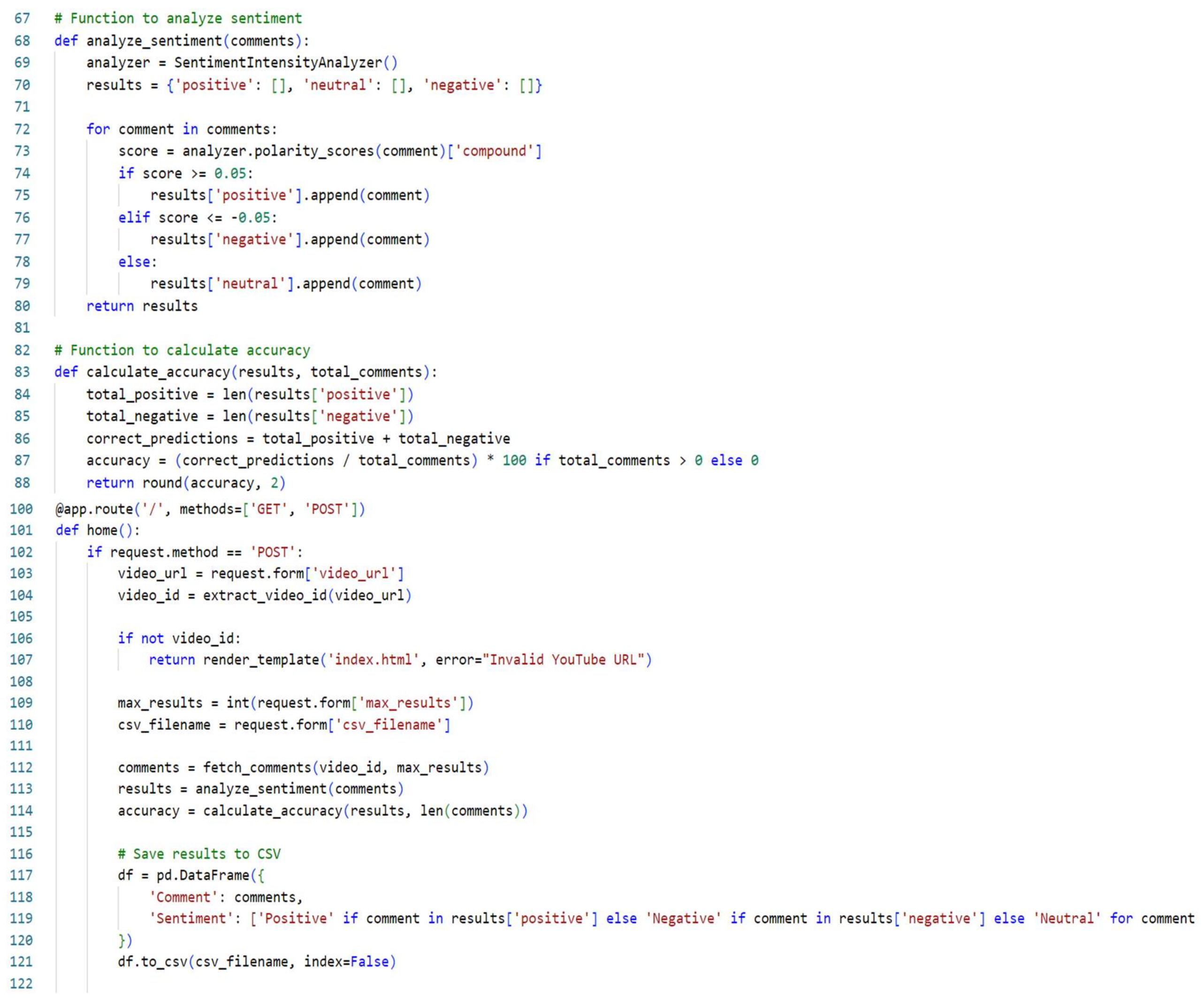
1. Sentiment Analysis Code (app.py)



```python
from flask import Flask, render_template, request, send_file, jsonify
from googleapiclient.discovery import build
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
import pandas as pd
import matplotlib.pyplot as plt
import re

app = Flask(__name__)

# Initialize YouTube API
API_KEY = 'AIzaSyDG1HEhYKO765JH3SY_CQARprIErYAMMpE'
youtube = build('youtube', 'v3', developerKey=API_KEY)

# Function to extract video ID from URL
def extract_video_id(url):
    regex = r'(?:https?:\/\/)?(?:www\.)?(?:youtube\.com\/(?:[^\/\n\s]+\/\S+\/|(?:v|e(?:mbed)?)\/|.*[?&]v=)|youtu\.be\/)([a-zA-Z0-9_-]{11})'
    match = re.search(regex, url)
    return match.group(1) if match else None

# Function to fetch total comments count
def get_total_comments(video_id):
    try:
        response = youtube.videos().list(
            part="statistics",
            id=video_id
        ).execute()

        if "items" in response and response["items"]:
            return int(response["items"][0]["statistics"].get("commentCount", 0))
```

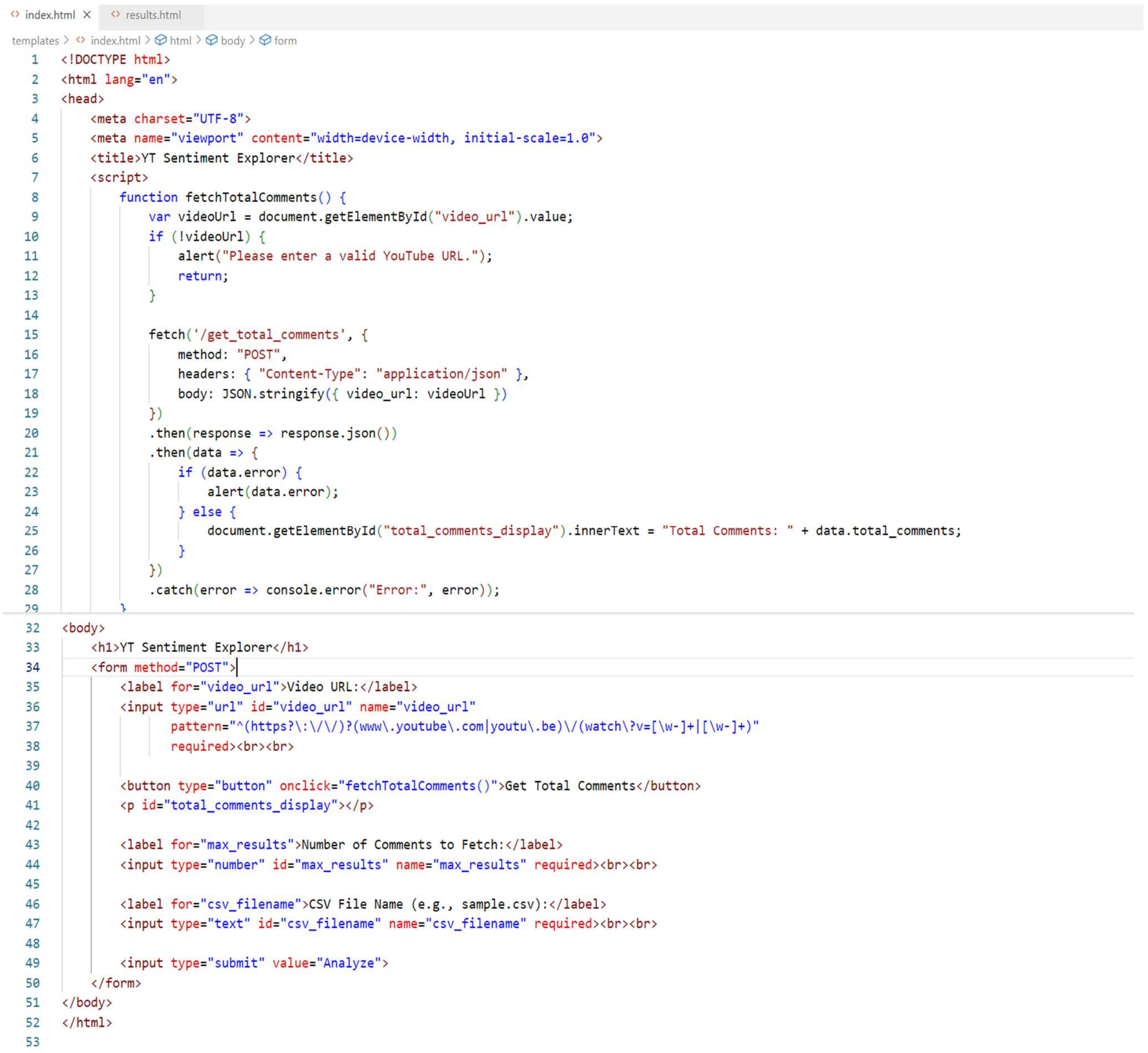


```python
    except Exception as e:
        print(f"Error fetching comment count: {e}")
    return 0

# Function to fetch comments
def fetch_comments(video_id, max_results):
    comments = []
    next_page_token = None
    results_fetched = 0

    try:
        while results_fetched < max_results:
            response = youtube.commentThreads().list(
                part='snippet',
                videoId=video_id,
                textFormat='plainText',
                maxResults=min(100, max_results - results_fetched),  # Fetch in batches of 100
                pageToken=next_page_token
            ).execute()

            for item in response['items']:
                comment = item['snippet']['topLevelComment']['snippet']['textDisplay']
                comments.append(comment)
                results_fetched += 1
                if results_fetched >= max_results:
                    break  # Stop if we reach the requested number of comments

```



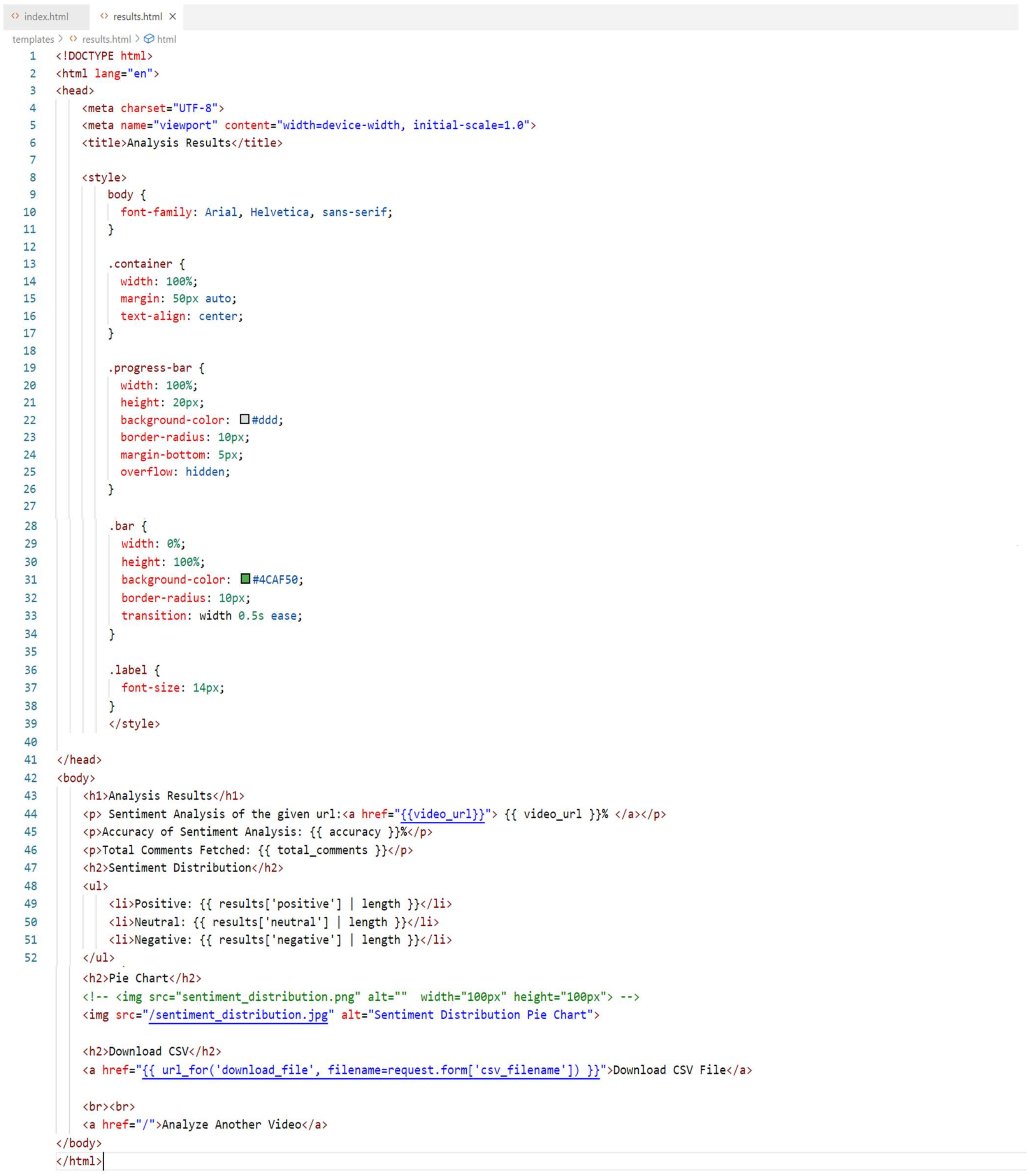
```python
# Function to analyze sentiment
def analyze_sentiment(comments):
    analyzer = SentimentIntensityAnalyzer()
    results = {'positive': [], 'neutral': [], 'negative': []}

    for comment in comments:
        score = analyzer.polarity_scores(comment)['compound']
        if score >= 0.05:
            results['positive'].append(comment)
        elif score <= -0.05:
            results['negative'].append(comment)
        else:
            results['neutral'].append(comment)
    return results

# Function to calculate accuracy
def calculate_accuracy(results, total_comments):
    total_positive = len(results['positive'])
    total_negative = len(results['negative'])
    correct_predictions = total_positive + total_negative
    accuracy = (correct_predictions / total_comments) * 100 if total_comments > 0 else 0
    return round(accuracy, 2)
@app.route('/', methods=['GET', 'POST'])
def home():
    if request.method == 'POST':
        video_url = request.form['video_url']
        video_id = extract_video_id(video_url)

        if not video_id:
            return render_template('index.html', error="Invalid YouTube URL")

        max_results = int(request.form['max_results'])
        csv_filename = request.form['csv_filename']

        comments = fetch_comments(video_id, max_results)
        results = analyze_sentiment(comments)
        accuracy = calculate_accuracy(results, len(comments))

        # Save results to CSV
        df = pd.DataFrame({
            'Comment': comments,
            'Sentiment': ['Positive' if comment in results['positive'] else 'Negative' if comment in results['negative'] else 'Neutral' for comment
        })
        df.to_csv(csv_filename, index=False)

```

2. Index.html



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>YT Sentiment Explorer</title>
    <script>
        function fetchTotalComments() {
            var videoUrl = document.getElementById("video_url").value;
            if (!videoUrl) {
                alert("Please enter a valid YouTube URL.");
                return;
            }

            fetch('/get_total_comments', {
                method: "POST",
                headers: { "Content-Type": "application/json" },
                body: JSON.stringify({ video_url: videoUrl })
            })
            .then(response => response.json())
            .then(data => {
                if (data.error) {
                    alert(data.error);
                } else {
                    document.getElementById("total_comments_display").innerText = "Total Comments: " + data.total_comments;
                }
            })
            .catch(error => console.error("Error:", error));
        }
<body>
    <h1>YT Sentiment Explorer</h1>
    <form method="POST">
        <label for="video_url">Video URL:</label>
        <input type="url" id="video_url" name="video_url"
               pattern="^(https?\:\/\/)?(www\.youtube\.com|youtu\.be)\/(watch\?v=[\w-]+|[\w-]+)"
               required><br><br>

        <button type="button" onclick="fetchTotalComments()">Get Total Comments</button>
        <p id="total_comments_display"></p>

        <label for="max_results">Number of Comments to Fetch:</label>
        <input type="number" id="max_results" name="max_results" required><br><br>

        <label for="csv_filename">CSV File Name (e.g., sample.csv):</label>
        <input type="text" id="csv_filename" name="csv_filename" required><br><br>

        <input type="submit" value="Analyze">
    </form>
</body>
</html>
```

3. Results.html



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Analysis Results</title>

    <style>
        body {
          font-family: Arial, Helvetica, sans-serif;
        }

        .container {
          width: 100%;
          margin: 50px auto;
          text-align: center;
        }

        .progress-bar {
          width: 100%;
          height: 20px;
          background-color: #ddd;
          border-radius: 10px;
          margin-bottom: 5px;
          overflow: hidden;
        }

        .bar {
          width: 0%;
          height: 100%;
          background-color: #4CAF50;
          border-radius: 10px;
          transition: width 0.5s ease;
        }

        .label {
          font-size: 14px;
        }
        </style>

</head>
<body>
    <h1>Analysis Results</h1>
    <p> Sentiment Analysis of the given url:<a href="{{video_url}}"> {{ video_url }}% </a></p>
    <p>Accuracy of Sentiment Analysis: {{ accuracy }}%</p>
    <p>Total Comments Fetched: {{ total_comments }}</p>
    <h2>Sentiment Distribution</h2>
    <ul>
        <li>Positive: {{ results['positive'] | length }}</li>
        <li>Neutral: {{ results['neutral'] | length }}</li>
        <li>Negative: {{ results['negative'] | length }}</li>
    </ul>
    <h2>Pie Chart</h2>
    <!-- <img src="sentiment_distribution.png" alt=""  width="100px" height="100px"> -->
    <img src="/sentiment_distribution.jpg" alt="Sentiment Distribution Pie Chart">

    <h2>Download CSV</h2>
    <a href="{{ url_for('download_file', filename=request.form['csv_filename']) }}">Download CSV File</a>

    <br><br>
    <a href="/">Analyze Another Video</a>
</body>
</html>
```

CHAPTER 7:

PROJECT FLOW

Flow of the YT Sentiment Explorer Project is as follows:

1. User Input

- ➢ The user enters a YouTube video URL in the web interface.
- ➢ The user specifies the number of comments to fetch.
- ➢ The user provides a filename to save the results in CSV format.

2. Extract Video URL

- ➢ The system extracts the YouTube Video ID from the provided URL using regex.
- ➢ If the URL is invalid, an error message is displayed.

3. Fetch Comments

- ➢ The system connects to the YouTube API using the provided API key.
- ➢ It fetches the specified number of top-level comments from the video.

4. Perform Sentiment Analysis

- ➢ Each comment is processed using VADER Sentiment Analysis to classify it as:
  - ▪ Positive (score ≥ 0.05)
  - ▪ Neutral ($-0.05 <$ score < 0.05)
  - ▪ Negative (score ≤ -0.05)

5. Calculate Metrics

- ➢ The system counts the number of positive, negative, and neutral comments.
- ➢ It calculates the accuracy of sentiment classification.

6. Save & Display Results

- The system saves the comments along with their sentiment labels in a CSV file.
- The results are displayed on the web page, including:
  - Total Positive, Neutral, and Negative comments.
  - Overall Accuracy.
  - A Pie Chart showing sentiment distribution.

7. Download CSV File

- The user can download the CSV file containing the comments and their sentiments.

8. Error Handling

- If an incorrect URL is entered, the system prompts an error message.

The technologies & tools used in this project are:

1. Natural Language Processing (NLP)

Natural Language Processing (NLP) is a branch of Artificial Intelligence (AI) that enables computers to understand, interpret, and generate human language. It involves various techniques such as tokenization, sentiment analysis, named entity recognition, text classification, and more.

- **Sentiment Analysis using NLP**
  - Sentiment Analysis determines whether a text expresses **positive, negative, or neutral** sentiment.
  - In our project, we use **VADER (Valence Aware Dictionary and sEntiment Reasoner)** for sentiment classification.

2. VADER (Valence Aware Dictionary and sEntiment Reasoner)

VADER is a rule-based sentiment analysis tool specifically designed for social media text, reviews, and comments. It is widely used in Natural Language Processing (NLP) for sentiment classification. It is used to analyze the sentiment of YouTube comments and classify them as positive, negative, or neutral.

VADER has a predefined sentiment lexicon, which assigns words predefined sentiment scores.

➢ **Lexicon-Based Sentiment Scoring**

VADER uses a predefined dictionary of words with associated sentiment scores. Each word in the comment is assigned a score:

- Positive words (e.g., "awesome", "love") → +ve score
- Negative words (e.g., "terrible", "hate") → -ve score
- Neutral words (e.g., "the", "video") → 0 score

Example:
"This video is amazing!"
"amazing" has a high positive score → Overall sentiment: Positive

➢ VADER calculates a compound score between -1 (negative) to +1 (positive) based on the overall sentiment of the sentence.

- Positive Sentiment → Compound score ≥ 0.05
- Negative Sentiment → Compound score ≤ -0.05
- Neutral Sentiment → Compound score between -0.05 and 0.05

Example:

- "I love this video, it's fantastic!" → Positive (0.75)
- "This video is not great, it's boring." → Negative (-0.55)
- "This video is okay." → Neutral (0.0)

- **Features of VADER**
  - Understands Emojis & Slang (e.g., 😊 = positive, "lol" = positive)
  - Handles Negation ("not good" is negative)
  - Considers Capitalization & Punctuation ("GREAT!!!" is stronger than "great"). More exclamation marks increase positivity.
  - Optimized for Social Media & Short Texts

- **Why is VADER used in our project?**
  - Fast & Efficient: No need for large machine learning models.
  - Rule-Based & Pre-Trained: No extra training required.
  - Best for Social Media Comments: Works well with YouTube comments.
  - Accurate for Short Texts: Since YouTube comments are short, VADER performs well.

3. NLTK (Natural Language Toolkit)

NLTK (Natural Language Toolkit) is one of the most widely used Python libraries for Natural Language Processing (NLP). It provides tools for text processing, tokenization, stemming, lemmatization, sentiment analysis, and more.

- **NLTK components used in our project**

1. Sentiment Analysis (VADER)

- The SentimentIntensityAnalyzer from nltk.sentiment.vader is used to analyze YouTube comments.
- It classifies comments as positive, negative, or neutral based on a compound sentiment score.

2. Text Processing

- NLTK provides tools for tokenization (splitting text into words/sentences), removing stopwords, and handling negations.
- This helps in improving comment analysis accuracy.

➢ Why NLTK in our project?

- Pre-trained Models: No need to train custom models.
- Lightweight: Works well for short YouTube comments.
- Optimized for Sentiment Analysis: Includes VADER, which is fine-tuned for social media and reviews.
- Easy to Implement: Simple functions for tokenization, stemming, and sentiment analysis.

➢ Other NLTK Features you can use

- Tokenization: Splitting comments into words.
- Stopword Removal: Removing common words like "is," "the," etc.
- Stemming & Lemmatization: Converting words to their root form
- POS Tagging: Identifying verbs, nouns, adjectives in comments
- Named Entity Recognition (NER): Identifying proper nouns, places, and names

CHAPTER 8:

TESTING

Testing is a crucial phase in the software development lifecycle, ensuring that the developed system meets specified requirements and performs as expected. This section outlines the testing strategies, methodologies, and results applied to the **YT Sentiment Explorer**.

8.1 Testing Objectives

The primary objectives of testing in this project are:

- To verify that the system fetches comments accurately from any provided YouTube video URL.
- To ensure the system retrieves the specified number of comments.
- To validate the ability to save comments in CSV format.
- To check the accuracy of sentiment analysis and report generation.
- To confirm that the web interface is user-friendly and error-free.
- To identify and fix any functional or performance issues.

8.2 System Testing

- **Objective:** To test the entire system under real-world conditions to verify compliance with functional requirements.

- **Test Cases:**
  - Test the end-to-end process from video URL input to report generation.
  - Evaluate response time for fetching and processing large data sets.
  - Verify storage of reports and logs.

8.3 Test Case Scenarios

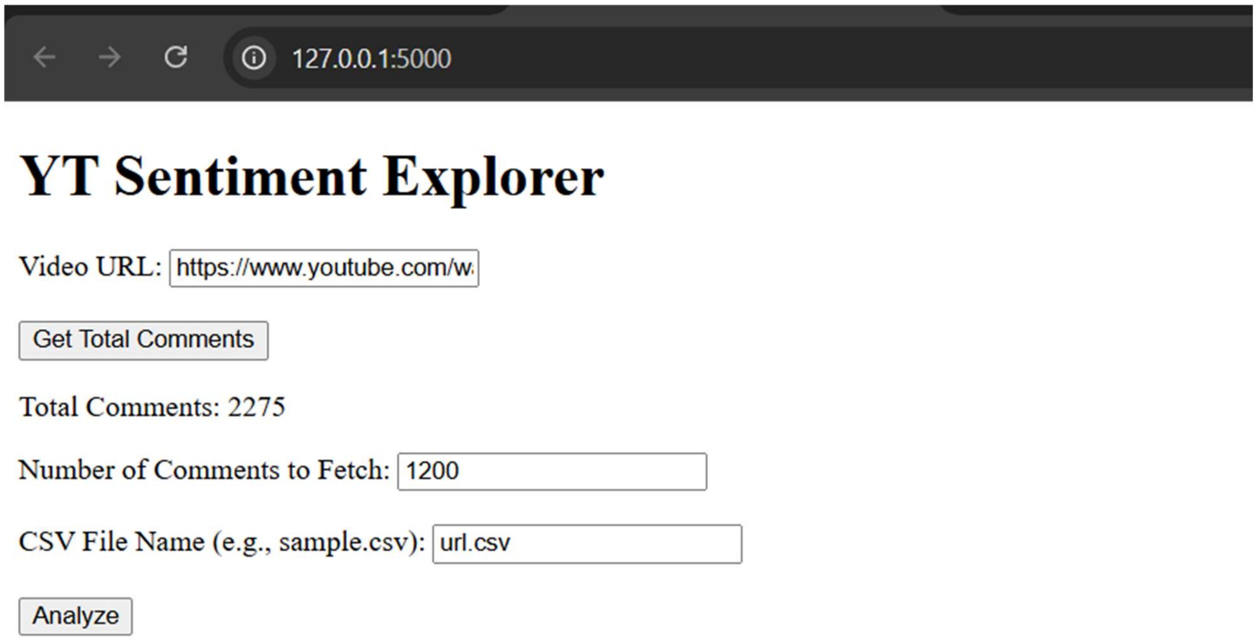
The following table presents key test scenarios and their expected outcomes:

Test ID	Test Scenario	Input	Expected Output	Status
TC_01	Valid YouTube video URL	URL: youtube.com/ watch?v=abc123	Comments fetched successfully	Pass
TC_02	Invalid YouTube video URL	URL: youtube.com/ watch?v=xyz456	Error message displayed	Pass
TC_03	CSV file generation	Button click	File downloaded successfully	Pass
TC_04	Sentiment analysis accuracy	Sample comments	Correct sentiment categorization	Pass
TC_05	System performance	Large dataset	Process within acceptable time	Pass

8.4 Conclusion

Based on the results of various tests conducted, the system meets the functional and performance requirements set during the planning phase. The project is now ready for deployment with confidence in its accuracy, efficiency, and user-friendliness.

RESULTS



YT Sentiment Explorer

Video URL: https://www.youtube.com/w

Get Total Comments

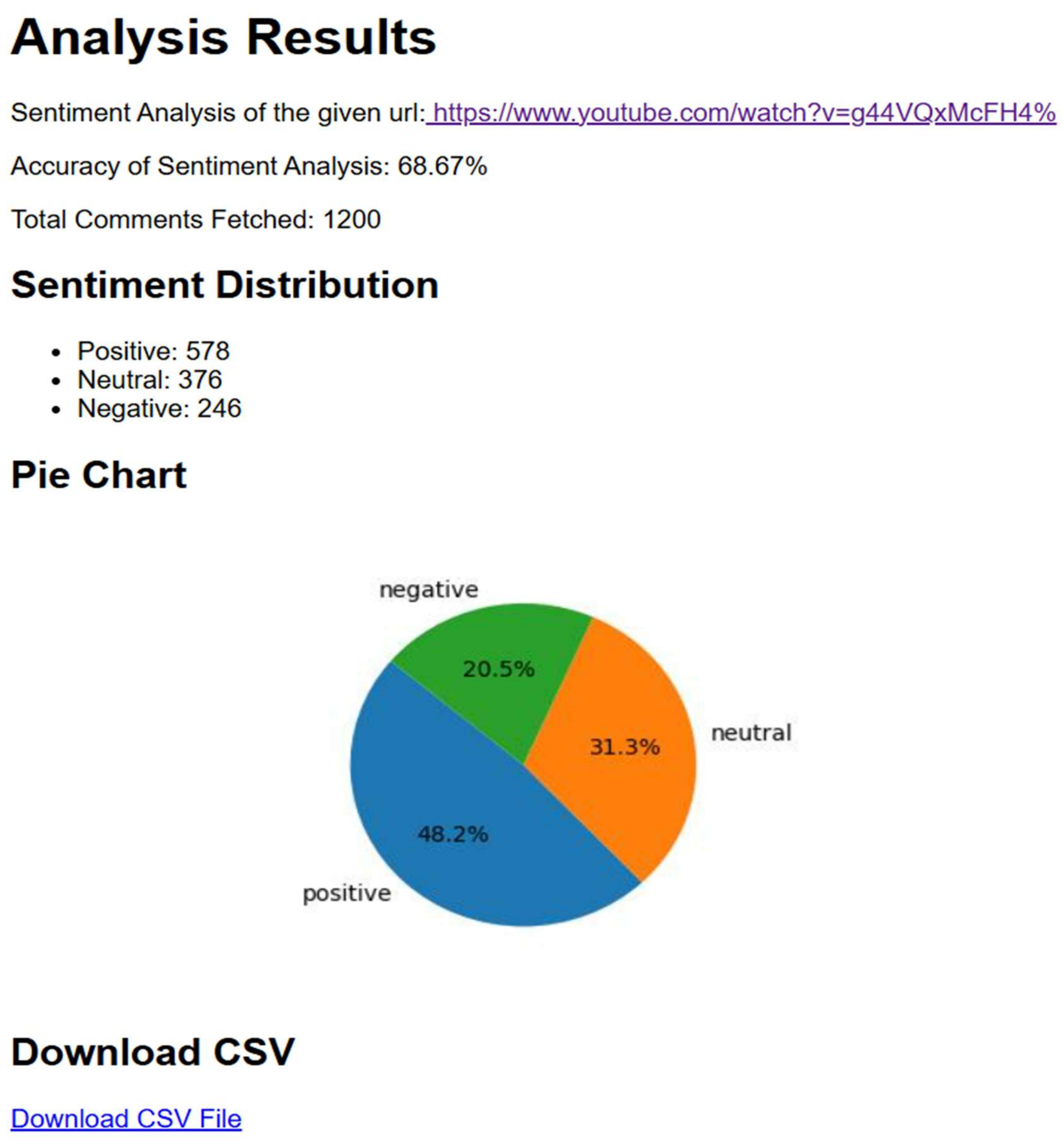
Total Comments: 2275

Number of Comments to Fetch: 1200

CSV File Name (e.g., sample.csv): url.csv

Analyze

Fig: Main page



Analysis Results

Sentiment Analysis of the given url: https://www.youtube.com/watch?v=g44VQxMcFH4%

Accuracy of Sentiment Analysis: 68.67%

Total Comments Fetched: 1200

Sentiment Distribution

- Positive: 578
- Neutral: 376
- Negative: 246

Pie Chart

Download CSV

Download CSV File

Fig: Output Page

CONCLUSION

This project successfully analyzes YouTube comments and determines their sentiment using Natural Language Processing (NLP) techniques. By integrating NLTK’s VADER sentiment analyzer, the system efficiently classifies comments into positive, negative, or neutral categories.

This project demonstrates the power of automated sentiment analysis, making it easier to understand audience opinions about YouTube videos. It provides valuable insights for content creators, marketers, and researchers by offering sentiment distribution, accuracy analysis, and data visualization through pie charts.

REFERENCES

1. https://www.researchgate.net/publication/351351202_YOUTUBE_COMMENTS_SENTIMENT_ANALYSIS

2. https://ieeexplore.ieee.org/document/9396049

3. https://www.geeksforgeeks.org/sentiment-analysis-of-youtube-comments/