1. Project Introduction & Goal

The SMA (Social Media Analytics) Project is a robust sentiment analysis system designed to interpret human emotions across digital platforms. In today's digital age, millions of comments are generated every minute. Our platform provides a simple, automated way to categorize these comments into Positive, Negative, or Neutral sentiments.

Key Objectives:

- Provide content creators with insight into audience feedback.
- Help businesses monitor brand reputation in real-time.
- Enable academic or market research through large-scale comment analysis.
- Create a 'Brand Suitability Score' to help influencers understand their profile health.

This documentation serves as a guide for the presentation, explaining how the backend, frontend, and Machine Learning components work together to deliver actionable insights.

2. System Architecture - Where is Everything?

The project is structured logically into components that handle different tasks. Here is the breakdown of what each file does and where it is located:

Core Backend Files:

- app.py: The Flask server. It handles web requests (API endpoints) and connects everything.
- sentiment_engine.py: The logic engine that performs the actual sentiment calculation.
- comment_fetcher.py: The scraper module. It uses libraries like Instaloader and BeautifulSoup to get comments.
- creator_analytics.py: Specialized logic for analyzing a creator's entire profile across multiple links.

Machine Learning & Processing:

- model_pickle: This is the actual PRE-TRAINED ML MODEL. It is stored as a binary file.
- text_processor.py: A heavy-duty cleaning script that prepares messy internet text for the ML model.

Frontend (The UI):

- templates/: Contains HTML files like 'index.html' (Home) and 'dashboard.html' (Results).
- static/: Stores the CSS for styling and Javascript (app.js, dashboard.js) for interactive charts.

3. How it Fetches Comments (Data Acquisition)

One of the project's most powerful features is the automated retrieval of comments directly from a URL, streamlining the data collection process across different social media platforms.

Step-by-Step Fetching Process:

The 'comment_fetcher.py' script uses a variety of techniques to gather data. Here is how it handles different platforms:

- YouTube: Extracts the Video ID from the link and uses a downloader to get the latest 200+ comments.
- Reddit: Appends '.json' to the Reddit post URL. This allows us to read the public data directly without needing a developer account.
- Instagram: Uses the 'instaloader' library to identify post shortcodes. If anonymous access is blocked, it notifies the user to paste comments manually, ensuring the tool never truly breaks.

Why This Matters:

By automating the fetch process, users save hours of manual work. The system handles the formatting, leaving only raw text for the next step: Sentiment Analysis.

4. The Intelligence: Sentiment Engine & VADER

This is the 'Heart' of the project. We use a Machine Learning algorithm called VADER (Valence Aware Dictionary and sEntiment Reasoner).

Where is the Algorithm used?

The algorithm is used inside 'sentiment_engine.py'. When the app starts, it loads the 'model_pickle' file. This file contains the pre-programmed 'vocabulary' and 'rules' that VADER uses.

How the ML Model Works:

- Loading: The SentimentAnalyzer class in 'sentiment_engine.py' opens 'model_pickle' using Python's pickle library.
- Scoring: For every comment, it calculates a 'Compound Score' between -1 (Very Negative) and +1 (Very Positive).
- Classification: If the score > 0.05, it is Positive. If < -0.05, it is Negative. Otherwise, it is Neutral.

The Cleaning Pipeline (text_processor.py):

Before the ML model sees a comment, we must 'clean' it. Our 'text_processor.py' script performs over 100 cleaning steps, including:

- Removing Emojis and Special Characters.
- Correcting slang and typos (e.g., 'don't' -> 'do not', 'lmao' -> 'laughing my ass off').
- Removing HTML tags and broken characters to ensure the Model gets high-quality text.

5. Features & The Analysis Dashboard

Once the analysis is complete, the results are sent to a beautiful, interactive dashboard built with Chart.js.

Main Features:

- Sentiment Breakdown: Clear counts of how many people are happy vs unhappy.
- Visual Charts: Pie charts and Bar charts for quick visual understanding.
- Brand Suitability Score: A unique metric that calculates if a creator is 'safe' for brand advertisements.
- Comment Filtering: Users can click on 'Positive' to read ONLY the positive comments, helping creators find their best fans quickly.

Creator Analytics:

In the 'Creator Mode', the platform can analyze multiple videos or posts at once to give a 'Creator Health Report'. This aggregates data across all links to provide a high-level summary of the audience sentiment.

6. Presentation Summary & Talking Points

If you are presenting this project tomorrow, here are the key 'Simple' things you should say to the audience:

Closing Talking Points:

- 'This project bridges the gap between raw data and audience understanding.'
- 'We don't just count comments; we understand their meaning using Machine Learning.'
- 'Our system is 'Universal' because it can fetch data from any site using its smarter fetcher module.'
- 'The core intelligence (VADER) allows us to detect subtle emotions even in informal social media language.'

Thank you! Prepared for the SMA Project Presentation.