Phase	Task	Date	Decisions & Rationale	Challenges & Solutions	Implementation
1:Data Collection Research and Assess	**Collect comments from YouTube using API**	**21 September 2025**	Decision: Use YouTube Data API to automatically collect comments. Rationale: Efficient, reliable, and supports keyword filtering.	Challenges: API key required; some videos had comments disabled. Solutions: Used valid API key and try/except blocks to skip problematic videos	**Implementation: Python with googleapiclient; fetched video IDs, retrieved comments for each video with engagement metrics, and saved to CSV using pandas.**
	Collect tweets from Twitter using API	**21 September 2025**	Decision: Use Twitter API (Tweepy) to collect recent tweets with relevant keywords. Rationale: Provides structured access to tweets with engagement metrics and filtering options.	Challenges: Need to collect relevant tweets and handle API limits. Solutions: Applied keyword filtering, limited results per request, excluded retweets, and collected engagement metrics.	Implementation: Python with tweepy.Client, search_recent_tweets(), extracted tweet text and metrics, stored in pandas.DataFrame and saved with to_csv().
	Collect comments from Reddit using API	**22 September 2025**	Used praw to collect comments automatically as it provides structured access and engagement data.	API limits and search errors → set post limits, applied time filter, and used try/except to skip failed requests.	Used praw.Reddit + subreddit.search() to fetch comments, stored results in a pandas.DataFrame, and exported to CSV.
	Clean the collected comments	**22 September 2025**	Decision: Apply text cleaning to improve data quality. Rationale: Cleaning ensures reliable and accurate analysis by removing noise and irrelevant information	Challenge: The dataset contained duplicate comments, empty rows, URLs, mentions, emojis, punctuation, and stopwords that could affect analysis. Solution: Applied a series of preprocessing steps to remove duplicates, empty rows, URLs, mentions, emojis, punctuation, stopwords, and reset the DataFrame index.	Used the following Python functions/methods: drop_duplicates() dropna() reset_index() re.sub() for URLs and mentions re.compile().sub() for emojis str.translate() for punctuation stopwords.words() and list comprehension for stopwords removal
	Measure and visualize the proportion of hate speech per platform.	**24 September 2025**	Decided to calculate hate speech proportions using normalized labels (1 = hate, 0 = non-hate) and visualize results with a bar chart. This provides a clear comparison across platforms.	Some datasets had different label formats (text vs numbers). Solution: Standardized labels by mapping text (e.g., "toxic", "hate") to numeric values and dropping invalid rows.	Normalized label column to numeric (0/1). Filtered valid rows only. Calculated hate speech proportion for each platform. Visualized results using matplotlib bar plot with percentages on top.

	Visualize engagement distribution for hate vs non-hate content across platforms.	**24 September 2025**	Use boxplots to compare engagement levels (likes, replies, etc.) between hate and non-hate posts. Boxplots clearly show medians, ranges, and spread, making comparisons easier.	Labels and engagement columns had inconsistent formats → Solution: Standardize labels using to_binary_label() and convert engagement to numeric. Some rows missing data → Solution: Filter out invalid rows before plotting.	Convert label column to numeric (0/1) using to_binary_label. Filter valid rows with both label and engagement. Separate data by platform and label. Plot boxplots with matplotlib, showing engagement distribution for each label per platform.
2: Data Collection, Processing, Cleaning, and Exploratory Data Analysis (EDA)	**Feature Scaling and Filtering.**	**14 October 2025**	**Scale numeric features (Engagement, toxicity, word count) to [0,1] and remove extreme word counts; needed for consistency and model readiness.**	Outliers in word count → filtered rows with word count > 0 and < 300; scaled using MinMaxScaler.	MinMaxScaler() from sklearn.preprocessing, filtering with boolean indexing.
	Exploratory Data Analysis (EDA) For Reddit .	**14 October 2025**	**Visualize distributions, label proportions, correlations, and engagement trends; needed to understand patterns and data quality.**	Outliers and varying distributions → used histograms, boxplots, heatmaps, and countplots to detect and summarize.	matplotlib, seaborn for histograms, boxplots, heatmaps, and countplots.
	Enhance Missing Value Handling	22 September 2025	**Decision:** Instead of deleting all rows with missing values, numeric columns were filled with zeros or mean values. **Rationale:** To minimize data loss and keep the dataset balanced.	**Challenge:** Many numeric columns had missing values which, if dropped, would reduce the dataset drastically. **Solution:** Applied mean imputation for continuous variables and zeros for counts.	**Used fillna(0) and me.["fillna(df["toxicity_score an()) in pandas.**
	Convert Text to Lowercase	22 September 2025	**Decision:** Standardize all tweet text to lowercase before analysis. **Rationale:** Ensures consistent text comparison (e.g., “Hate” vs “hate”).	**Challenge:** Mixed capitalization caused inconsistent counts for the same word. **Solution:** Converted all text to lowercase using Python string methods.	**Used text.lower() during the text cleaning process.**
	Label Encoding	22 September 2025	**Decision:** Convert categorical labels (“Hate”, “Non-Hate”) to numerical form (1, 0).	**Challenge:** Label column contained inconsistent values (different cases and formats).	**Used :"replace({"Hate.["df["Label 0}) in pandas.:"Non-Hate" ,1**

			Rationale: Required for statistical analysis and model training.	**Solution:** Mapped all text labels to numerical equivalents using a replace dictionary.	
	Normalize Numeric Columns	22 September 2025	**Decision:** Scale numerical features between 0 and 1 for uniform comparison. **Rationale:** Prevent bias caused by differences in feature scales.	**Challenge:** Variables like engagement and word count had wide value ranges. **Solution:** Applied normalization to compress scales evenly.	**Used MinMaxScaler() from sklearn.preprocessing on numeric columns.**
	Filter Text by Word Count	22 September 2025	**Decision:** Remove rows with very short or extremely long tweets (0–300 words kept). **Rationale:** To ensure text quality and remove noise.	**Challenge:** Some rows had empty or excessively long texts affecting analysis. **Solution:** Applied condition to keep text with reasonable word count range.	**Used df = df[(df["word_count"] > 0) & (df["word_count"] < 300)].**
	Perform Exploratory Data Analysis (EDA) on Twitter Dataset	23 September 2025	**Decision:** Conduct EDA to understand data characteristics, detect outliers, and identify trends and relationships between variables. **Rationale:** EDA provides insights into data structure, quality, and relationships that inform the modeling phase and guide decision-making.	**Challenges:** - Needed to separate univariate and bivariate analysis as outlined in lecture guidelines. - Some variables showed very small variation after normalization. **Solutions:** Applied both graphical and non-graphical EDA methods to interpret patterns; adjusted histogram bin sizes and visual scales to enhance interpretability.	**,pandas ,Python :Tools used - seaborn. ,matplotlib :(Univariate (Non-Graphical - ,()describe. ,()info. Used value_counts() to summarize . data. :(Univariate (Graphical - Created histograms and boxplots for numeric columns using sns.histplot() and sns.boxplot(). Created :(Bivariate (Graphical - correlation heatmap with sns.heatmap() and analyzed .relationships such as Label vs word_count. Findings: - • Hate (Label=1) tweets tend to have slightly higher word counts. • Engagement and toxicity are weakly correlated. • Some outliers observed in engagement values. Confirmed :Metadata Review - ,(source as Twitter API (Tweepy ,collected on 21 Sept 2025 focused on hate-speech-related keywords. Recognized :Bias Awareness - overrepresentation of negative ,limited to English users ,tweets and higher visibility bias due to engagement filtering.**
	Perform Exploratory Data Analysis (EDA) on YouTube Dataset	23 September 2025	**Decision:** Conducted Exploratory Data Analysis (EDA) to understand the structure, quality, and relationships within the YouTube comments dataset. The analysis focused on numerical variables such as	**Challenges:** • Presence of skewed distributions in numerical columns (especially toxicity_score and Engagement).	Seaborn: • Used **describe()**, **info()**, and **value_counts()** for statistical summaries. • Created **histograms** to visualize distributions of numeric features (**Engagement**, **toxicity_score**,

			Engagement, toxicity_score, word_count, and negative_word_count, as well as categorical label distribution (Hate = 1, Non-Hate = 0). Detected outliers and patterns using both graphical and non-graphical methods. **Rationale:** EDA helps reveal hidden trends in user engagement, toxicity, and word usage. It assists in understanding how hate-related comments differ from normal comments, and ensures the dataset is suitable for modeling and classification**.**	• High imbalance between hate and non-hate labels. • Some variables contained extreme or near-zero values after scaling. Solutions: Applied both univariate and bivariate graphical EDA methods using matplotlib and seaborn libraries.	**word_count**, **negative_word_count**). • Generated **boxplots** for numeric columns to detect outliers and compare label-based distributions. • Built **correlation heatmap** (**sns.heatmap()**) to identify relationships among numeric variables. • Plotted **countplot** (**sns.countplot()**) to display label distribution between hate and non-hate comments. Findings: • The dataset is label-imbalanced, with a larger number of non-hate comments. • Hate comments (**Label = 1**) tend to have slightly higher word counts and engagement values. • Engagement **and** toxicity_score **show a weak positive correlation (~0.34).** • Most **toxicity_score** and **Engagement** values are concentrated near zero, indicating skewed data. • Some outliers appear in **word_count** and **Engagement**, but they represent valid extreme user interactions.Tools Used:**Python, Pandas, Matplotlib, Seaborn.**