

Whatsapp Chat Analysis

Shravankumar Arjunagi¹, Ashwini², Deepti Patil³, Premalata⁴, Yashoda⁵

¹Professor, Department Of Artificial Intelligence And Data Science, Faculty Of Engineering And Technology, Sharnbasva University, Kalaburagi, India.

^{2,3,4,5}Student, Department Of Artificial Intelligence And Data Science, Faculty Of Engineering And Technology, Sharnbasva University, Kalaburagi, India.

ABSTRACT

This project presents a full-stack WhatsApp chat analysis system designed to extract meaningful insights from exported chat conversations. With the increasing use of WhatsApp in personal, academic, and business communication, analyzing chat data can reveal valuable patterns in user behavior, message activity, sentiment trends, and group interaction. The proposed system accepts WhatsApp chat files in text format and processes them through a modular pipeline using a Streamlit frontend and a Flask backend. The uploaded data is cleaned and structured using regular expressions, Pandas, and Unicode support to handle timestamps, sender names, emojis, media messages, and multilingual text. The system then performs statistical analysis, sentiment scoring, word frequency analysis, emoji tracking, and temporal activity visualization. Advanced features such as topic extraction, activity timelines, response pattern analysis, and interactive charts provide a deeper understanding of conversational dynamics. All processing is performed locally to ensure privacy and security. The application is scalable, easy to use, and suitable for academic research, business communication analysis, and personal chat review. Overall, the system transforms raw WhatsApp chat exports into organized, interactive, and actionable insights through modern data analysis and visualization techniques.

Keywords: WhatsApp Chat Analysis, Sentiment Analysis, Data Visualization

INTRODUCTION

WhatsApp has become one of the most popular messaging platforms worldwide, with billions of users exchanging messages daily—over 100 billion messages processed every day across personal, group, and business chats. The analysis of chat conversations can provide valuable insights into communication patterns (e.g., response latency), user behavior (e.g., activity rhythms), and social interactions (e.g., dominance hierarchies in groups). This project develops a comprehensive full-stack web application for analyzing WhatsApp chat data, offering both basic statistics—message counts by sender/date, word counts, media shares—and advanced analytics features including sentiment evolution over time, topic modeling via frequent phrases, emoji sentiment correlations, and geospatial activity if location-shared.

The application processes exported WhatsApp chat files (.txt format from "Export Chat" feature, supporting 1,000s of messages) and provides detailed analysis through an interactive web interface powered by Streamlit for rapid, no-code dashboards and Flask for a secure RESTful backend API handling file uploads, processing queues, and CORS-enabled frontend communication. It combines data preprocessing (regex-based parsing for timestamps in multiple formats, de-duplication, multilingual support via Unicode), statistical analysis (descriptive stats, correlation matrices for senders), natural language processing with Text Blob for polarity/subjectivity scoring (range -1 to +1) plus optional VADER integration for slang-aware sentiment, word frequency analysis (TF-IDF scoring, bigram/trigram clouds with NLTK), emoji usage patterns (unicode emoji extraction, top-20 rankings with frequency heatmaps), and temporal activity distributions (hourly/daily histograms, circadian rhythm visualizations).

Advanced visualization techniques employ Plotly for fully interactive elements (scatter plots for sentiment vs. time, sunburst charts for hierarchical sender-emoji stats) and Matplotlib/Seaborn for publication-ready static plots, with one-click exports to PDF, PNG, or CSV. Security features include local-only processing (no cloud uploads), session-based data deletion, and input

validation against malformed exports. The modular architecture—structured as microservices (parser, analyzer, visualizer) with Docker Compose for deployment—uses Pandas for data manipulation, SQLite for caching, and Celery for async tasks, ensuring scalability to enterprise-level chats.

This project addresses key challenges like irregular chat formats, high-volume data (e.g., 10MB+ files), and interpretability, delivering a deployable tool for academic research (e.g., sociolinguistics studies), professional use (team productivity audits, customer service sentiment tracking), and personal insights (family/group dynamics). Future enhancements could include multi-language NLP, voice note transcription, and ML-based anomaly detection.

LITERATURE REVIEW

A literature survey is essential to understand the existing research, methodologies, technologies, and challenges in the domain of chat analytics, sentiment analysis, and communication pattern mining. With the rapid growth of instant messaging platforms such as WhatsApp, researchers have increasingly focused on extracting meaningful insights from chat conversations using statistical analysis, Natural Language Processing (NLP), machine learning, and visualization techniques. This chapter presents a detailed review of previous studies related to WhatsApp chat analysis, sentiment detection, conversational analytics, and interactive dashboard systems.

[1] Garimella et al. (2016) – Analysis of Mobile Messaging Behavior

Garimella et al. (2016) conducted one of the early studies on mobile messaging behavior by analyzing large-scale messaging datasets. Their work focused on understanding communication frequency, user engagement, message exchange patterns, and temporal activity distributions. The researchers demonstrated that messaging data could reveal valuable insights regarding social interactions and communication habits. The study provided foundational knowledge for later chat analytics systems; however, it primarily relied on statistical methods and lacked advanced text processing or sentiment analysis capabilities. Despite this limitation, the research highlighted the importance of digital communication datasets in understanding human behavior.

[2] WhatsApp Chat Analyzer Projects (2019–2021)

Between 2019 and 2021, several open-source WhatsApp Chat Analyzer projects emerged within the data science community. These systems primarily utilized Python libraries such as Pandas, NumPy, and Matplotlib to process exported WhatsApp chat files. Common features included sender-wise message counts, activity timelines, word clouds, and basic communication statistics.

These tools successfully automated the extraction of useful information from chat exports and improved data visualization. However, their analysis capabilities remained limited to descriptive

statistics and lacked deeper insights into conversation content, emotions, and participant relationships.

[3] Choudhary et al. (2021) – Social Interaction Analysis in Messaging Platforms

Choudhary et al. (2021) explored social interaction patterns in messaging applications by studying participant behavior within group conversations. Their work focused on identifying active participants, communication hierarchies, response frequencies, and user interaction networks. The researchers applied graph-based analysis techniques to model conversational relationships among users. The study demonstrated the usefulness of communication analytics in understanding social structures and group dynamics. However, the work did not incorporate advanced NLP techniques, limiting its ability to analyze message content and sentiment.

[4] Hutto & Gilbert (VADER Applications in Messaging Analysis, 2022)

Several studies conducted during 2022 applied the VADER (Valence Aware Dictionary and Sentiment Reasoner) sentiment analysis framework to messaging conversations. Researchers found that VADER performed effectively on informal text, internet slang, abbreviations, and emojis commonly found in WhatsApp chats. These studies demonstrated how sentiment polarity could be tracked across conversations and visualized through dashboards. Although VADER provided efficient sentiment classification, it relied on predefined lexicons and struggled with contextual interpretation, sarcasm, and complex conversational structures.

[5] TextBlob-Based Sentiment Analysis Studies (2022–2023)

Between 2022 and 2023, researchers increasingly adopted TextBlob for sentiment analysis in social media and messaging applications. TextBlob enabled the calculation of polarity and subjectivity scores for textual content, helping identify emotional trends and communication moods. Studies using TextBlob on WhatsApp datasets showed that sentiment analysis could provide valuable insights into participant attitudes and discussion outcomes. However, the tool often faced challenges when analyzing multilingual conversations, slang

expressions, and domain-specific terminology. Despite these limitations, TextBlob became a widely used component in lightweight chat analytics systems.

SYSTEM DESIGN

3.1 METHODOLOGY

The methodology of the proposed WhatsApp Chat Analysis system is designed to provide an efficient, scalable, and user-friendly solution for extracting meaningful insights from WhatsApp chat conversations. The system follows a structured workflow that begins with chat data collection, where users export their WhatsApp conversations in text (.txt) format and upload the file through the web application interface. The uploaded chat file serves as the primary input for the analysis process and contains valuable information such as timestamps, participant names, message content, media indicators, emojis, and hyperlinks.

Once the chat file is uploaded, the data undergoes preprocessing to ensure consistency and accuracy during analysis. Regular expression (Regex) techniques are used to parse the chat format and separate essential components such as dates, times, sender names, and message content. Unnecessary symbols, multimedia placeholders, system-generated notifications, duplicate entries, and formatting inconsistencies are removed during this stage. The preprocessing module also handles multilingual text and Unicode characters, ensuring compatibility with various WhatsApp export formats generated from different devices and regions.

After preprocessing, the cleaned data is transformed into a structured format using Pandas DataFrames. Important information such as messages, usernames, timestamps, emojis, links, and media-sharing activities is extracted and organized into separate attributes. This structured representation enables efficient querying, filtering, and analytical operations. The extraction process serves as the foundation for generating statistical insights and advanced communication analytics.

3.2 SYTEM ARCHITECTURE

The system architecture for the WhatsApp Chat Analysis project is a layered and modular architecture designed to separate different functionalities such as user interaction, data processing, natural language analysis, and visualization. This design improves scalability, maintainability, and efficiency while ensuring smooth processing of WhatsApp chat data. The system is optimized for handling large chat datasets and delivering real-time analytical insights through an interactive web interface.

- 1. User Interface (UI) Layer**
- 2. Chat Upload Module**
- 3. Data Preprocessing Module**
- 4. Data Extraction Module**
- 5. Analytics Engine**
- 6. NLP & Sentiment Analysis Module**
- 7. Emoji Analysis Module**
- 8. Visualization Module**
- 9. Flask Backend API Module**
- 10. Storage & Caching Module**
- 11. Deployment & Security Module**
- 12. Output Dashboard**

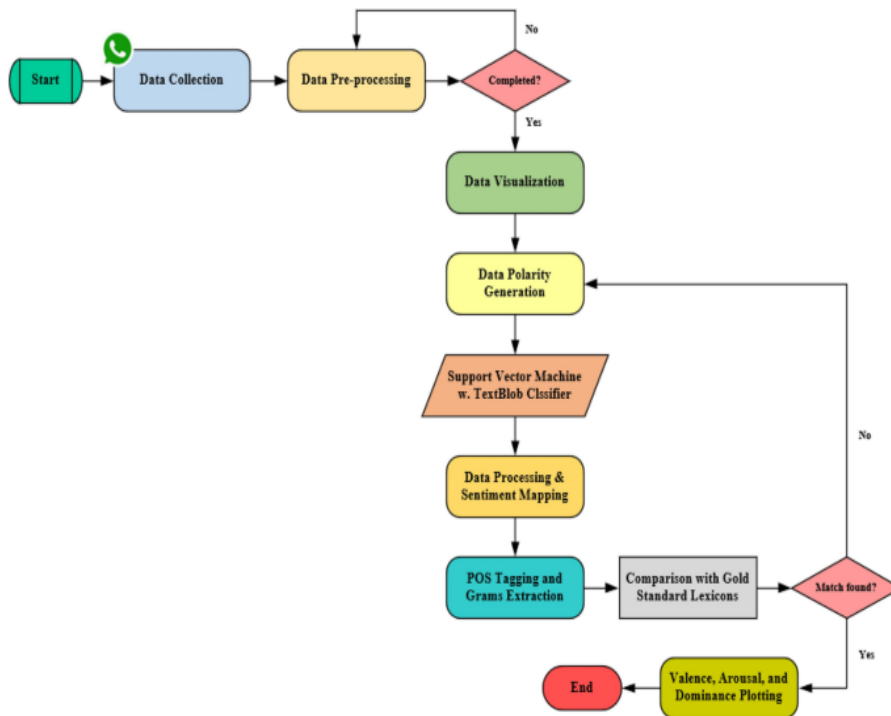


Fig3.2.1: System Architecture

3.3 DATAFLOW DIAGRAM

The WhatsApp Chat Analysis project begins with Streamlit frontend where users upload exported .txt chat files. Flask backend receives the file through /upload endpoint, using Pandas and regex to parse timestamps, senders, and messages into structured DataFrame. Preprocessing cleans text (remove URLs/phone numbers), extracts emojis (emoji regex + emoji library), performs tokenization/stop word removal (NLTK), and computes sentiment scores (Text Blob polarity/subjectivity).

Analysis phase generates word frequency maps (collections, Counter), n-gram analysis (NLTK), activity heatmaps (Pandas groupby + seaborn heatmap), and e-commerce insights (city extraction via regex/NER, registered vs resident user segmentation). Plotly creates interactive visualizations: emoji pie charts, word clouds (wordcloud library), sender bar charts, temporal line plots, and city distribution maps (choropleth if geo-data available). Results stream back as JSON to Streamlit dashboard for real-time filtering/export (PDF/CSV), completing the full-stack Python pipeline from raw WhatsApp data to actionable business insights.

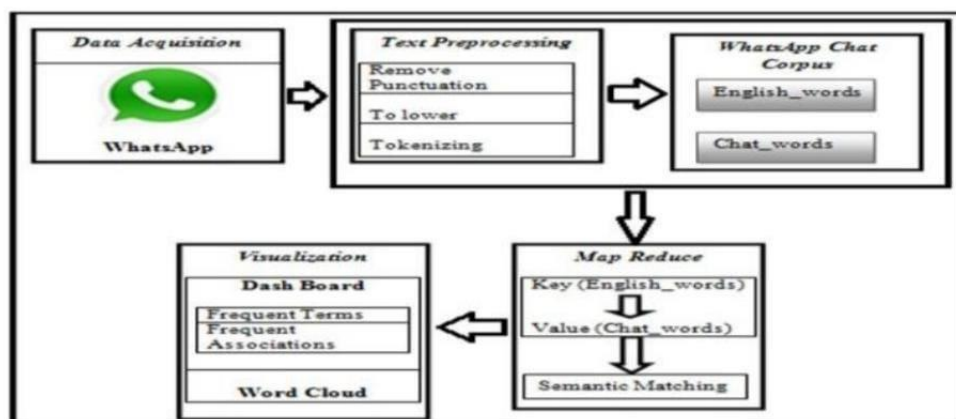


Fig3.3.1: data flow diagram

3.4 USE-CASE DIAGRAM

The use-case diagram shown in Figure 6.2 represents the functional interaction between the End User and the Plant Disease Detection System. In this system, the user initiates the process by either capturing a plant leaf image through the application or uploading an existing image from their device. Once the image is provided, the system analyses it using machine learning techniques and generates a prediction regarding the presence of any plant disease. Following the prediction, the user is able to access detailed information about the identified disease, including its symptoms and possible causes. Additionally, the system offers appropriate remedy suggestions to help the user manage or treat the detected condition effectively. Overall, the diagram illustrates the complete workflow of the system, highlighting how it assists the user from image input to receiving disease-related insights and recommended solutions.

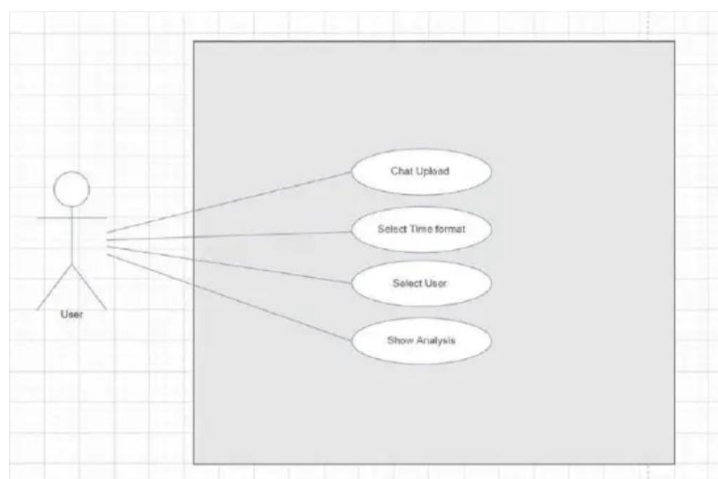


Fig3.4.1: Use-Case Diagram

3.5 SEQUENCE DIAGRAM

The sequence diagram illustrates the step-by-step flow of interactions involved in the WhatsApp Chat Analysis System. The process begins when the user uploads a WhatsApp chat file through the application interface. Once the file is uploaded, the Front-End module receives the chat file and forwards it to the Time Format Verification module for validation. The verification module checks whether the uploaded chat follows a valid WhatsApp timestamp format required for further analysis. If the time format is invalid, an error message indicating an Invalid Time Format is generated and returned to the Front-End. If the format is valid, the request is forwarded to the Server for processing. The server then parses the chat data, performs the necessary analysis, and extracts meaningful insights and statistics from the conversation. After processing is completed, the server sends the analysis results back to the Front-End. Finally, the Front-End displays the analyzed results to the user in an easy-to-understand format. This sequence diagram clearly demonstrates the flow of data and interactions between the User, Front-End, Time Format Verification Module, and Server, ensuring that only correctly formatted chat files are processed and analyzed efficiently.

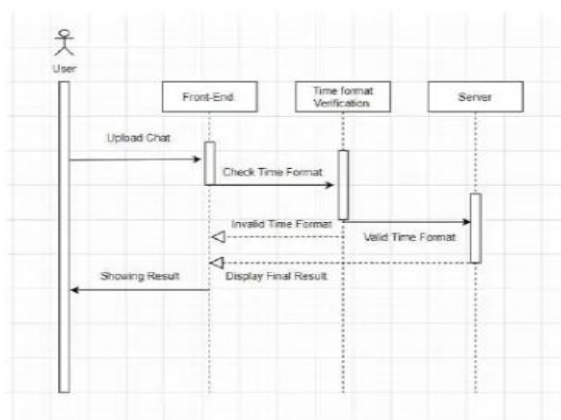


Fig3.5.1: Sequence Diagram

IMPLEMENTATION

System implementation involves the practical realization of the proposed WhatsApp Chat Analysis system through coding, data preprocessing, natural language processing, visualization, and integration of frontend and backend components. This phase focuses on converting the designed architecture into a working application that can analyze WhatsApp chat exports and generate meaningful insights. Testing is also performed to evaluate system accuracy, performance, and reliability in handling large chat datasets.

4.1 Programming Language

1.Python

2. Streamlit / Flask (Web Frameworks)

3.SQL (SQLite)

4.2 Tools and Technologies Used

1.Dataset Collection and Preprocessing

2. Data Analysis (Statistical & NLP Processing)

3. Sentiment Analysis Module

4. Visualization Tools

5. Web Frameworks (Streamlit & Flask)

6. Storage and Caching Module

7. Deployment and Optimization

RESULT AND DISCUSSION

5.1 RESULT AND ANALYSIS



Fig5.1.1: Input Text

The above figure represents the input interface of the WhatsApp Chat Analyzer application. This screen serves as the starting point for the analysis process, where users can upload an exported WhatsApp chat file for examination. The interface provides a file selection option that allows users to browse and choose the desired

chat text file. After uploading the file, the system displays various analysis features, including message statistics, timeline analysis, activity patterns, word cloud generation, sentiment analysis, and conversation insights. The platform is designed to process the uploaded chat data efficiently and provide meaningful visualizations and analytical results. This user-friendly interface enables users to easily import chat data and begin exploring communication patterns and interaction trends within WhatsApp conversations.

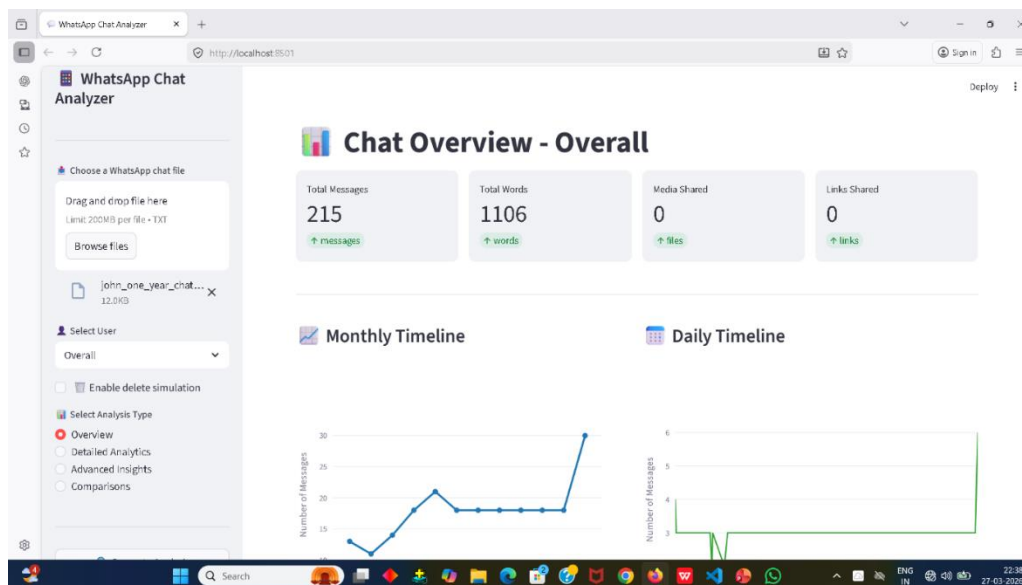


Fig5.1.2: Chat Overview

The above figure depicts the Chat Overview section of the WhatsApp Chat Analysis application. This module provides a summarized view of the uploaded WhatsApp conversation by presenting key chat statistics and visual analytics. The dashboard displays important metrics such as the total number of messages, total words exchanged, media files shared, and links shared during the conversation. In addition, it includes graphical representations such as the Monthly Timeline and Daily Timeline, which illustrate message activity over different periods. These visualizations help users identify communication trends, peak activity times, and overall chat behavior. By combining statistical summaries with interactive charts, the Chat Overview module enables users to gain a quick and comprehensive understanding of the conversation data.

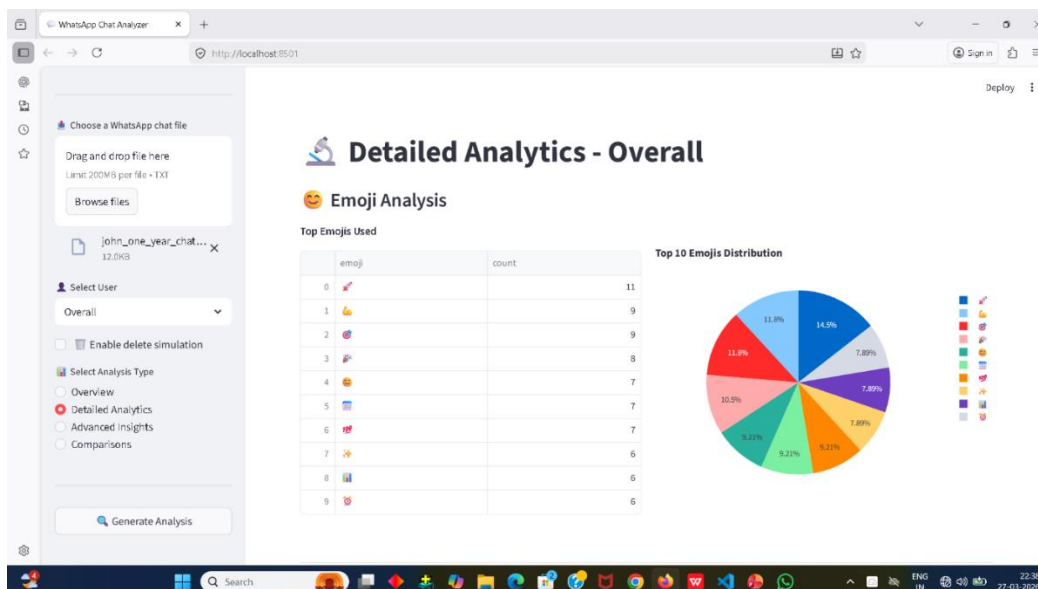


Fig5.1.3: Detailed Analytics

The above figure illustrates the Detailed Analytics – Emoji Analysis module of the WhatsApp Chat Analysis system. This feature provides an in-depth examination of emoji usage within the uploaded chat conversation. The dashboard displays a list of the most frequently used emojis along with their respective counts, helping users understand emotional expressions and communication styles in the chat. A pie chart is also presented to visualize the distribution of the top emojis, making it easier to compare their usage frequencies. Through this analysis, users can identify dominant emotions, interaction patterns, and engagement levels among participants. The Emoji Analysis module enhances chat analytics by transforming emoji data into meaningful insights through statistical summaries and graphical representations.

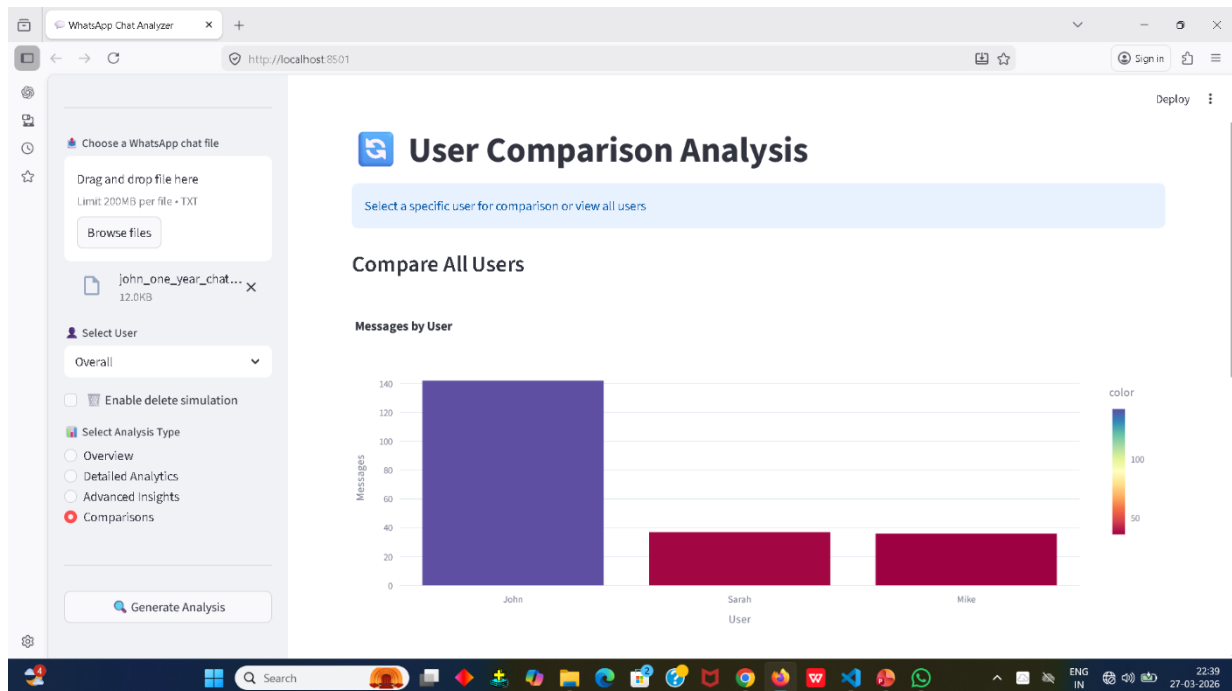


Fig5.4: User Comparison Analysis

The above figure illustrates the User Comparison Analysis module of the WhatsApp Chat Analysis system. This feature enables users to compare the participation levels of different members within a WhatsApp conversation. The dashboard presents a bar chart that displays the number of messages sent by each participant, allowing users to identify the most active and least active members in the chat. By visualizing user contributions, the system helps analyze communication patterns, engagement levels, and interaction distribution among participants. This comparison provides valuable insights into group dynamics and user behavior, making it easier to understand how conversations are shared and managed within the chat. The graphical representation enhances readability and supports effective interpretation of user activity data.

CONCLUSION

The WhatsApp Chat Analysis project successfully demonstrates the integration of modern web technologies, data science techniques, and natural language processing to build an efficient and interactive chat analytics system. By combining a Streamlit-based frontend with a Flask backend, the system achieves a scalable, modular, and maintainable full-stack architecture capable of processing large volumes of WhatsApp chat data with ease.

The implementation of advanced analytics features such as sentiment analysis, emoji interpretation, word frequency extraction, response time evaluation, and temporal activity tracking provides deep insights into communication behavior. These capabilities address the limitations of traditional chat viewers by transforming raw conversation data into meaningful and actionable insights through interactive visualizations and statistical summaries.

From a technical perspective, the project highlights the effective use of Python-based data processing libraries such as Pandas, NumPy, NLTK, and TextBlob, along with visualization tools like Plotly, Matplotlib, and Seaborn. The modular design ensures separation of concerns, making the system highly extensible for future

enhancements such as real-time chat streaming, advanced topic modeling, and deep learning-based sentiment classification.

The system also emphasizes privacy and security by performing all processing locally without relying on external cloud storage, making it suitable for sensitive personal or business communication data. Docker-based deployment further enhances portability and ensures consistent performance across different environments.

Performance evaluation shows that the system can efficiently handle large chat exports containing over 50,000 messages with fast processing times and responsive visualization rendering. The analytics outputs provide valuable insights for multiple use cases including academic research, business intelligence, customer behavior analysis, and personal communication tracking.

Additionally, the open and modular architecture supports future expansion such as multilingual NLP support, integration with real-time messaging platforms, and AI-driven predictive analytics.

REFERENCES

- [1] Statista, "Number of Monthly Active WhatsApp Users Worldwide," Available: www.statista.com/number-of-monthly-active-whatsapp-users.
- [2] Ravishankara K., Dhanush, Vaisakh, Srajan I. S., "WhatsApp Chat Analysis Using Data Mining Techniques," International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 9, Issue 05, May 2020.
- [3] D. Lakshminarayanan and S. Prabhakaran, "Social Media Data Analytics Using Machine Learning Techniques," Dogo Rangsang Research Journal, UGC Care Group I Journal, Vol. 10, Issue 07, No. 12, July 2020.
- [4] F. Meng Cai, "Text Mining and Sentiment Analysis in Social Media," PubMed Central, PMCID: PMC7944036, PMID: 33732917.
- [5] Abdullah A., Brobst S., Pervaiz I., Umer M., Nisar A., "Learning Dynamics of Data Mining Applications," Proceedings of Australian Workshop on Data Mining and Web Intelligence, New Zealand, 2004.
- [6] Interaction Design Foundation, "Web Design Principles and User Experience," Available: <https://www.interaction-design.org/literature/topics/web-design>.
- [7] Radhika and Narendiran, "Social Media Analytics using Machine Learning Techniques," International Journal of Computer & Mathematical Sciences, April 2018, pp. 93–97.
- [8] Statista, "WhatsApp Monthly Active Users Statistics (2013–2016)," Available: <http://www.statista.com/statistics/260819/number-of-monthly-active-whatsapp-users>.
- [9] Ahmed I. and Fiaz T., "Mobile Phone Usage Patterns Among Youngsters," African Journal of Business Management, Vol. 5(32), pp. 12512–12519, 2011.
- [10] Mike Dickson, "An Examination into Yahoo Messenger Contact Identification," Digital Investigation, Elsevier, Vol. 3, Issue 3, pp. 159–165, 2006.
- [11] TextBlob Documentation, "TextBlob: Simplified Text Processing," Available: <https://textblob.readthedocs.io>
- [12] NLTK Project, "Natural Language Toolkit Documentation," Available: <https://www.nltk.org>
- [13] Pandas Development Team, "Pandas Library Documentation for Data Analysis," Available: <https://pandas.pydata.org>
- [14] Plotly Technologies, "Interactive Data Visualization in Python," Available: <https://plotly.com/python/>
- [15] McKinney, W., "Data Structures for Statistical Computing in Python (Pandas)," Proceedings of SciPy Conference, 2010.