WhatsApp Chat Analyzer

A Data-Driven Approach to
Understanding Group Communication

Created by: Shivansh Satpute

Introduction

- Objective: Analyze WhatsApp chats for communication insights.
- Tools: Built using Streamlit, Python, and data visualization libraries.
- Use Case: Track activity trends, media sharing, and user engagement.

Technology Stack

- Python
- Streamlit
- Pandas, Matplotlib, Seaborn
- WordCloud, Emoji, URLEXtract

Features

- Upload WhatsApp exported chat (.txt)
- Analyze messages for selected user or entire chat
- Generate timeline, activity maps, and word/emoji statistics

preprocessor.py - Data Preprocessing

- **Cleans and Structures Raw WhatsApp Chat Data**
  - Parses .txt file exported from WhatsApp.
  - Uses regex to split messages by timestamps and extract dates.
- **Date & Time Conversion**
  - Converts extracted date strings to Python datetime objects.
  - Supports 12-hour format with AM/PM.
- **User & Message Extraction**
  - Separates sender name and message content using pattern matching.
  - Detects system notifications (e.g., group updates, media omitted).
- **Feature Engineering**
  - Adds multiple time-based columns:
    - only_date, year, month_num, month, day, day_name, hour, minute
- **Hourly Period Binning**
  - Creates a new period column (e.g., "14-15") to represent time ranges.
  - Supports plotting heatmaps of hourly activity.
- **Returns a Structured Pandas DataFrame**
  - Final dataframe is ready for statistical and visual analysis.

helper.py - Stats & Visuals

- Calculate message, word, media, and link counts.
- Identify most active users in group chats.
- Generate monthly & daily timelines, and activity heatmaps.
- Create WordCloud and extract most common words.
- Analyze emoji usage across chats.

helper.py - Functions

- **fetch_stats(selected_user, df)**
  - Returns total messages, word count, media count, and link count.
  - Filters data based on selected user (or "Overall").
- **most_busy_users(df)**
  - Identifies top contributors in group chats.
  - Returns both absolute count and percentage contribution.
- **create_wordcloud(selected_user, df)**
  - Removes Hinglish stop words.
  - Generates a wordcloud from the cleaned messages.
- **most_common_words(selected_user, df)**
  - Extracts and ranks the top 20 most used words.
  - Excludes media, group notifications, and stopwords.
- **emoji_helper(selected_user, df)**
  - Analyzes and counts emojis used in messages.
  - Returns a DataFrame of most used emojis.

helper.py - Functions

- **monthly_timeline(selected_user, df)**
  - Groups message counts by month and year.
  - Prepares data for plotting monthly trends.
- **daily_timeline(selected_user, df)**
  - Counts daily message frequency.
  - Useful for observing short-term engagement spikes.
- **week_activity_map(selected_user, df)**
  - Bar chart data: distribution of messages by day of week.
  - **month_activity_map(selected_user, df)**
  - Bar chart data: distribution of messages by month.
- **activity_heatmap(selected_user, df)**
  - Creates a pivot table of activity by day and hour.
  - Used for generating a heatmap of user activity.

app.py - Streamlit Web App

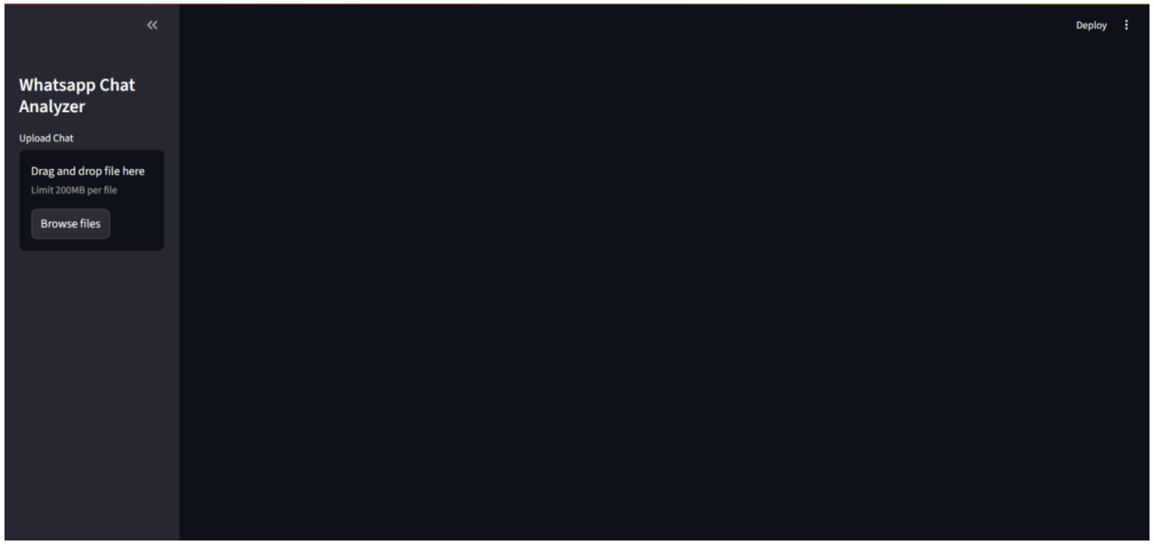
- **File Upload via Streamlit Sidebar**
  - Accepts .txt chat exports from WhatsApp.
  - Initiates preprocessing using preprocessor.py.
- **User Selection**
  - Dropdown to select either a specific user or "Overall".
  - All analytics update dynamically based on the selection.
- **Display of Key Statistics**
  - Total messages, words, media files, and links shared.
- **Time-Based Analysis**
  - **Monthly Timeline**: Message count per month.
  - **Daily Timeline**: Message frequency per day.
- **Activity Maps**
  - **Most Active Day & Month**: Bar charts.
  - **Weekly Heatmap**: Visualizes hourly activity per day.
- **Chat Insights**
  - Identifies most active participants (if “Overall” is selected).
  - Displays both chart and table of user activity.

app.py - Streamlit Web App

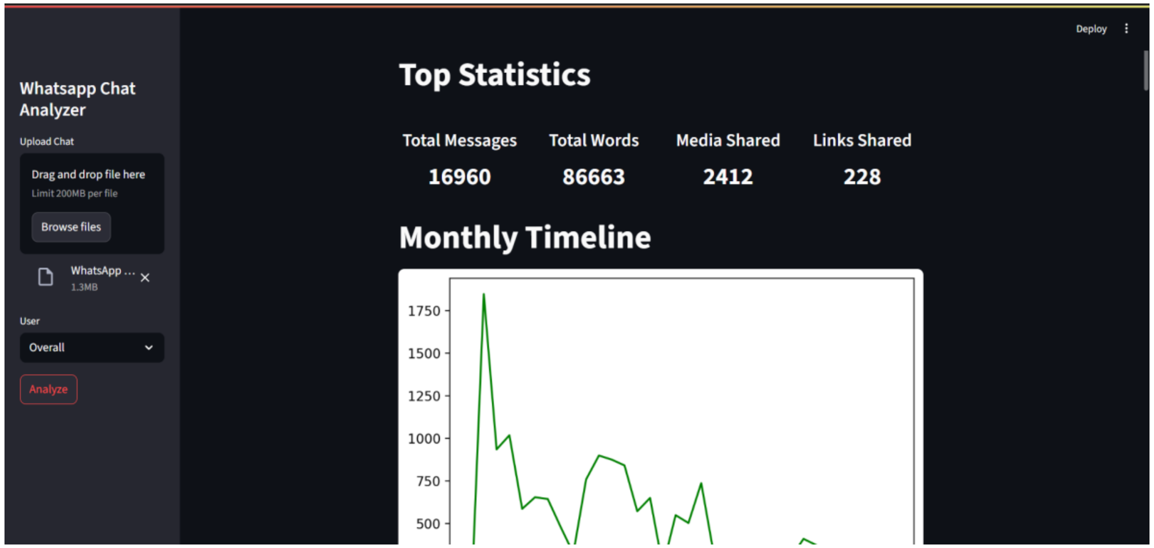
- **WordCloud and Text Analysis**
  - Generates WordCloud excluding Hinglish stop words.
  - Shows most commonly used words.
- **Emoji Analysis**
  - Table of frequently used emojis.
  - Pie chart for visual summary.
- **Uses Matplotlib & Seaborn for Plots**
  - All visualizations are rendered within Streamlit using st.pyplot.

Web App - Detailed Look

Before Uploading Chat:



After Uploading Chat:



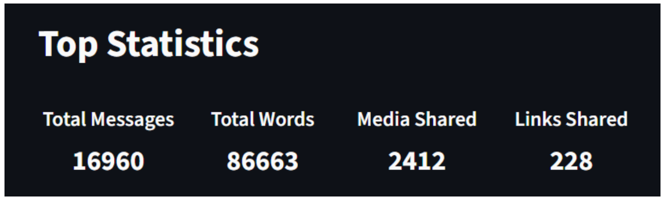
Top Statistics

Total Messages	Total Words	Media Shared	Links Shared
16960	86663	2412	228

Monthly Timeline

Web App - Detailed Look

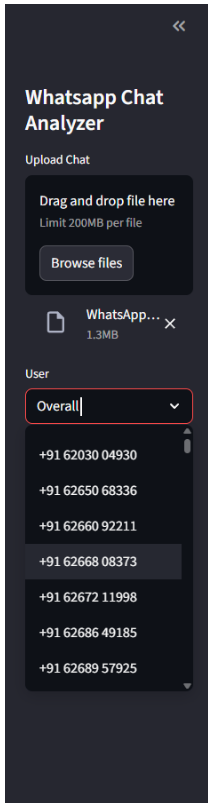
Top Statistics:



Top Statistics

Total Messages	Total Words	Media Shared	Links Shared
16960	86663	2412	228

File Upload & User Selection:



Whatsapp Chat Analyzer

Upload Chat

Drag and drop file here
Limit 200MB per file

Browse files

WhatsApp…
1.3MB

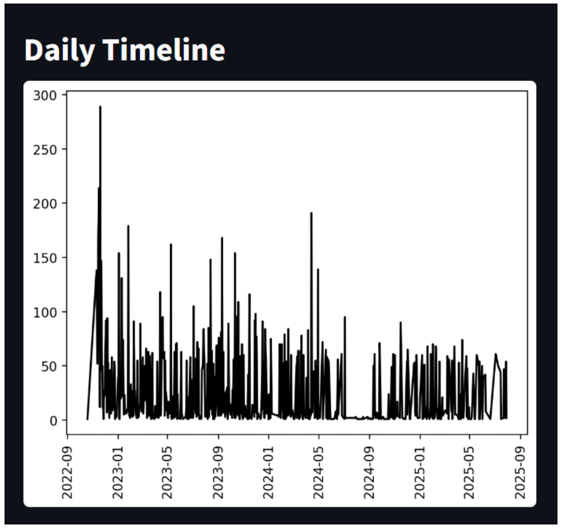
User

Overall

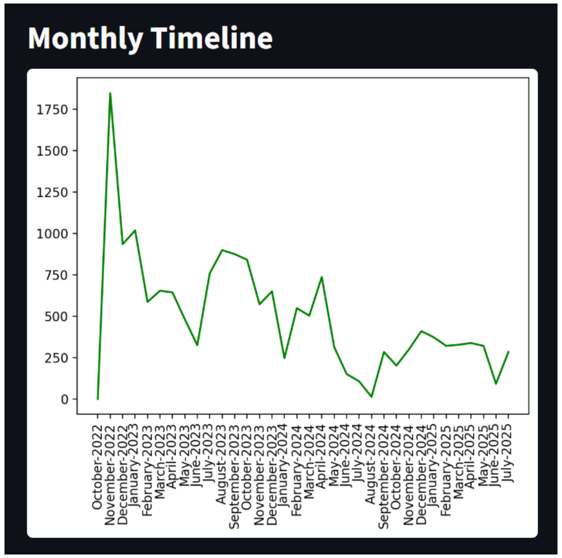
+91 62030 04930
+91 62650 68336
+91 62660 92211
+91 62668 08373
+91 62672 11998
+91 62686 49185
+91 62689 57925

Visualisations:

Daily Timeline

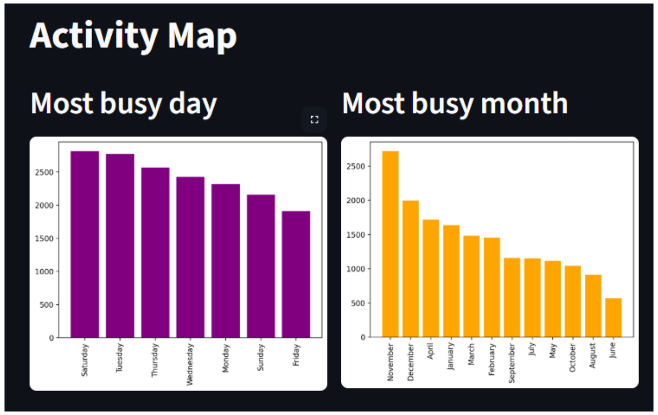


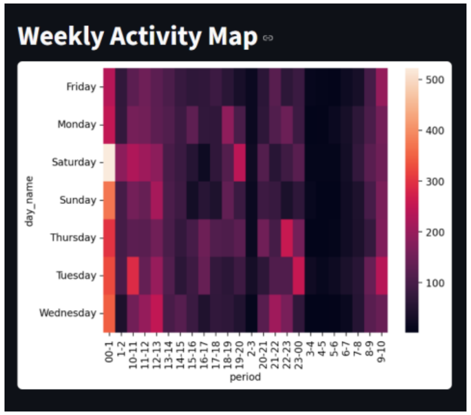
Monthly Timeline

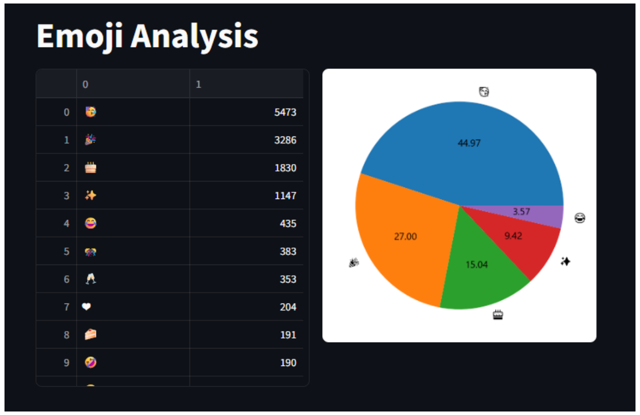


Web App - Detailed Look

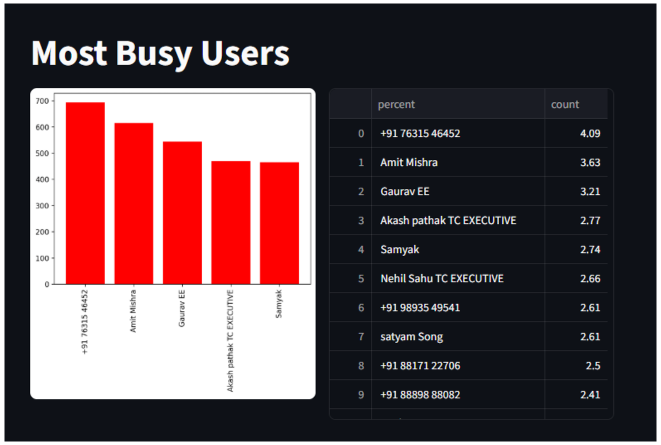
Visualisations:



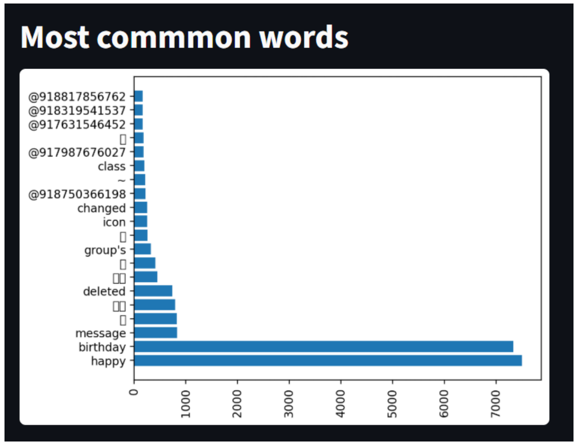


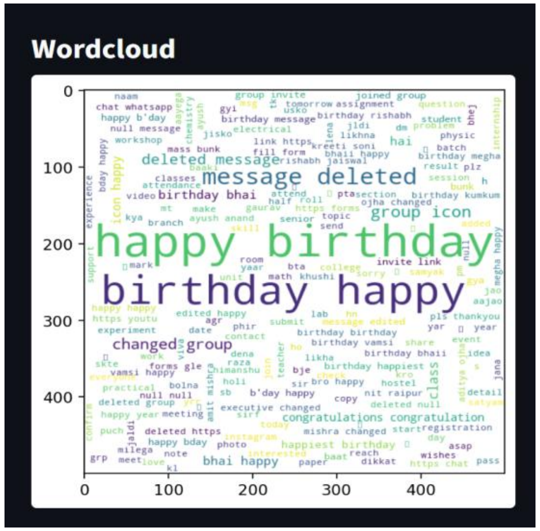


	0	1
0	🥳	5473
1	🎉	3286
2	🎂	1830
3	✨	1147
4	😂	435
5	🎊	383
6	🥂	353
7	🤍	204
8	🍰	191
9	🤣	190



	percent	count
0	+91 76315 46452	4.09
1	Amit Mishra	3.63
2	Gaurav EE	3.21
3	Akash pathak TC EXECUTIVE	2.77
4	Samyak	2.74
5	Nehil Sahu TC EXECUTIVE	2.66
6	+91 98935 49541	2.61
7	satyam Song	2.61
8	+91 88171 22706	2.5
9	+91 88898 88082	2.41





Conclusion

- Most active days and users were identified.
- Media sharing and link sharing patterns were visualized.
- WordCloud and Emoji analysis showed group sentiment and behavior.
- Application is scalable and useful for group communication studies.

Project Links

- 🔗 GitHub Repository
- 🌐 Live Web App