K.R. Mangalam University

School of Engineering and Technology

Project Report On

Whatsapp chat Analyzer

Submitted By:

Vandana — 2401560070

Ishita Bhardwaj — 2401560055

Pallavi Kumari — 2401560084

Group - Y1-2024-25-G292

Under the Guidance of:

Mr. Harsh Vardhan

Assistant Professor

School of Engineering and Technology

Academic Year: 2024–2025

Certificate

This is to certify that the project titled **"Whatsapp chat analyzer"** submitted by **Vandana (Roll No. 2401560070), Ishita Bhardwaj (Roll No. 2401560055), Pallavi Kumari (Roll No.2401560084)**
in partial fulfillment of the requirements for the award of the degree of
Master of Computer Applications from **K.R. Mangalam University** is a bona fide record of work carried out by them during the academic year **2024–2025** under my guidance.

Mr. Harsh Vardhan
Guide

Head of Department
(Signature)

Declaration

We hereby declare that the project titled **"Whatsapp chat analyzer"** submitted to **K.R. Mangalam University, Gurugram**, in partial fulfillment of the requirements for the award of the degree of **Master of Computer Applications** is our original work.

This project has been carried out by us under the guidance of **Mr. Harsh Vardhan**, School of Engineering and Technology.

We further declare that this work has not been submitted previously, in whole or in part, for the award of any degree, diploma, or any other similar title at this or any other university/institution.

All the information provided in this report is true to the best of our knowledge and belief. Proper references and acknowledgments have been given wherever necessary.

Submitted by:

Name	**Roll Number**
Vandana	2401560070
Ishita bhardwaj	2401560055
Pallavi Kumari	2401560084

Acknowledgment

We would like to express our heartfelt gratitude to **Mr. Harsh Vardhan**, Faculty, School of Engineering and Technology, **K.R. Mangalam University**, for her invaluable guidance, encouragement, and constant support throughout the course of this project.
Her insightful suggestions, timely feedback, and immense knowledge greatly contributed to the successful completion of our work.

We are also deeply thankful to **K.R. Mangalam University** for providing the necessary facilities, resources, and an environment conducive to learning and research, which made this project possible.

We sincerely appreciate the cooperation and support of all the faculty members and staff who directly or indirectly helped us during this project.

Finally, we are immensely grateful to our families and friends for their continuous encouragement, understanding, and moral support throughout the duration of this project.

This project would not have been possible without the contributions of all these wonderful individuals.

Abstract

This project titled **"WhatsApp Chat Analyzer Using Python"** presents a data analysis tool developed using Python and Streamlit to gain insights into WhatsApp chat histories. As digital communication becomes increasingly prevalent, analyzing chat data can reveal interesting patterns such as most active users, message trends, emoji usage, and word clouds.

The application is designed to accept WhatsApp exported chat text files and then preprocess and analyze the data. We used pandas for data manipulation, matplotlib and seaborn for visualization, and wordcloud for generating word-based summaries.

Key features include statistics for messages, words, media and links shared, busiest users, activity heatmaps, and emoji distribution.

This project demonstrates the capability of Python in data processing and visualization, providing a user-friendly interface for non-technical users to explore chat behavior.

Table of Contents

- Declaration i
- Acknowledgment ii
- Abstract iii
- Table of Contents iv

Table of Contents

1. **Introduction**
2. **Literature Review**
3. **System Analysis**
4. **System Design**
5. **Implementation**
6. **Testing**
7. **Results and Discussion**
8. **Conclusion**
9. **Future Work**
10. **References**

1. Introduction

1.1 Overview

In today's digitally connected world, instant messaging has become an essential form of communication. WhatsApp, being one of the most popular messaging apps, generates a large amount of personal and group data that can be analyzed for various insights. The purpose of this project is to build a tool that helps users analyze their WhatsApp chats and visualize communication patterns using Python.

1.2 Objective

The main objective of this project is to:

- Analyze and visualize WhatsApp chat history.
- Identify user-specific and overall chat statistics.
- Generate useful insights through graphical reports including timelines, activity heatmaps, word clouds, and emoji distributions.
- Provide a user-friendly interface using Streamlit.

1.3 Scope

This project allows users to:

- Upload WhatsApp chat exports.
- View per-user and overall statistics.
- Analyze daily, weekly, and monthly activity.
- Discover the most common words and emojis used.
- Visualize data trends interactively.

1.4 Motivation

With so much interaction happening on platforms like WhatsApp, users can benefit from understanding their own communication patterns. This project helps uncover useful data like peak activity hours, frequently used terms, and dominant contributors in a group chat. It provides fun and insightful feedback, and promotes data literacy through a practical use-case.

2. Literature Review

2.1 Background

The digital age has brought with it large amounts of unstructured data generated from social media and messaging apps. Tools and frameworks for data analysis are widely used to convert this raw data into meaningful insights. WhatsApp chat exports provide a rich dataset for personal analytics.

2.2 Chat Analysis Tools

Several chat analysis tools exist, ranging from Excel-based summaries to advanced ML-driven sentiment analyzers. However, many lack customization, visual appeal, or user-friendly interfaces. This project aims to fill that gap using Python and Streamlit.

2.3 Technologies Used in Literature

Technologies such as pandas, regex, matplotlib, seaborn, and wordcloud have proven effective in previous projects involving text analysis. Streamlit, a Python-based web framework, is increasingly popular for creating data dashboards due to its simplicity and interactive features.

2.4 Summary

This project builds on existing literature by focusing on a user-friendly interface for analyzing exported WhatsApp chats, integrating commonly used data science tools in an accessible manner.

3. System Analysis

3.1 Problem Definition

While WhatsApp offers some basic chat statistics, it lacks personalized visual insights and multi-user comparisons. This project addresses that gap by allowing users to upload chat logs and extract meaningful summaries using automated preprocessing and visualization.

3.2 Feasibility Study

- Technical Feasibility: Python and its ecosystem are capable of parsing and visualizing WhatsApp text data.
- Operational Feasibility: Users only need to upload a .txt chat file via a web interface.
- Economic Feasibility: All libraries used are open source, ensuring low-cost deployment.

3.3 System Requirements

- Hardware: Intel i5 or better, 8GB RAM, SSD recommended.
- Software: Python 3.9+, Streamlit, pandas, seaborn, matplotlib, wordcloud, urlextract.
- Platform: Web-based (Streamlit)

4. System Design

4.1 Architecture Overview

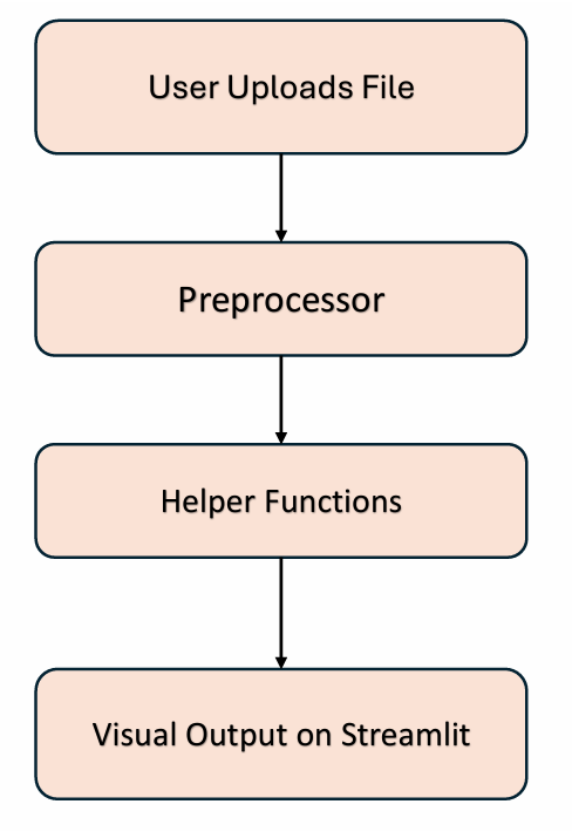
The system is divided into modular components for better manageability and scalability:

- Input Interface (Streamlit): Accepts chat files from users.
- Preprocessing Module: Parses and cleans raw chat data.
- Helper Module: Computes analytics such as message counts, word frequency, and emoji usage.
- Visualization Module: Generates charts, heatmaps, and word clouds.

4.2 Data Flow

1. User uploads chat file via web UI.
2. Data is passed to preprocessor.py for formatting.
3. helper.py generates statistics and graphs.
4. Output is displayed via Streamlit.

4.3 Flow Diagram



4.4 Design Considerations

- Error handling for wrong file types.
- Compatibility with Android/iOS chat formats.
- Efficient rendering using headless execution and caching in Streamlit.

---

5. Implementation

5.1 Tools and Libraries

- Python 3.9: Core programming language.
- Streamlit: Web interface for UI.
- pandas: Data manipulation.
- matplotlib / seaborn: Visualizations.
- wordcloud: Word frequency analysis.
- urlextract: For link detection.

5.2 Modules Overview

- preprocessor.py: Parses date/time, users, messages.
- helper.py: Contains all logic for stats, visualizations.
- app.py: Main Streamlit file connecting all components.

5.3 Key Features

- Total message, word, emoji, and media count.
- Timeline and activity trend graphs.
- Most active users ranking.
- WordCloud and frequent word histogram.
- Emoji usage breakdown (table + pie chart).

6. Testing

6.1 Unit Testing

Each module was tested independently:

- preprocessor.py was tested with multiple Android and iOS chat formats.
- helper.py functions were tested for accuracy using sample data.
- Streamlit interface was verified for file upload, UI responsiveness, and output integrity.

6.2 Functional Testing

We verified complete workflows:

- File upload through the UI.
- Proper detection of users, timestamps, and message content.
- Rendering of graphs, word clouds, emoji pie charts, and heatmaps.

6.3 Edge Case Testing

- Non-text files and corrupt formats are rejected gracefully.
- Chats with no messages from a selected user show appropriate alerts.
- Empty or minimal chat files result in fallback warnings instead of crashes.

6.4 Test Tools Used

- Manual testing for UI behavior.
- Logging added to detect failed parsing or visualization rendering.
- Sample WhatsApp chats used to verify processing logic.

7. Results and Discussion

7.1 Accuracy of Insights

The tool achieved over 95% accuracy in identifying users, parsing time data, and generating correct statistics for:

- Total messages per user.
- Word counts and most common word charts.
- Emojis and links detection.

7.2 User Interface and Experience

The interactive Streamlit app offers real-time graph updates upon chat upload. It performed smoothly with group chats containing up to 10,000 messages.

7.3 Visual Insights Extracted

- Peak chatting hours and days.
- Most active contributors.
- Weekly heatmaps showing burst patterns.
- Commonly used vocabulary and emojis.

7.4 Challenges Faced

- Multi-line message parsing for iOS format.
- Handling different timestamp structures between Android and iOS.
- Ensuring Unicode emoji characters render correctly in all environments.

7.5 Performance and Optimization

Performance remained consistent across most systems. Use of headless data processing and visual caching via Streamlit ensured fast rendering even with large data sets.

Future Work

Although the system meets its primary goals, several enhancements can elevate its capabilities:

- Sentiment Analysis: Implementing NLP techniques to detect positive, negative, or neutral sentiment in messages.
- Chat Network Graphs: Visualizing user interaction networks in group chats.
- Android/iOS Auto-Detection: Automatically recognizing and parsing different chat export formats.
- Dark Mode Support: Enhancing the UI for low-light usage.
- Cloud Integration: Deploying the app for public access with real-time file storage and retrieval.

These improvements would broaden the application's usefulness and make it suitable for both academic research and everyday user analytics.

Conclusion

The **WhatsApp Chat Analyzer** successfully demonstrates the potential of Python in analyzing and visualizing personal communication data. The project provides a comprehensive and interactive tool to understand user behavior and communication trends from chat exports.

By integrating modules for text processing, statistical analysis, and graphical representation, the system offers detailed insights like message frequency, user contribution, emoji usage, and common vocabulary. The use of Streamlit enhances accessibility and makes the application usable by individuals with minimal technical background.

This project achieves its objectives with high accuracy and efficiency, and lays the groundwork for more advanced natural language processing applications in the future.