```
# project :- https://colab.research.google.com/drive/1ZD60nPbE2z16tiE9sZyGY_etS7KZ7suB?usp=sharing
```

WhatsApp data analysis

WhatsApp chat analysis provided insights into the chat data, including word frequencies, message patterns over time, and participation trends. These visualizations offer a comprehensive overview of the chat dynamics and can assist in extracting meaningful information from the chat log.

The data i used on this project is my personal chat history with my friend. Whatsapp allows their users to export the data of their conversation with a particular chat or group.

```
!pip install calmap
```

```
Collecting calmap
  Downloading calmap-0.0.11-py2.py3-none-any.whl.metadata (2.2 kB)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from calmap) (3.8.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from calmap) (1.26.4)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from calmap) (2.2.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (1.3.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (4.54.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->calmap) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->calmap) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas->calmap) (2024.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->calmap)
Downloading calmap-0.0.11-py2.py3-none-any.whl (7.3 kB)
Installing collected packages: calmap
Successfully installed calmap-0.0.11
```

```
!pip install emoji
```

```
Collecting emoji
  Downloading emoji-2.14.0-py3-none-any.whl.metadata (5.7 kB)
Downloading emoji-2.14.0-py3-none-any.whl (586 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 586.9/586.9 kB 7.8 MB/s eta 0:00:00
Installing collected packages: emoji
Successfully installed emoji-2.14.0
```

Mounting google drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Importing necessary libraries

```
import re
import regex
import pandas as pd
import numpy as np
import emoji
import plotly.express as px
from collections import Counter
import matplotlib.pyplot as plt
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
```

Data Preparation:

This plain text file will have to be parsed and tokenized in a meaningful manner in order to be served (stored) in a Pandas dataframe.

Let us consider just a single line from the text (which we will call “raw text”) and see how we can extract relevant columns from it:

18/06/17, 9:47 PM - Teja: Why do you have 2 numbers?

In our sample line of text, our main objective is to automatically break down the raw message into 4 tokens.

{Date}, {Time} - {Author}: {Message}

{18/06/17}, {9:47 PM} - {Teja}: {Why do you have 2 numbers?}

```
def date_time(s):
  # Your regex pattern
    pattern = '^([0-9]+)(\/)([0-9]+)(\/)([0-9]+), ([0-9]+):([0-9]+)[ ]?(AM|PM|am|pm)? -'
    # Match the pattern against the string
    result = regex.match(pattern, s)
     # Check if there was a match and return True or False
    if result:
        return True
    return False

def find_author(s):
   # Split the string at the colon
    s = s.split(":")
    # Check if there are exactly two parts
    if len(s)==2:
        return True
    else:
        return False

def getDatapoint(line):
   # Split the line by the delimiter ' - '
    splitline = line.split(' - ')
    # Extract the date and time from the first part
    dateTime = splitline[0]

    date, time = dateTime.split(", ")
    # Combine the rest of the splitline into a message
    message = " ".join(splitline[1:])
     # Check if the message contains an author
    if find_author(message):
        splitmessage = message.split(": ")
        author = splitmessage[0]
        message = " ".join(splitmessage[1:])
    else:
        author= None
    return date, time, author, message
```

```
cd /content/drive/MyDrive/create
```

```
/content/drive/MyDrive/create
```

[To analyze your chat file make changes in the cell below]

```
data = []  # List to keep track of data so it can be used by a Pandas dataframe

conversation = '/content/drive/MyDrive/create/chart45.txt'
with open(conversation, encoding="utf-8") as fp:
    fp.readline() # Skipping first line of the file because contains information related to something about end-to-end encryption
    messageBuffer = []
    date, time, author = None, None, None
    while True:
        line = fp.readline()
        if not line:
            break

        line = line.strip()
        if date_time(line):
            if len(messageBuffer) > 0: # If there are messages to save
                data.append([date, time, author, ' '.join(messageBuffer)])
            messageBuffer.clear()
            date, time, author, message = getDatapoint(line)
            messageBuffer.append(message)
        else:
            messageBuffer.append(line)


def getDatapoint(line):
   # Split the line by the delimiter ' - '
    splitline = line.split(' - ')
    # Extract the date and time from the first part
    dateTime = splitline[0]

    date, time = dateTime.split(", ")
```

```
    # Combine the rest of the splitline into a message
    message = " ".join(splitline[1:])
     # Check if the message contains an author
    if find_author(message):
        splitmessage = message.split(": ")
        author = splitmessage[0]
        message = " ".join(splitmessage[1:])
    else:
        author= None
    return date, time, author, message
```

Data formatting

```
from datetime import datetime
# Assuming 'data' is the list created in the previous cell
# Extract messages and dates from 'data'
messages = [item[3] for item in data]
dates = [item[0] + ', ' + item[1] for item in data]  # Combine date and time

# Now create the DataFrame
df = pd.DataFrame({'user_message': messages, "message_date": dates})

def convert24(time):
    time = time.split(",")
    # Parse the time string into a datetime object
    t = datetime.strptime(time[1].strip(" -"), '%I:%M %p') # Use the imported datetime class
    # Format the datetime object into a 24-hour time string
    ta = time[0] + "," + t.strftime('%H:%M')
    return ta

# convert into 24hr format
df['message_date'] = df['message_date'].apply(lambda x: convert24(x))
df["message_date"] = pd.to_datetime(df["message_date"], format="%d/%m/%y,%H:%M")
df.tail()
```

	user_message	message_date
3532	*TODAY IS THE LAST DAY TO GIVE YOUR WEB MINING...	2024-04-27 08:26:00
3533	NLP ka project kon kon submitted kiya?	2024-04-28 17:14:00
3534	😄	2024-04-28 19:40:00
3535	+91 97683 13673: https://docs.google.com/sprea...	2024-04-29 08:17:00
3536	Faculty ko forward kar ra hu....jisne bhi nhi ...	2024-04-29 08:17:00

```
df.shape
```

```
(3537, 2)
```

```
users=[]
messages=[]
for message in df["user_message"]:
    entry = re.split("([\w+\W]+?):\s",message)
    if entry[1:]:
        users.append(entry[1])
        messages.append(entry[2])
    else:
        users.append("group_notification")
        messages.append(entry[0])
df['user']=users
df['message']=messages
df.drop(columns=['user_message'],inplace=True)
```

```
df.head()
```

	message_date	user	message
0	2023-10-09 09:32:00	group_notification	Areej Clg created group "notes~"
1	2023-10-18 09:07:00	group_notification	You joined using this group's invite link
2	2023-10-18 10:02:00	group_notification	Ohh
3	2023-10-18 10:02:00	group_notification	Areej Clg added +91 99676 09749
4	2023-10-18 10:57:00	group_notification	<Media omitted>

Next steps: Generate code with df | View recommended plots | New interactive sheet

Preparing extra columns

```
df['day'] = df['message_date'].dt.strftime('%a')
df['month'] = df['message_date'].dt.strftime('%b')
df['year'] = df['message_date'].dt.year
df['date'] = df['message_date'].apply(lambda x: x.date())
```

```
df
```

	message_date	user	message	day	month	year	date
0	2023-10-09 09:32:00	group_notification	Areej Clg created group "notes~"	Mon	Oct	2023	2023-10-09
1	2023-10-18 09:07:00	group_notification	You joined using this group's invite link	Wed	Oct	2023	2023-10-18
2	2023-10-18 10:02:00	group_notification	Ohh	Wed	Oct	2023	2023-10-18
3	2023-10-18 10:02:00	group_notification	Areej Clg added +91 99676 09749	Wed	Oct	2023	2023-10-18
4	2023-10-18 10:57:00	group_notification	<Media omitted>	Wed	Oct	2023	2023-10-18
...	...	...	...	...	...	...	...
3532	2024-04-27 08:26:00	group_notification	*TODAY IS THE LAST DAY TO GIVE YOUR WEB MINING...	Sat	Apr	2024	2024-04-27
3533	2024-04-28 17:14:00	group_notification	NLP ka project kon kon submitted kiya?	Sun	Apr	2024	2024-04-28
3534	2024-04-28 19:40:00	group_notification	😂	Sun	Apr	2024	2024-04-28
3535	2024-04-29 08:17:00	+91 97683 13673	https://docs.google.com/spreadsheets/d/1r5F4fE...	Mon	Apr	2024	2024-04-29
3536	2024-04-29 08:17:00	group_notification	Faculty ko forward kar ra hu....jisne bhi nhi ...	Mon	Apr	2024	2024-04-29

3537 rows × 7 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3537 entries, 0 to 3536
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   message_date  3537 non-null   datetime64[ns]
 1   user          3537 non-null   object
 2   message       3537 non-null   object
 3   day           3537 non-null   object
 4   month         3537 non-null   object
 5   year          3537 non-null   int32
 6   date          3537 non-null   object
 7   emoji         3537 non-null   object
dtypes: datetime64[ns](1), int32(1), object(6)
memory usage: 207.4+ KB
None
```

Printing Participants

```
df.user.unique()
```

```
array(['group_notification', 'Areej Clg', '+91 96199 36420',
       '+91 99676 09749',
       'connectionless, no acknowledgement, out of order packets we will discard them Internet protocol, service is insecure,
connection-oriented service',
       'Shraddha Panchal', '+91 90042 80656', '+91 97683 13673',
       'Shilpa Dhanure', '+91 87790 51155', '+91 90040 75303',
       'Anshu Clg Mumbai', 'Monika Kharkwal', 'Preeti Clg Mumbai',
       'Pratiksha Awate',
       "SDN ka if your topics for project are approved then ye list mai add karo so that baadme clashes na ho.... Pahile ma'am
ko pucho and approve hone ke baad hi add karo ye list mai 👀  • Vehicular AdHoc Network (VANET) - Pratiksha • IPv4 and IPv6
Addressing - Nijin • Cloud Computing - Vamsi • IOT AND ITS COMPONENTS - ULLAS • WAN - Anshu • Application Layer Protocol - Areej
• Transport Layer Protocol - Swapnil •Design and Implementation of A New Location Management Scheme in Mobile Ad Hoc Network-
Vasudha Arora • Unicast Routing Protocol ",
       '+91 77383 28626',
       "*Algorithms for Optimization Project Topics* 1)Newton's method - VamsiKrishna 2) Adagrad - ULLAS 3)Rosenbrock Function -
Anshu 4) Adadelta - Swapnil 5) Booth Function - Areej 6) Genetic Algorithm - Ashwin 7) Momentum   - Pratiksha 8) Gradient
descent- Tejas 9) Styblinski-Tang Function - Avinash 10) Conjugate Gradient - Shilpa 11) Matyas function - Shraddha 12) Camel
function - Monika Kharkwal 13) BEALE FUNCTION - NIJIN 14) McCormick function - preeti 15) Secant method",
       '+91 70234 89118',
       "*Algorithms for Optimization Project Topics* 1)Newton's method - VamsiKrishna 2) Ant colony optimization- ULLAS
```

```
3)Rosenbrock Function - Anshu 4) Adadelta - Swapnil 5) Booth Function - Areej 6) Genetic Algorithm - Ashwin 7) Momentum   -
Pratiksha 8) Gradient descent- Tejas 9) Styblinski-Tang Function - Avinash 10) Conjugate Gradient - Shilpa 11) Matyas function -
Shraddha 12) Camel function - Monika Kharkwal 13) BEALE FUNCTION - NIJIN 14) McCormick function - preeti 15) Secant method",
       '+91 87793 59887', '+91 82916 85824', '+91 70392 29744',
       '+91 70234 77723',
       '*upsampling and downsampling* the process of converting the sampling rate of a digital signal from one rate to another
is Sampling Rate Conversion. Increasing the rate of already sampled signal is Upsampling whereas decreasing the rate is called
downsampling.  *Upsampling* is the process of inserting zero-valued samples between original samples to increase the sampling
rate. (This is sometimes called “zero-stuffing”.) This kind of upsampling adds undesired spectral images to the original signal,
which are centered on multiples of the original sampling rate.  The *Fourier Transform* is an important image processing tool
which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image
in the Fourier or frequency domain, while the input image is the spatial domain equivalent.  The *Fast Fourier Transform (FFT)*
is commonly used to transform an image between the spatial and frequency domain.FFT method preserves all original data & FFT
fully transforms images into the frequency domain.  What type of Fourier transformation is used for image processing? In the
frequency domain, pixel location is represented by its x- and y-frequencies and its value is represented by amplitude. The Fast
Fourier Transform (FFT) is commonly used to transform an image between the spatial and frequency domain.  *discrete Fourier
transform* The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a
set of samples which is large enough to fully describe the spatial domain image.  *convolution in digital image processing* In
image processing, convolution is the process of transforming an image by applying a kernel over each pixel and its local
neighbors across the entire image. The kernel is a matrix of values whose size and values determine the transformation effect of
the convolution process.  *What is template matching in image processing?* Image result for Template Matching in digital image
processing Template matching is a technique in digital image processing for finding small parts of an image which match a
template image. It can be used in manufacturing as a part of quality control, a way to navigate a mobile robot, or as a way to
detect edges in images.  *Log transformation* s = c log(r + 1) Where s and r are the pixel values of the output and the input
image and c is a constant. The value 1 is added to each of the pixel value of the input image because if there is a pixel
intensity of 0 in the image, then log (0) is equal to infinity. So 1 is added, to make the minimum value at least 1. During log
transformation, the dark pixels in an image are expanded as compare to the higher pixel values. The value of c in the log
transform adjust the kind of enhancement you are looking for.  *Power law transformation* The power-law transformation is given
as',
       'import csv import mysql.connector import pandas as pd  product_data = pd.read_csv(\'D:\\\\Amazon_product.csv\',
index_col=False, delimiter = ",") product_data.head()  conn = mysql.connector.connect(host = "localhost", user = "root", passwd
= "1234")  mycursor = conn.cursor()  mycursor.execute("use sys") mycursor.execute("DROP TABLE IF EXISTS Amazon_prod;")
mycursor.execute("CREATE TABLE Amazon_prod(Sr_no VARCHAR(10), P_name VARCHAR(200),REVIEW VARCHAR(30),PRICES VARCHAR(100))") for
i,row in product_data.iterrows()',
       '✨“First, solve the problem. Then, write the code.” ✨ ~John Johnson  🛡 *Department of Computer Science of SIES
(Nerul) College of Arts, Science and Commerce,NAAC Reaccredited with Grade A in 3rd Cycle* 🛡  💥PRESENTS 💥  💻 *CODE DVANDVA*
💻 -A Coding Competition  📍 *Objectives* ',
       "```“Research is seeing what everybody else has seen and thinking what nobody else has thought.”``` ~Albert Szent-Györgyi
```

Group Wise Stats

```
total_messages = df.shape[0]
print(total_messages)
```

```
3537
```

Let us now find out the total Media Messages

```
media_messages = df[df['message'] == '<Media omitted>'].shape[0]
print(media_messages)
```

```
314
```

Messages sent by each user

```
messages_by_user = df['user'].value_counts()
messages_by_user
```

user	count
group_notification	3397
Areej Clg	32
+91 97683 13673	24
Pratiksha Awate	11
Shraddha Panchal	9
+91 70234 89118	9
+91 90042 80656	8
+91 99676 09749	5
+91 77383 28626	5
+91 96199 36420	4
+91 82916 85824	3
SDN ka if your topics for project are approved then ye list mai add karo so that baadme clashes na ho.... Pahile ma'am ko pucho and approve hone ke baad hi add karo ye list mai 👀 • Vehicular AdHoc Network (VANET) - Pratiksha • IPv4 and IPv6 Addressing - Nijin • Cloud Computing - Vamsi • IOT AND ITS COMPONENTS - ULLAS • WAN - Anshu • Application Layer Protocol - Areej • Transport Layer Protocol - Swapnil •Design and Implementation of A New Location Management Scheme in Mobile Ad Hoc Network-Vasudha Arora • Unicast Routing Protocol	3
Anshu Clg Mumbai	3
+91 90040 75303	3
+91 87790 51155	2
Algorithms for Optimization Project Topics 1)Newton's method - VamsiKrishna 2) Adagrad - ULLAS 3)Rosenbrock Function - Anshu 4) Adadelta - Swapnil 5) Booth Function - Areej 6) Genetic Algorithm - Ashwin 7) Momentum - Pratiksha 8) Gradient descent- Tejas 9) Styblinski-Tang Function - Avinash 10) Conjugate Gradient - Shilpa 11) Matyas function - Shraddha 12) Camel function - Monika Kharkwal 13) BEALE FUNCTION - NIJIN 14) McCormick function - preeti 15) Secant method	2
Shilpa Dhanure	2
+91 87793 59887	2
+91 70392 29744	2
Monika Kharkwal	1
import csv import mysql.connector import pandas as pd product_data = pd.read_csv('D:\\Amazon_product.csv', index_col=False, delimiter = ",") product_data.head() conn = mysql.connector.connect(host = "localhost", user = "root", passwd = "1234") mycursor = conn.cursor() mycursor.execute("use sys") mycursor.execute("DROP TABLE IF EXISTS Amazon_prod;") mycursor.execute("CREATE TABLE Amazon_prod(Sr_no VARCHAR(10), P_name VARCHAR(200),REVIEW VARCHAR(30),PRICES VARCHAR(100))") for i,row in product_data.iterrows()	1
ASSIGNMENT 1 Q.1 WHAT IS WEB MINING? ANS	1
```“Research is seeing what everybody else has seen and thinking what nobody else has thought.”``` ~Albert Szent-Györgyi 🔰 *Department of Computer Science* *SIES(Nerul) College of Arts, Science and Commerce* *NAAC Re-accredited with 'A' Grade in 3rd Cycle* 🔰 💥 PRESENTS💥 📄 _*RESEARCH PAPER REVIEW COMPETITION*_ 📄 🔵Objectives:- 🔶 To inculcate a habit of research. 🔶 To develop a competitive environment in research. 🔶 To help students explore the field of research and learn the craft of research paper writing. 🔵Rules :- ▫Participants can select the topics as per their choice. ▫Undergraduate and postgraduate students of all streams of SIES ASCN. ▫Last date for submission of the paper is *5th March, 2024*. ▫Competition date	1
✨“First, solve the problem. Then, write the code.” ✨ ~John Johnson 🛡 *Department of Computer Science of SIES (Nerul) College of Arts, Science and Commerce,NAAC Reaccredited with Grade A in 3rd Cycle* 🛡 💥PRESENTS 💥 💻 *CODE DVANDVA* 💻 -A Coding Competition 📍*Objectives*	1
connectionless, no acknowledgement, out of order packets we will discard them Internet protocol, service is insecure, connection-oriented service	1
upsampling and downsampling the process of converting the sampling rate of a digital signal from one rate to another is Sampling Rate Conversion. Increasing the rate of already sampled signal is Upsampling whereas decreasing the rate is called downsampling. *Upsampling* is the process of inserting zero-valued samples between original samples to increase the sampling rate. (This is sometimes called “zero-stuffing”.) This kind of upsampling adds undesired spectral images to the original signal, which are centered on multiples of the original sampling rate. The *Fourier Transform* is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. The *Fast Fourier Transform (FFT)* is commonly used to transform an image between the spatial and frequency domain.FFT method preserves all original data & FFT fully transforms images into the frequency domain. What type of Fourier transformation is used for image processing? In the frequency domain, pixel location is represented by its x- and y-frequencies and its value is represented by amplitude. The Fast Fourier Transform (FFT) is commonly used to transform an image between the spatial and frequency domain. *discrete Fourier transform* The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. *convolution in digital image processing* In image processing, convolution is the process of transforming an image by applying a kernel over each pixel and its local neighbors across the	1

Messages sent by each user in %

```
percent_messages_by_user = 100*(messages_by_user)/df.shape[0]
p = round(percent_messages_by_user, 2)
p_D = pd.DataFrame({'email':p.index, 'list':p.values})
```

p_D

	email	list
0	group_notification	96.04
1	Areej Clg	0.90
2	+91 97683 13673	0.68
3	Pratiksha Awate	0.31
4	Shraddha Panchal	0.25
5	+91 70234 89118	0.25
6	+91 90042 80656	0.23
7	+91 99676 09749	0.14
8	+91 77383 28626	0.14
9	+91 96199 36420	0.11
10	+91 82916 85824	0.08
11	SDN ka if your topics for project are approved...	0.08
12	Anshu Clg Mumbai	0.08
13	+91 90040 75303	0.08
14	+91 87790 51155	0.06
15	*Algorithms for Optimization Project Topics* 1...	0.06
16	Shilpa Dhanure	0.06
17	+91 87793 59887	0.06
18	+91 70392 29744	0.06
19	Monika Kharkwal	0.03
20	import csv import mysql.connector import panda...	0.03
21	ASSIGNMENT 1 Q.1 WHAT IS WEB MINING? ANS	0.03
22	```“Research is seeing what everybody else has...	0.03
23	✨“First, solve the problem. Then, write the co...	0.03
24	connectionless, no acknowledgement, out of ord...	0.03
25	*upsampling and downsampling* the process of c...	0.03
26	+91 70234 77723	0.03
27	Preeti Clg Mumbai	0.03
28	*Algorithms for Optimization Project Topics* 1...	0.03
29	+91 84259 79051	0.03

Next steps: Generate code with p_D | View recommended plots | New interactive sheet

Messages sent per day over a time period

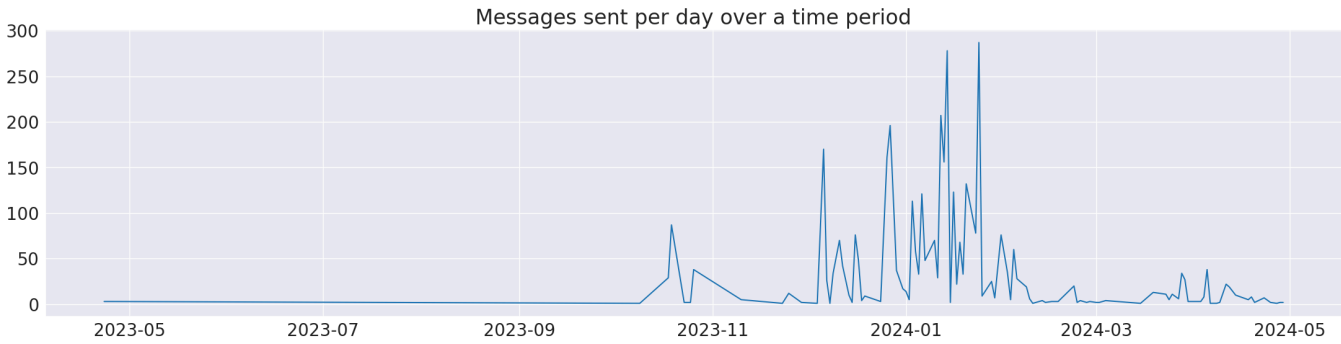
```
import pandas as pd

df1 = df.copy()
df1['message_count'] = [1] * df1.shape[0]
df1.drop(columns='year', inplace=True)
# Exclude the 'date' and other datetime columns from the aggregation
# by explicitly selecting the columns you want to sum.
df1 = df1.groupby('date')['message_count'].sum().reset_index()
df1
```

	date	message_count
0	2023-04-23	3
1	2023-04-25	3
2	2023-10-09	1
3	2023-10-18	29
4	2023-10-19	87
...	...	...
95	2024-04-23	7
96	2024-04-25	2
97	2024-04-27	1
98	2024-04-28	2
99	2024-04-29	2

100 rows × 2 columns

Next steps: Generate code with df1 | View recommended plots | New interactive sheet

```
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib


df1 = df.copy()
df1['message_count'] = [1] * df1.shape[0]
df1.drop(columns='year', inplace=True)
# Exclude the 'date' and other datetime columns from the aggregation
# by explicitly selecting the columns you want to sum.
df1 = df1.groupby('date')['message_count'].sum().reset_index()


sns.set_style("darkgrid")
matplotlib.rcParams['font.size'] = 20
matplotlib.rcParams['figure.figsize'] = (27, 6)

plt.plot(df1.date, df1.message_count)
plt.title('Messages sent per day over a time period');
```



Top 10 days where maximum number of texts sent

```
top10days = df1.sort_values(by="message_count", ascending=False).head(10)
top10days.reset_index(inplace=True)
top10days.drop(columns="index", inplace=True)
top10days
```

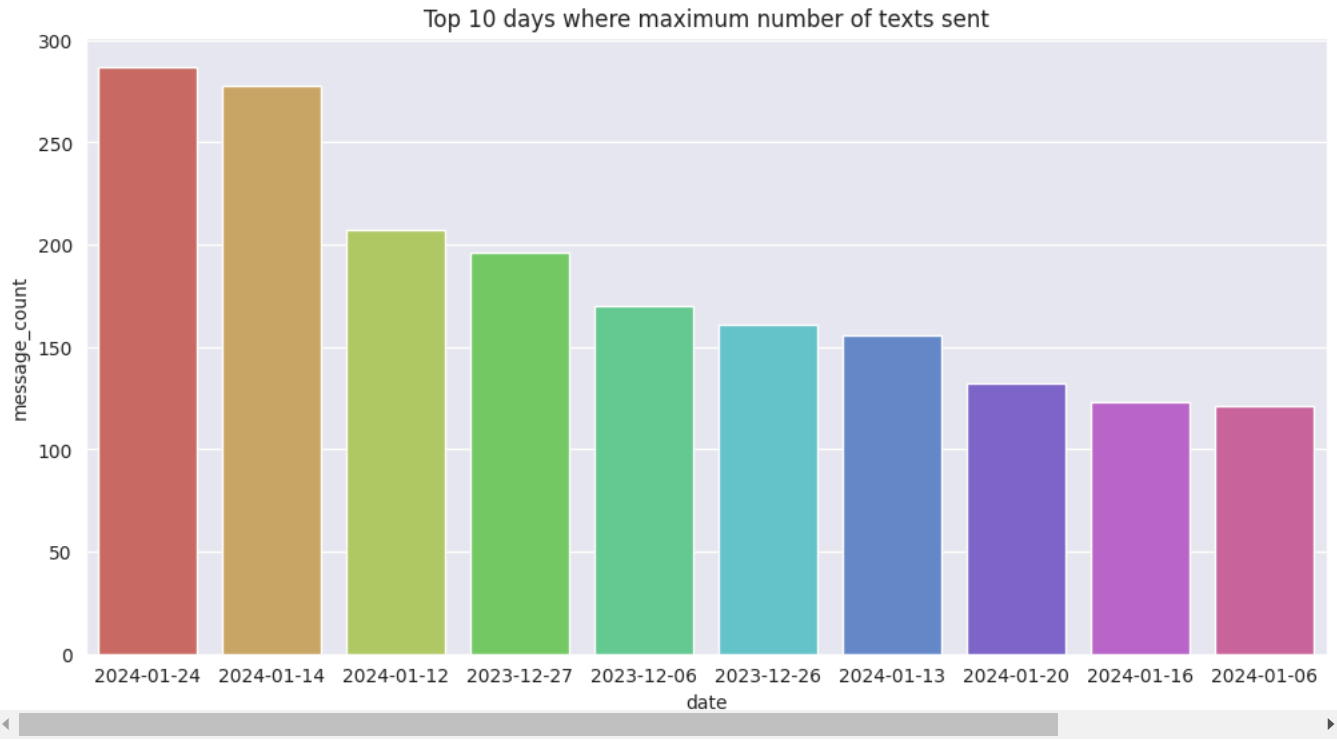
```
<ipython-input-23-527e2da8f452>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  top10days.drop(columns="index", inplace=True)
```

	date	message_count
0	2024-01-24	287
1	2024-01-14	278
2	2024-01-12	207
3	2023-12-27	196
4	2023-12-06	170
5	2023-12-26	161
6	2024-01-13	156
7	2024-01-20	132
8	2024-01-16	123
9	2024-01-06	121

Next steps: Generate code with top10days | View recommended plots | New interactive sheet

```
sns.set_style("darkgrid")

matplotlib.rcParams['font.size'] = 10
matplotlib.rcParams['figure.figsize'] = (12, 6)
sns.barplot(x=top10days.date, y=top10days.message_count, palette="hls")
plt.title('Top 10 days where maximum number of texts sent');
```

```
<ipython-input-24-918f08b6e3e1>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=top10days.date, y=top10days.message_count, palette="hls")
```



Most used Emoji in Group

Unique emojis used in group

```
import emoji
import regex
import pandas as pd

def split_count(text):
    emoji_list = []
    data = regex.findall(r'\X', text)
    for word in data:
```

```
    for word in data:
        # Use emoji.is_emoji() to check for emojis
        if any(emoji.is_emoji(char) for char in word):
            emoji_list.append(word)
    return emoji_list

# Assuming the column containing the message text is 'message'
# Change 'message' to the actual column name if it's different.
# The error was occuring because 'Message_Clean' column name was used whereas the actual column name is 'message'.
df['emoji'] = df["message"].apply(split_count)

emojis = sum(df['emoji'].str.len())
print(emojis)
```

```
748
```

```
media_messages_df = df[df['message'] == '<Media omitted>'] # Filter out media messages, changed 'Message' to 'message'
messages_df = df.drop(media_messages_df.index)
messages_df['Letter_Count'] = messages_df['message'].apply(lambda s : len(s)) # Calculate letter count, changed 'Message' to 'message'
messages_df['Word_Count'] = messages_df['message'].apply(lambda s : len(s.split(' '))) # Calculate word count, changed 'Message' to 'mess
messages_df["MessageCount"]=1 # Create a MessageCount column
```

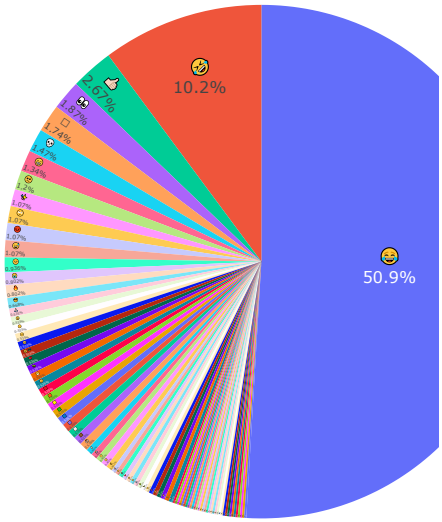
```
total_emojis_list = list(set([a for b in messages_df.emoji for a in b]))  # Extract unique emojis
total_emojis = len(total_emojis_list)

total_emojis_list = list([a for b in messages_df.emoji for a in b]) # Count all emojis
emoji_dict = dict(Counter(total_emojis_list))
emoji_dict = sorted(emoji_dict.items(), key=lambda x: x[1], reverse=True)
for i in emoji_dict:
  print(i)
```

```
('🟫', 4)
('🎋', 4)
('✌️', 3)
('🤦\u200d♀', 3)
('🙃', 3)
('😊', 3)
('👌', 3)
('🤦', 2)
('🙂', 2)
('🤷\u200d♂', 2)
('❤️', 2)
('🙂', 2)
('😔', 2)
('🤦\u200d♂', 2)
('🐍', 2)
('🐛', 2)
('🤫', 2)
('🚨', 2)
('😵', 2)
('♥', 2)
```

```
('☝', 1)
('✋', 1)
('😅', 1)
('😉', 1)
('👆', 1)
```

[message show the large emoji count %](#)

```
emoji_df = pd.DataFrame(emoji_dict, columns=['emoji', 'count']) # Create DataFrame for emojis
import plotly.express as px
fig = px.pie(emoji_df, values='count', names='emoji') # Plotting with Plotly
fig.update_traces(textposition='inside', textinfo='percent+label')
fig.show()
```




```
line_list = df['message']
new_line_list = []

for word in line_list:
  emojis = emoji.distinct_emoji_list(word)
  new_line_list.extend([emoji.demojize(is_emoji) for is_emoji in emojis])
```

```
import collections # Import the collections module

new_line_counter = collections.Counter(new_line_list)

# ... (Rest of your code remains the same)
```

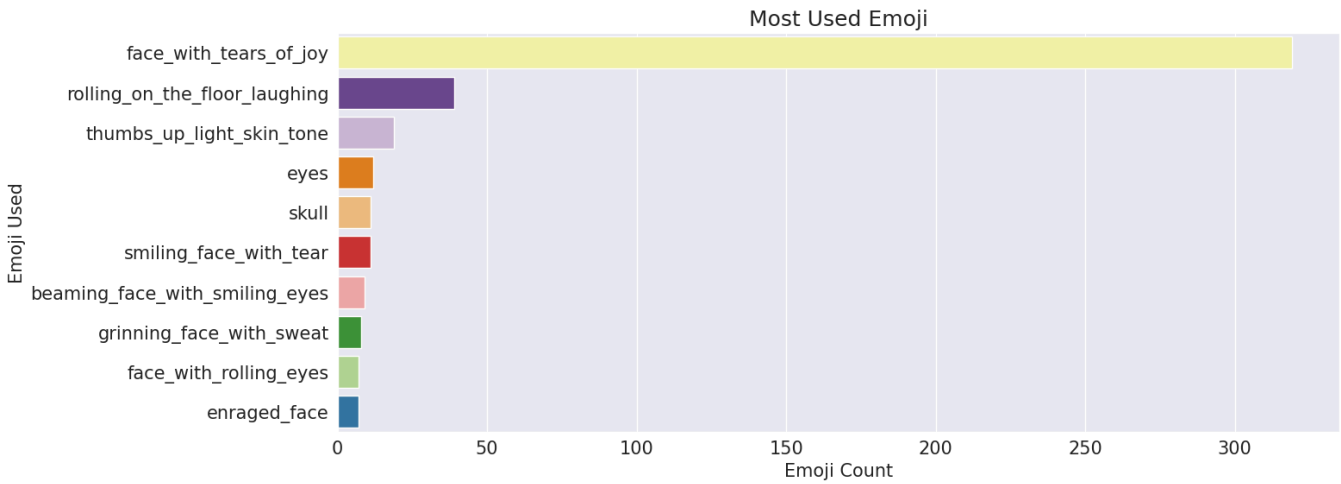
```
new_line_counter = collections.Counter(new_line_list)

top10emojis = pd.DataFrame()
top10emojis['emoji'] = [''] * 10
top10emojis['emoji_count'] = [0] * 10
top10emojis['emoji_description'] = [''] * 10

i = 0
for item in new_line_counter.most_common(10):
    emoji_char = item[0]
    description = emoji.demojize(emoji_char)[1:-1]

    # Convert emoji name to actual emoji character
    emoji_character = emoji.emojize(emoji_char)

    top10emojis.loc[i, 'emoji'] = emoji_character
    top10emojis.loc[i, 'emoji_count'] = int(item[1])
    top10emojis.loc[i, 'emoji_description'] = description
    i += 1

top10emojis
```

	emoji	emoji_count	emoji_description
0	😂	319	face_with_tears_of_joy
1	🤣	39	rolling_on_the_floor_laughing
2	👍	19	thumbs_up_light_skin_tone
3	👀	12	eyes
4	💀	11	skull
5	□	11	smiling_face_with_tear
6	😁	9	beaming_face_with_smiling_eyes
7	😅	8	grinning_face_with_sweat
8	🙄	7	face_with_rolling_eyes
9	😡	7	enraged_face

Next steps: Generate code with top10emojis | View recommended plots | New interactive sheet

```
plt.figure(figsize=(15, 6))
matplotlib.rcParams['font.size'] = 15
sns.set_style("darkgrid")
sns.barplot(x='emoji_count', y='emoji_description', data=top10emojis, palette='Paired_r')

plt.title('Most Used Emoji')
plt.xlabel('Emoji Count')
plt.ylabel('Emoji Used')
plt.show()
```

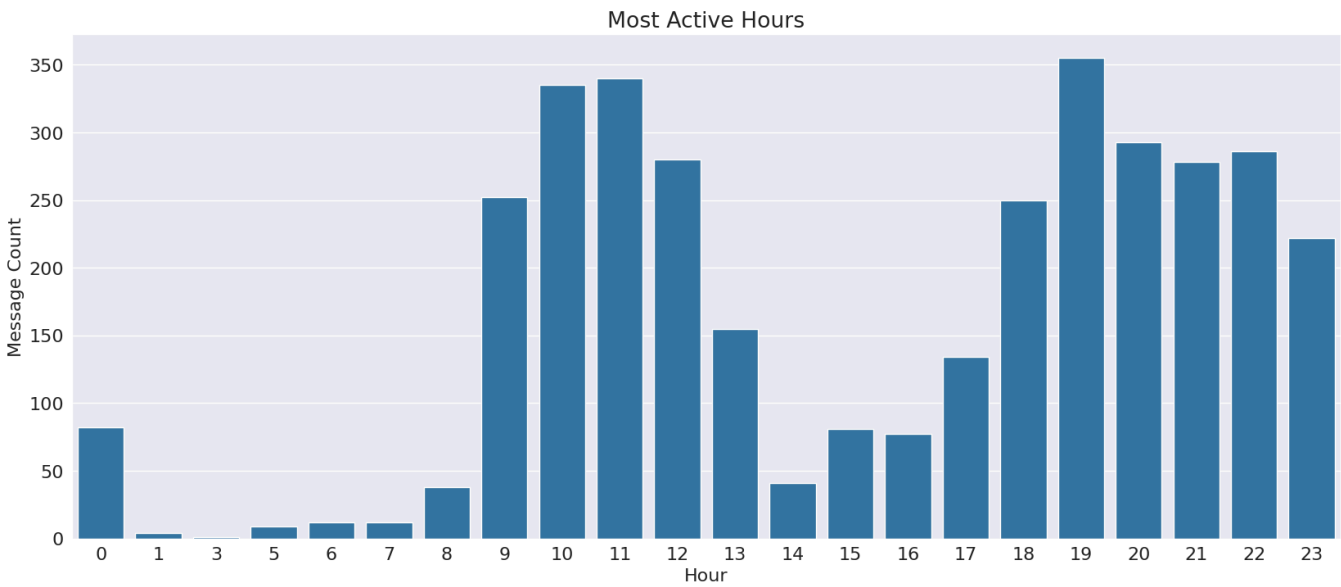
```
<ipython-input-32-a551b8f887eb>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `le
```



Most active hours

```
df3 = df.copy()
df3['message_count'] = [1] * df.shape[0]

df3['hour'] = df3['message_date'].apply(lambda x: x.hour)

grouped_by_time = df3.groupby('hour')['message_count'].sum().reset_index().sort_values(by = 'hour')
```

```
matplotlib.rcParams['font.size'] = 16
matplotlib.rcParams['figure.figsize'] = (20, 8)
sns.set_style("darkgrid")

sns.barplot(x=grouped_by_time.hour, y=grouped_by_time.message_count)
plt.title('Most Active Hours')
plt.xlabel('Hour')
plt.ylabel('Message Count')
plt.show()
```



Most used words group

```
# Replace 'Message' with the actual column name containing the messages.
# For example, if the column is named 'message', the code should be:
print(messages_df.columns) # Print the column names to check for the correct one

text = " ".join(review for review in messages_df['message']) # Use the appropriate column name here

print ("There are {} words in all the messages.".format(len(text)))
stopwords = set(STOPWORDS)
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)

plt.figure( figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

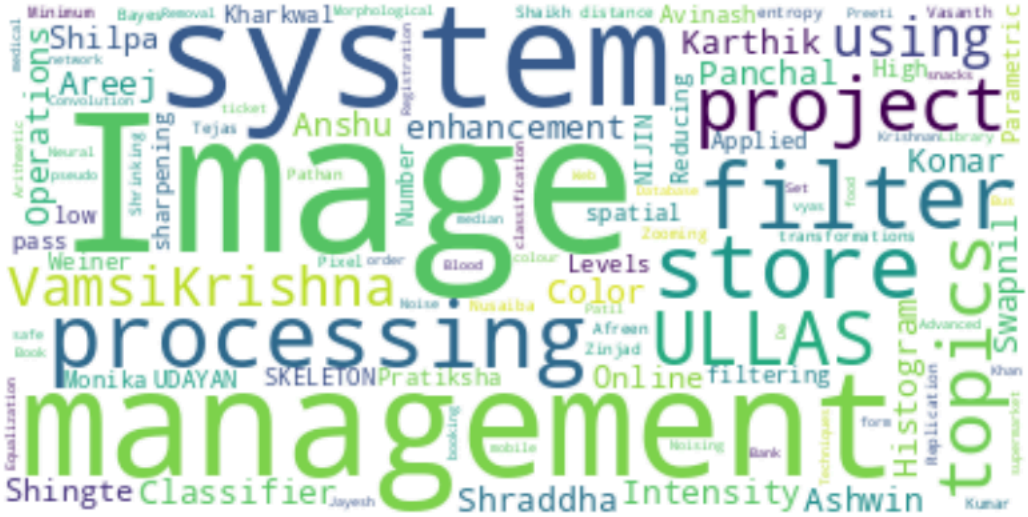
```
Index(['message_date', 'user', 'message', 'day', 'month', 'year', 'date',
       'emoji', 'Letter_Count', 'Word_Count', 'MessageCount'],
      dtype='object')
There are 142531 words in all the messages.
```

Most used words in differnt one user dataset analysis to another user

Output data analysis

```
messages_df = df  # Assign the dataframe 'df' to 'messages_df' to make the 'Author' column available
l = ["Shilpa Dhanure","Areej Clg", "Pratiksha Awate","Monika Kharkwal"]
# Assuming the column name is 'Author', replace 'Author' with 'author' in the code
# Print the available columns in the dataframe to verify
print(messages_df.columns)

for i in range(len(l)):
    # Access the column using the correct name, which appears to be 'Author' based on the traceback
    # Use 'author' instead of 'Author' to match the actual column name in the dataframe
    dummy_df = messages_df[messages_df['user'] == l[i]] # Indented this line
    # Replace 'Message' with the actual column name in your dataframe, likely 'message'
    text = " ".join(review for review in dummy_df['message']) # Indented this line and changed 'Message' to 'message'
    stopwords = set(STOPWORDS) # Indented this line
    # Generate a word cloud image
    print('Author name',l[i]) # Indented this line

    # Check if text is empty after stop word removal
    words = [word for word in text.split() if word.lower() not in stopwords] # Indented this line
    if len(words) == 0: # Indented this line
        print(f"No words found for author {l[i]} after removing stop words. Skipping word cloud generation.") # Indented this line
        continue  # Skip to next author if no words are found # Indented this line

    wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(" ".join(words)) # Indented this line
    # Display the generated image
    plt.figure( figsize=(10,5)) # Indented this line
    plt.imshow(wordcloud, interpolation='bilinear') # Indented this line
    plt.axis("off") # Indented this line
    plt.show() # Indented this line
```

```
Index(['message_date', 'user', 'message', 'day', 'month', 'year', 'date',
       'emoji'],
      dtype='object')
Author name Shilpa Dhanure
```




```
Author name Areej Clg
```