INTERNSHIP REPORT ON

WHATSAPP CHAT ANALYSIS WITH PYTHON

Submitted for the Partial Fulfilment of The Bachelor Degree of Computer Science

Submitted By

N. SRIVAISHNAVI (194228)

Y. SOMASEKHAR DINESH (194229)

To the Internship In charge

GVS.NARASIMHA

DEPARTMENT OF COMPUTER SCIENCE

ANDHRA LOYOLA COLLEGE

(AUTONOMOUS)

(2019-2022)

VIJAYAWADA-520008

ANDHRA LOYOLA COLLEGE
(AUTONOMOUS)
VIJAYAWADA

DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

This is to certificate that this project **"WHATSAPP CHAT ANALYSIS WITH PYTHON"** and bonafide work done by **N.SRI VAISHNAVI (194228), Y. SOMASEKHAR DINESH (194229)** of final year BSc Computer Science in the partial fulfilment of the award of Degree as a part of curriculum of "KRISHNA UNIVERSITY" for the academic year 2019-2022

HEAD OF DEPARTMENT

DATE:

LECTURER IN-CHARGE

DATE:

EXETERNAL EXAMINE

DATE:

ANDHRA LOYOLA COLLEGE

(AUTONOMOUS)

VIJAYAWADA

DEPARTMENT OF COMPUTER SCIENCE

DECLARTION

I declare that the project entitled **"WHATSAPP CHAT ANALYSIS WITH PYTHON"** is submitted by me in the partial fulfilment of the requirements for the award for the Degree of Bachelor of Computer Science, under the guidance and supervision of **GVS.NARASIMHA** the department of ComputerScience, Andhra Loyola College.

Signature of Student:

1. N. SRIVAISHNAVI (194228)
2. Y. SOMASEKHAR DINESH (194229)

ACKNOWLEDGEMENT

We sincerely thank **Mr. T. Mallikarjun** (Co-ordinator) for their timely advice and guidance regarding the project report.

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning the people who made it possible and whose Constant guidance and encouragement crown all the efforts with success.

We would like to take this opportunity to express our profound sense of gratitude to **S.A.B NEHRU.** Head of the Department of Computer Science, for his encouragement regard of the project. His annotations, insinuations and criticisms are the key behind the successful completion of the project and for providing us all the required guides.

In particular, we are very grateful to our Project Guide **Mr.GVS.Narasimha** the department of computer science, for her technical guidance and support throughout the project. We are deeply indebted for her support and cooperation. We profusely thank

Mr. Lakshman for providing good facilities and for helping us to make our project

Even though it is first time for us to work for a project it has been grate time of learning and coming to known the difficulties and constrains of software development. First and foremost, we thank to the management for giving us this opportunity by including the project in our curriculum

THANK YOU

INDEX

INTRODUCTION

1.ABSTRACT:

The most used and efficient method of communication in recent times is an application called WhatsApp. WhatsApp chats consists of various kinds of conversations held among group of people. This chat consists of various topics. This information can provide lots of data for latest technologies such as machine learning. The most important thing for a machine learning model is to provide the right learning experience which is indirectly affected by the data that we provide to the model. This tool aims to provide in depth analysis of this data which is provided by WhatsApp. Irrespective of whichever topic the conversation is based our developed code can be applied to obtain a better understanding of the data. The advantage of this tool is that is implemented using simple python modules such as pandas, matplotlib, seaborn and sentiment analysis which are used to create data frames and plot different graphs, where then it is displayed in the flutter application which is efficient and less resources consuming algorithm, therefor it can be easily applied to largest dataset.

1.2INDRODUCTION:

This tool is based on data analysis and processing. The first step in implementing a machine learning algorithm is to understand the right learning experience from which the model starts improving on. Data pre-processing plays a major role when it comes to machine learning. In order to make the model more efficient we need lots of data, so we turned our focus primarily on one of the large scale data producers owned by Facebook which is nothing but WhatsApp. WhatsApp claims that nearly 55 billion messages are sent each day. The average user spends 195 minutes per week on WhatsApp, and is a member of plenty of groups. With this treasure house of data right under our very noses, it is but imperative that we embark on a mission to gain insights on the messages which our phones are forced to bear witness to.

PROBLEM STATEMENT:

WhatsApp-Analyser is a statistical analysis tool for WhatsApp chats. Working on the chat files that can be exported from WhatsApp it generates various plots showing, for example, which another participant a user responds to the most. We propose to employ dataset manipulation techniques to have a better understanding of WhatsApp chat present in our phones.

STUDY PHASE

EXISTING SYSTEM:

There is a lot of development in the current system. In the older version there was no feature to display status, there was no feature to share documents and there was no feature to share location. In the current version, all of these features are available. In older version we couldn't share images through doc's format. In this system user is able to access WhatsApp in windows through WhatsApp web application, which can be connected through QR code. There is another feature called export chat where user can send or share or get the chat detail for data analysis through email, Facebook or some messenger application.

PROPOSED SYSTEM:

Data pre-processing, the initial part of the project is to understand implementation and usage of various python built modules. The above process helps us to understand why different modules are helpful rather than implementing those functions from scratch by the developer. These various modules provide better code representation and user understandability. The following libraries are used such as NumPy, SciPy pandas, csv, sklearn, matplotlib, sys, re, emoji, nltk seaborn etc. Exploratory data analysis, first step in this to apply a sentiment analysis algorithm which provides positives negative and neutral part of the chat and is used to plot pie chart based on these parameters. To plot a line graph which shows author and message count of each date, to plot a line graph which shows author and message count of each author, Ordered graph of date vs message count, media sent by authors and their count, Display the message which is di not have authors, plot graph of hour vs message count

SOFTWARE REQUIREMENTS:

- Operating System : Windows
- Coding Languages : Python
- IDE : Google Collaboratory
- Database : WhatsApp

HARDWARE REQUIREMENTS:

- System :HP
- Hard Disk :1TB
- Ram : 8GB

FUNCTIONAL REQUIREMENTS:

Functional requirements specify which output file should be produced from the given file they describe the relationship between the input and output of the system for each functional requirements a detailed description of all data inputs and their source and the range of valid inputs must be specified.

INTEGRATED DEVELOPMENT ENVIRONMENT SOFTWARE:

GOOGLE COLABORATORY

- Colab is a Python development environment that runs in the browser using Google Cloud4
- Colab is a virtual machine we can access directly

WHATSAPP

- Whatsapp was founded by Brian Acton and jankoum in 2009.
- Whatsapp is one of the most used messenger application with 2 billions users worldwide.
- Daily more than 65 billions of messages are passing through WhatsApp.
- Now we can use WhatsApp to analyse the chat with friend or group of people.

WHATSAPP CHAT ANALYSIS WITH PYTHON USING ANDROID

- Open Whatsapp.
- In Android smartphone then tap on the 3 dots above.
- Hit on more option.
- Tap on export chat.
- Select without media.
- Then email this chat to yourself and download it to your system.

LIBRARIES

- import re
- import regex
- import pandas as pd
- import NumPy as np
- import emojis
- import plotly. express as px
- from collections import Counter
- import matplotlib.pyplot as plt
- from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

LIBRARIES DESCRIPTION

1.import re

- Regex is a Regular Expression .It is a sequence of characters to form a search pattern.
- It is used to check if the string contains the specified search pattern.
- Regular expression are a powerful language for matching string patterns.

findall	Returns a list containing all matches
search	Returns a Match object if there is a match anywhere in the string
split	Returns a list where the string has been split at each match
sub	Replaces one or many matches with a string

2.import pandas as pd

- Pandas makes it simple to do many of the time consuming, repetitive tasks associated with working with data, including:
- Data cleansing
- Data fill
- Data normalization
- Merges and joins
- Data visualization
- Statistical analysis
- Data inspection

- ❖ Loading and saving data

3.import NumPy as np

- ▶ NumPy is a Python package. It stands for Numerical Python.
- ▶ Using NumPy, mathematical and logical operations on arrays can be performed.
- ▶ NumPy provides various powerful data structures, implementing multi-dimensional arrays and matrices.

4.import emojis

- ▶ Everything in memory will store in binary format.it is difficult to identify the emoji.
- Methods
- Unicode
- CLDR names
- Emoji module

Emoji module

- ▶ Emojize() function requires the CLDR short name to be passed in it as the parameter. It then returns the corresponding emoji. Replace the spaces with underscore in the CLDR short name.

5.import plotly. express as px

- ▶ It was a open source plotting library.
- ▶ 40 unique chart types like
  - Statistical
  - Financial
  - Geographical
  - Scientific
  - 3 dimensional charts
- ▶ It was a beautiful interactive web based visualizations.

6.from collections import Counter

- ▶ Counter is a container it will hold the count of each of the elements present in container.
- ▶ The counter holds the data in a unordered collection .
- ▶ The elements are keys and the count are values.
- ▶ Arithmetic operations are used to perform on a counter.
- ▶ A counter can also count elements from another counter.

7.import matplotlib.pyplot as plt

- ▶ It is a collection of comment style functions.
- ▶ Each pyplot function makes some changes in figure.
  - Creates a figure
  - Creates a plotting area in a figure
  - Plots some lines in a plotting area
  - Decorates the plot labels

8.from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

- ▶ WORDCLOUD
  - It is also called Tag clouds
  - It used for annotating texts and especially websites
  - It will find the most important words or classifying a text
- ▶ STOPWORDS
  - It is used to count the no. of occurrences in the important words
  - But it leads very interesting results
- ▶ IMAGECOLORGENERATOR
  - We can color a word by using an image-based coloring strategy

SOURCE CODE

EMOJI INSTALLATION

```
!pip install emojis
```

IMPORTINGLIBRARIES

```
import re
import regex
import pandas as pd
import numpy as np
import emojis
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
% matplotlib inline
```

SOURCE CODE FOR DATETIME AND AUTHOR AND MESSAGE

```
def date_time(s):
    pattern = '^([0-9]+)(\/)([0-9]+)(\/)([0-9]+), ([0-9]+):([0-9]+)[ ]?(AM|PM|am|pm)? -'
    result = regex.match(pattern, s)
    if result:
        return True
    return False

def find_author(s):
    s = s.split(":")
    if len(s)==2:
        return True
    else:
        return False
def getDatapoint(line):
    splitline = line.split(' - ')
    dateTime = splitline[0]
    date, time = dateTime.split(", ")
    message = " ".join(splitline[1:])
    if find_author(message):
        splitmessage = message.split(": ")
```

```
      author = splitmessage[0]
        message = " ".join(splitmessage[1:])
    else:
        author= None
    return date, time, author, message
```

SOURCECODE FOR WHATSAPP CHAT ANALYSIS

```
data = []
conversation = 'vaishu.txt'
with open(conversation, encoding="utf-8") as fp:
    fp.readline()
    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:
                data.append([date, time, author, ' '.join(messageBuffer)])
            messageBuffer.clear()
            date, time, author, message = getDatapoint(line)
            messageBuffer.append(message)
        else:
            messageBuffer.append(line)
```

PANDAS DATAFRAMES

```
df = pd.DataFrame(data, columns=["Date", 'Time', 'Author', 'Message'])
df['Date'] = pd.to_datetime(df['Date'])
```

OUTPUT

```
print(df.head(31600))
```

EMOJIS

```
def split_count(text):
  emoji_list=[]
  data=emojis.get(str(text))
  return data
  df["Emoji"]=df["Message"].apply(split_count)

message_df["Date"] = pd.to_datetime(message_df.Date)

message_df["Time"] = pd.to_datetime(message_df.Time).dt.strftime('%H:%M')
```

DATAFRAMES FOR ALL ATTRIBUTES

```
message_df.head(5)
```

TOTAL NO OF EMOJIS SENT

```
total_emojis_list=list(set([a for b in message_df.Emoji for a in b]))

total_emojis = len(total_emojis_list)
print(total_emojis)
```

VISULIZATION OF EMOJIS

```
total_emojis_list = list(set([a for b in message_df.Emoji for a in b]))
total_emojis = len(total_emojis_list)

total_emojis_list = list([a for b in message_df.Emoji for a in b])
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)

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

TOTAL MESSAGES IN THE CHAT

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

TOTAL MEDIA MESSAGES

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

TOTAL LINKS,MESSAGES,MEDIA

```
URLPATTERN = r'(https?://\S+)'
df['urlcount'] = df.Message.apply(lambda x: regex.findall(URLPATTERN, x)).str.len()
links = np.sum(df.urlcount)

print("Chats between vaishu and sowmya")
print("Total Messages: ", total_messages)
print("Number of Media Shared: ", media_messages)
print("Number of emojis sent:",total_emojis)
print("Number of Links Shared", links)

media_messages_df = df[df['Message'] == '<Media omitted>']

messages_df = df.drop(media_messages_df.index)
```

LETTER,WORD,MESSAGE COUNT

```
messages_df['Letter_Count'] = messages_df['Message'].apply(lambda s : len(s))
messages_df['Word_Count'] = messages_df['Message'].apply(lambda s : len(s.split(' ')))
messages_df["MessageCount"]=1
```

PANDAS DATAFRAMES FOR ALL ATTRIBUTES

```
messages_df.head(31600)
```

ATTRIBUTES

```
messages_df.info()
```

CHAT ANALAYSIS IN WEEKDAYS

```
def f(i):
  l = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]
  return l[i];
day_df=pd.DataFrame(messages_df["Message"])
day_df['day_of_date'] = messages_df['Date'].dt.weekday
day_df['day_of_date'] = day_df["day_of_date"].apply(f)
day_df["messagecount"] = 1
day = day_df.groupby("day_of_date").sum()
day.reset_index(inplace=True)
print(day_df)
```

VISUALIZATION OF WEEKDAYS

```
fig = px.line_polar(day, r='messagecount', theta='day_of_date', line_close=True)
fig.update_traces(fill='toself')
fig.update_layout(
  polar=dict(
    radialaxis=dict(
      visible=True,
    )),
  showlegend=False
)
fig.show()
```

VISULAIZATION OF MESSAGES

```
date_df = messages_df.groupby("Date").sum()
date_df.reset_index(inplace=True)
fig = px.line(date_df, x="Date", y="MessageCount")
fig.update_xaxes(nticks=20)
fig.show()
```

MOST ACTIVE PERSON IN THE CHAT

```
 plt.figure(figsize=(9,6))
mostly_active = df['Author'].value_counts()
m_a = mostly_active.head(2)
bars = ['vaishu','sowmya']
x_pos = np.arange(len(bars))
m_a.plot.bar()
plt.xlabel('Authors',fontdict={'fontsize': 14,'fontweight': 10})
plt.ylabel('No. of messages',fontdict={'fontsize': 14,'fontweight': 10})
plt.title('Mostly active member of chat',fontdict={'fontsize': 20,'fontweight': 8})
plt.xticks(x_pos, bars)
plt.show()
```

WORDS USED SO MANY TIMES

```
 text = " ".join(review for review in messages_df.Message)
print ("There are {} words in all the messages.".format(len(text)))
stopwords = set(STOPWORDS)
# Generate a word cloud image
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(te
xt)
# Display the generated image:
# the matplotlib way:
plt.figure( figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

VISUALIZATION OF ACTIVE AUTHOR

```
auth = messages_df.groupby("Author").sum()
auth.reset_index(inplace=True)
fig = px.bar(auth, y="Author", x="MessageCount", color='Author', orientation="h",
             color_discrete_sequence=["red", "green"],
             title="Explicit color sequence"
            )

fig.show()
```

AUTHOR INFORMATION

```
media_messages_df = df[df['Message'] == '<Media omitted>']
messages_df = df.drop(media_messages_df.index)
messages_df['Letter_Count'] = messages_df['Message'].apply(lambda s : len(s))
messages_df['Word_Count'] = messages_df['Message'].apply(lambda s : len(s.split(' ')))
messages_df["MessageCount"]=1

l = ["Vaishu", "Sowmya"]
for i in range(len(l)):
  # Filtering out messages of particular user
  req_df= messages_df[messages_df["Author"] == l[i]]
  # req_df will contain messages of only one particular user
  print(f'Stats of {l[i]} -')
  # shape will print number of rows which indirectly means the number of messages
  print('Messages Sent', req_df.shape[0])
  #Word_Count contains of total words in one message. Sum of all words/ Total Message
s will yield words per message
  words_per_message = (np.sum(req_df['Word_Count']))/req_df.shape[0]
  print('Average Words per message', words_per_message)
  #media conists of media messages
  media = media_messages_df[media_messages_df['Author'] == l[i]].shape[0]
  print('Media Messages Sent', media)
  # emojis conists of total emojis
  total_emojis_list=list(set([a for b in message_df.Emoji for a in b]))
  total_emojis = len(total_emojis_list)
  print('total emojis Sent:',total_emojis)
  #links consist of total links
  links = sum(req_df["urlcount"])
  print('Links Sent', links)
```

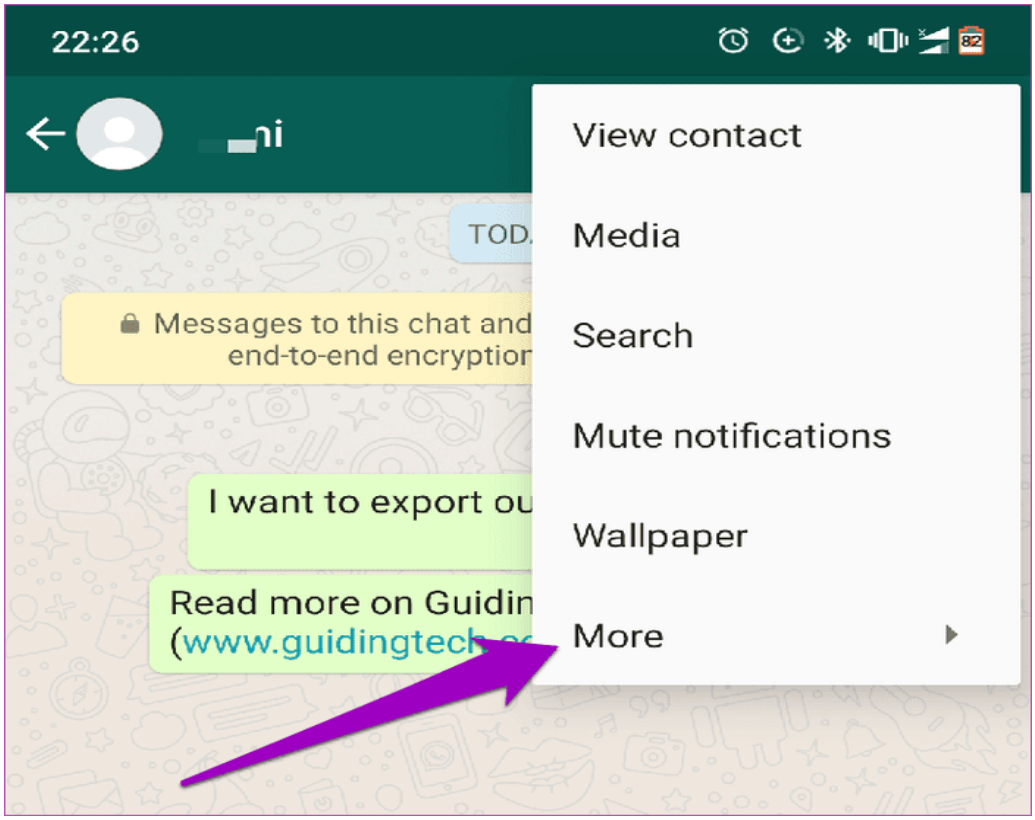
CONVERTING THE DATA TO EXCEL FILE:

```
df.to_csv('chat.csv')
```

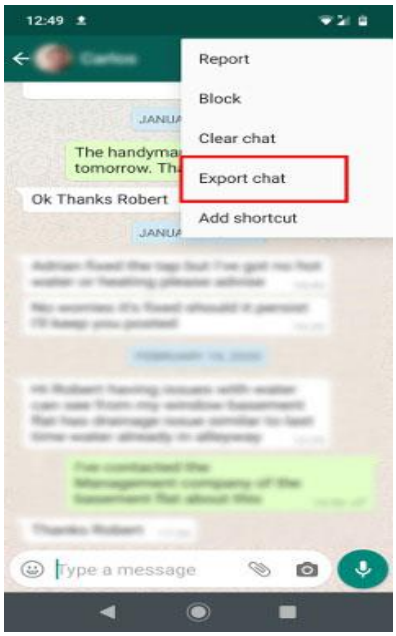
SNAPSHOTS

- **Open the WhatsApp**

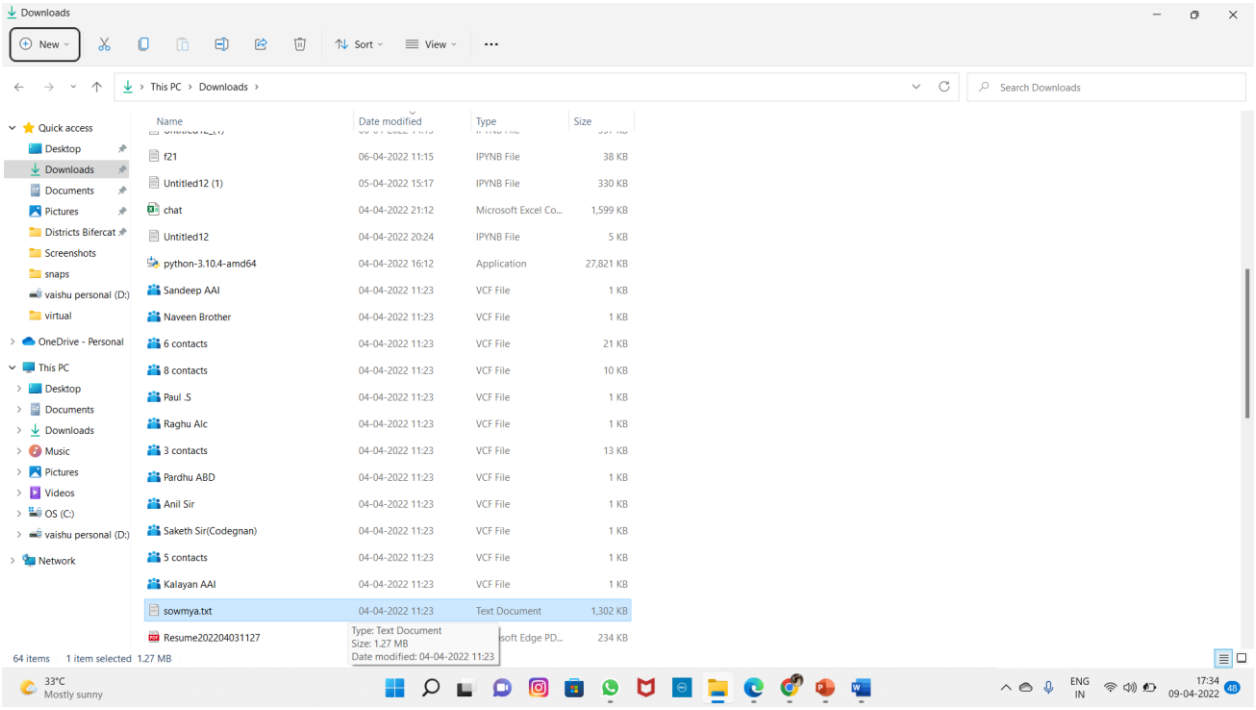
- **Select any chart and click the above 3 dots and hit more option.**



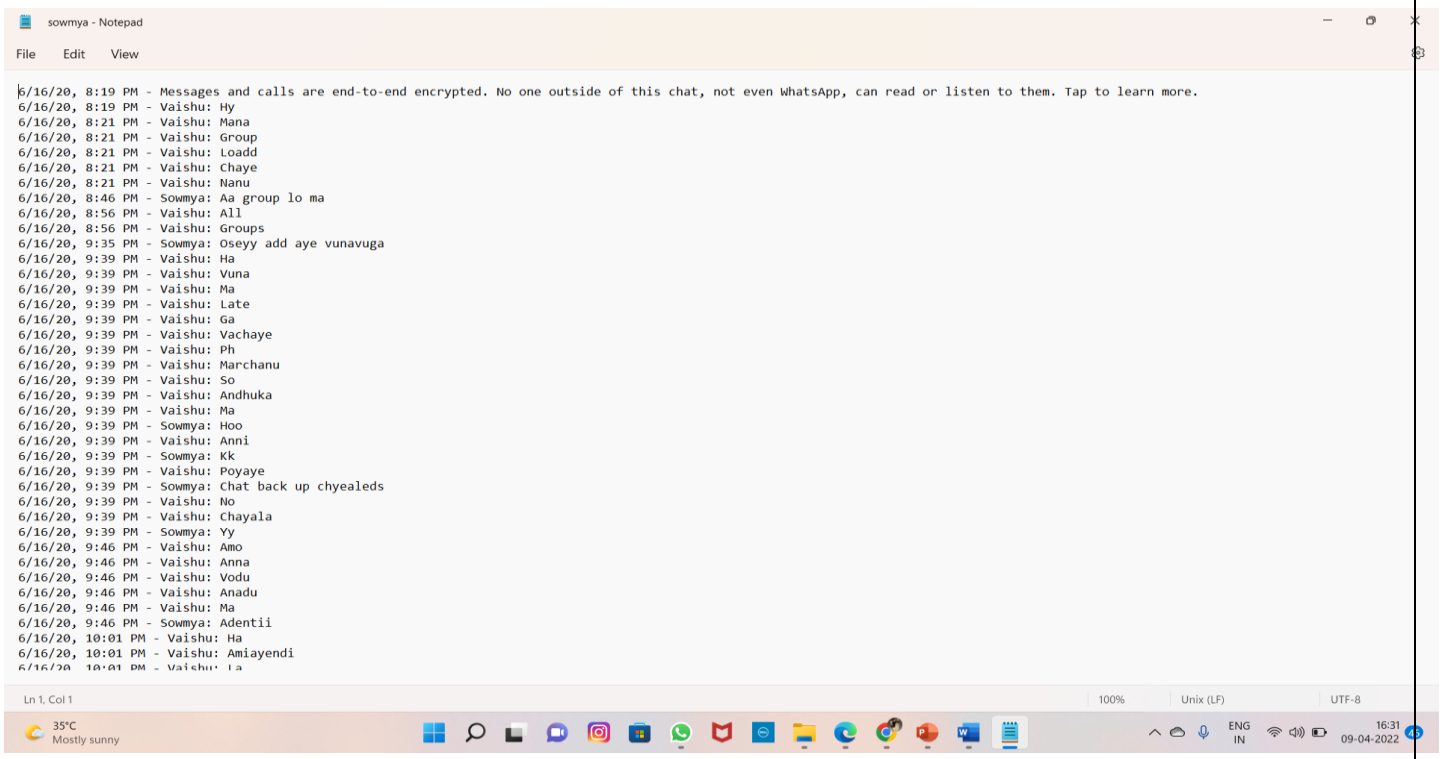
➢ Select the Export chat.



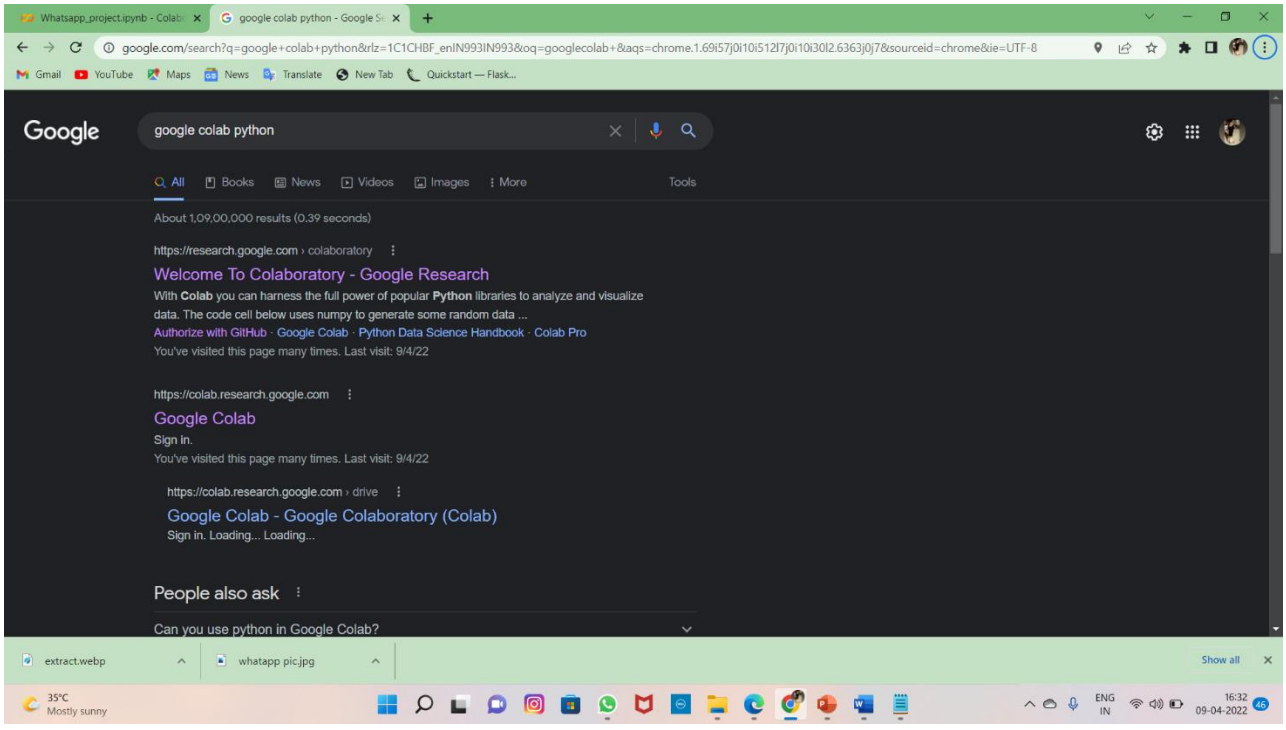
➢ Download the chat in your system.



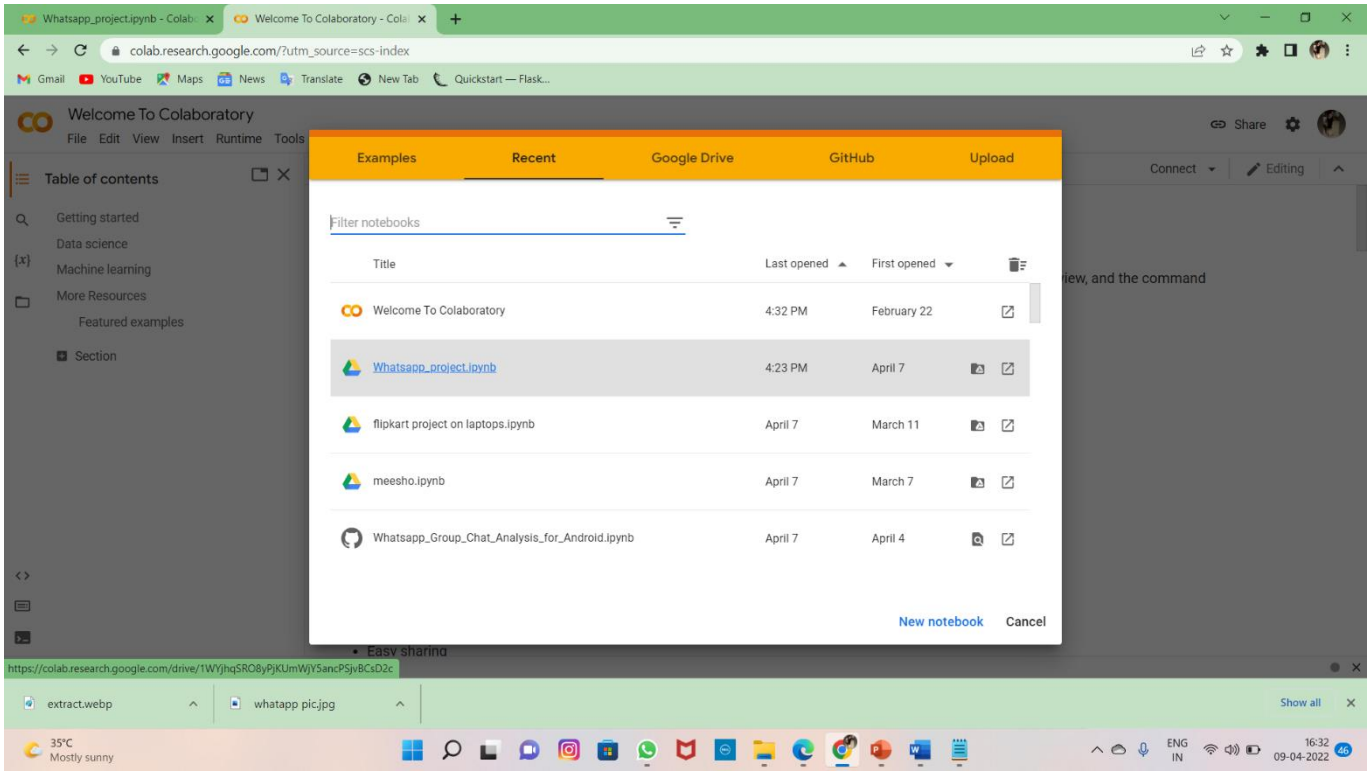
➢ Here is a text file of the chat



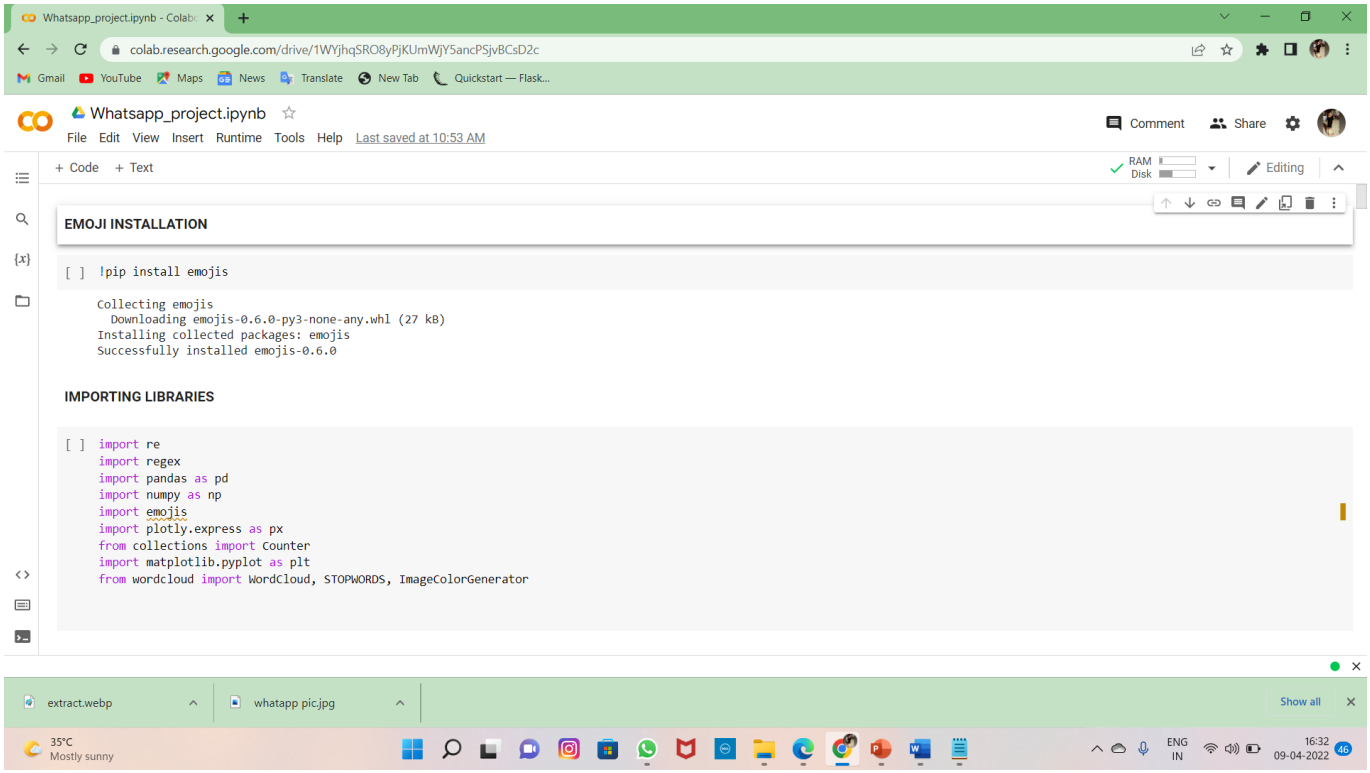
➢ Open Google Colaboratory



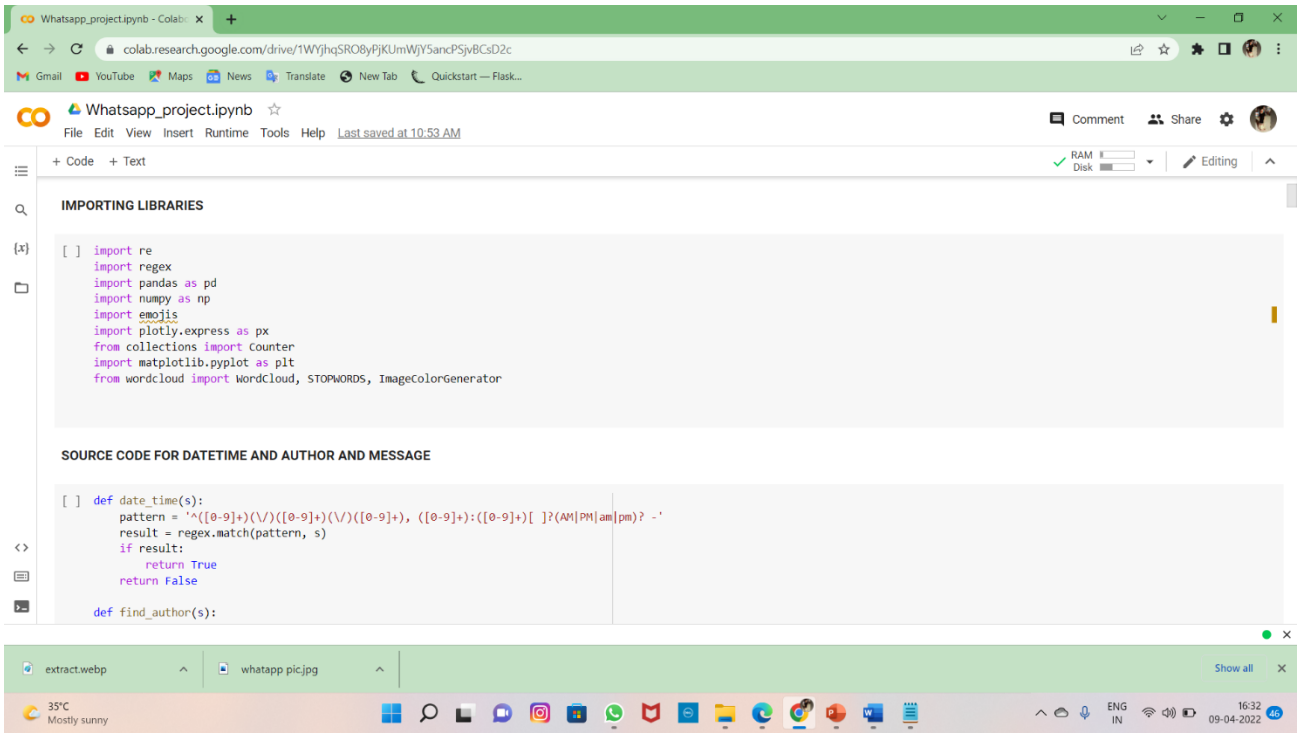
➢ Select the notebook



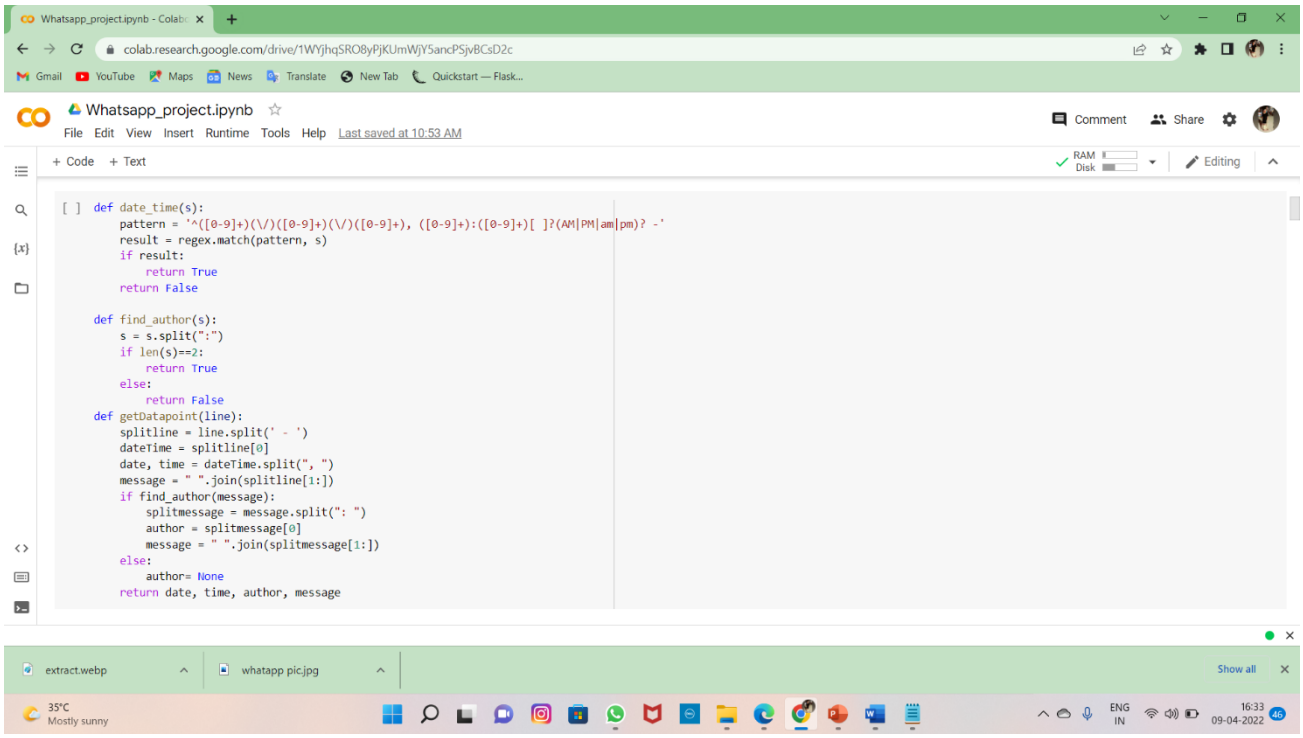
➢ Install the emoji Library



➢ Import the libraries

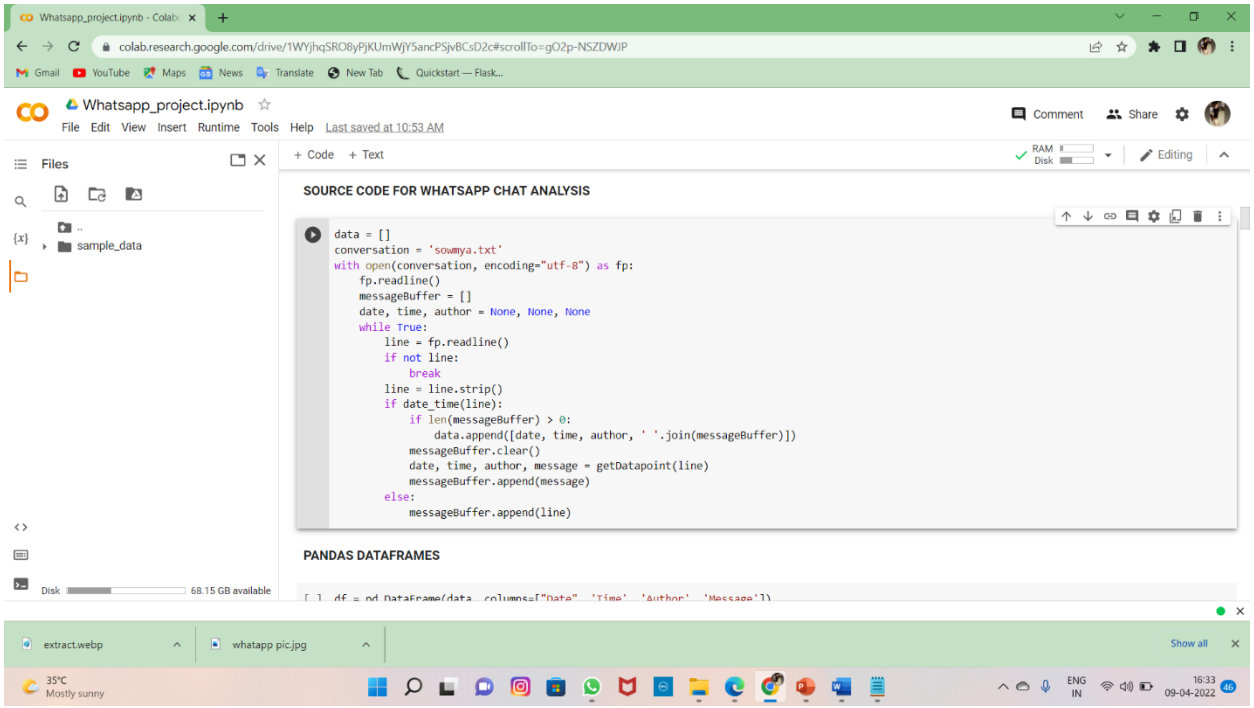


➢ Source code for Date,Time and Author



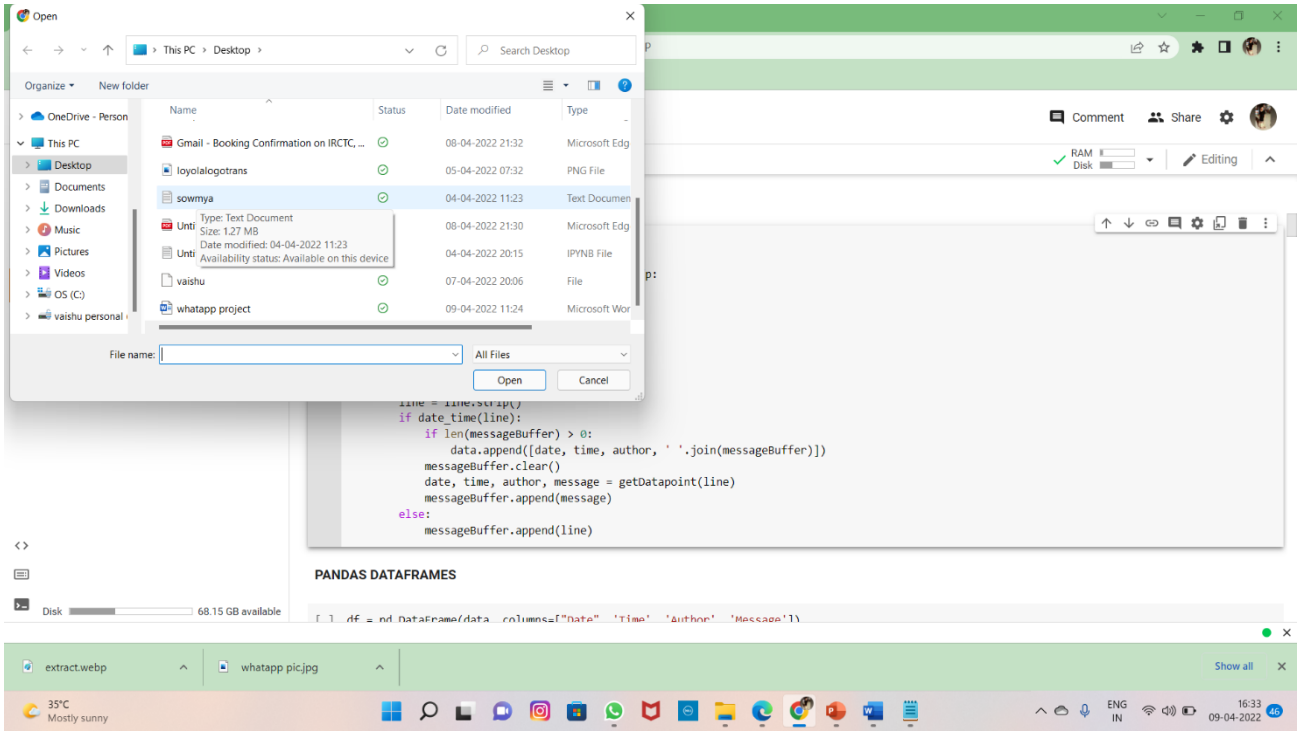
```
def date_time(s):
    pattern = '^([0-9]+)(\/)([0-9]+)(\/)([0-9]+), ([0-9]+):([0-9]+)[ ]?(AM|PM|am|pm)? -'
    result = regex.match(pattern, s)
    if result:
        return True
    return False

def find_author(s):
    s = s.split(":")
    if len(s)==2:
        return True
    else:
        return False
def getDatapoint(line):
    splitline = line.split(' - ')
    dateTime = splitline[0]
    date, time = dateTime.split(", ")
    message = " ".join(splitline[1:])
    if find_author(message):
        splitmessage = message.split(": ")
        author = splitmessage[0]
        message = " ".join(splitmessage[1:])
    else:
        author= None
    return date, time, author, message
```

➢ Now click the file option at left side

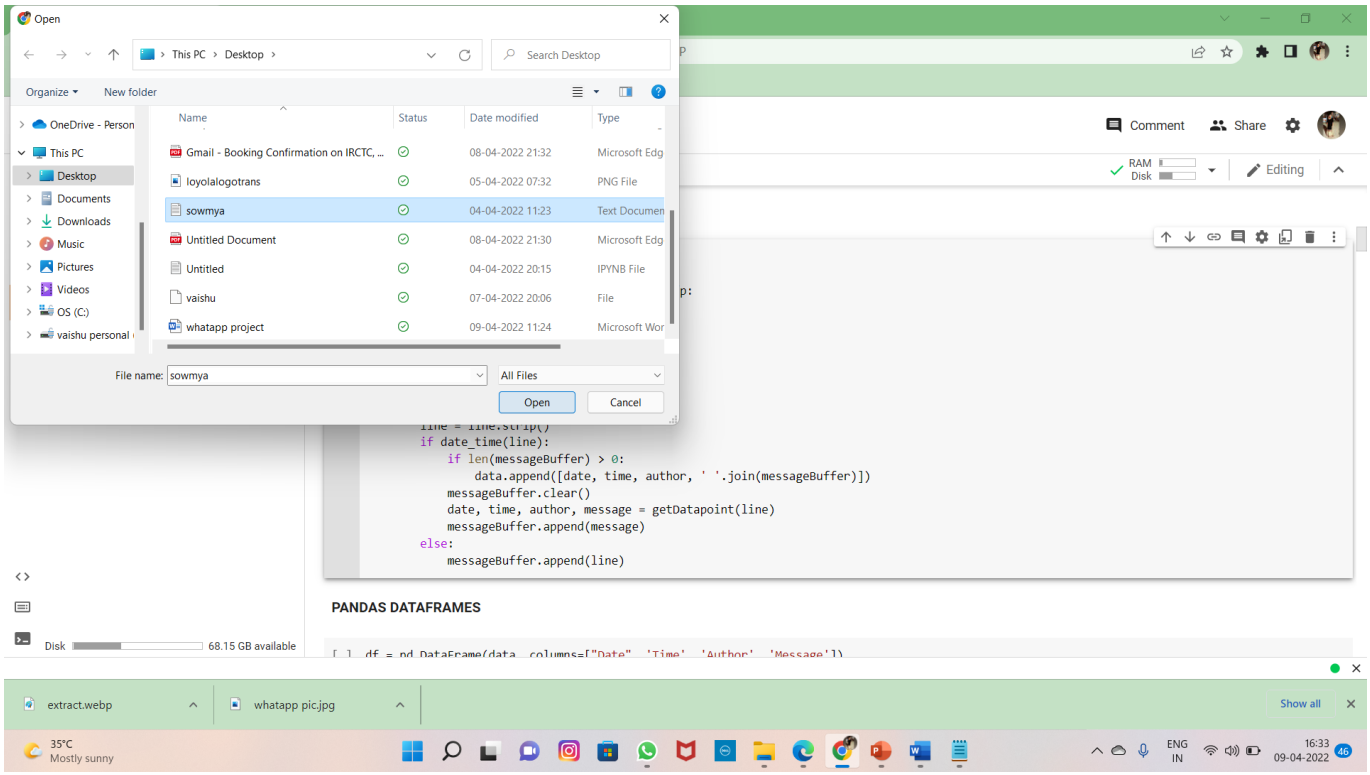


```
data = []
conversation = 'sowmya.txt'
with open(conversation, encoding="utf-8") as fp:
    fp.readline()
    messageBuffer = []
    date, time, author = None, None, None
    while True:
        line = fp.readline()
        if not line:
            break
        line = line.strip()
        if date_time(line):
            if len(messageBuffer) > 0:
                data.append([date, time, author, ' '.join(messageBuffer)])
            messageBuffer.clear()
            date, time, author, message = getDatapoint(line)
            messageBuffer.append(message)
        else:
            messageBuffer.append(line)
```

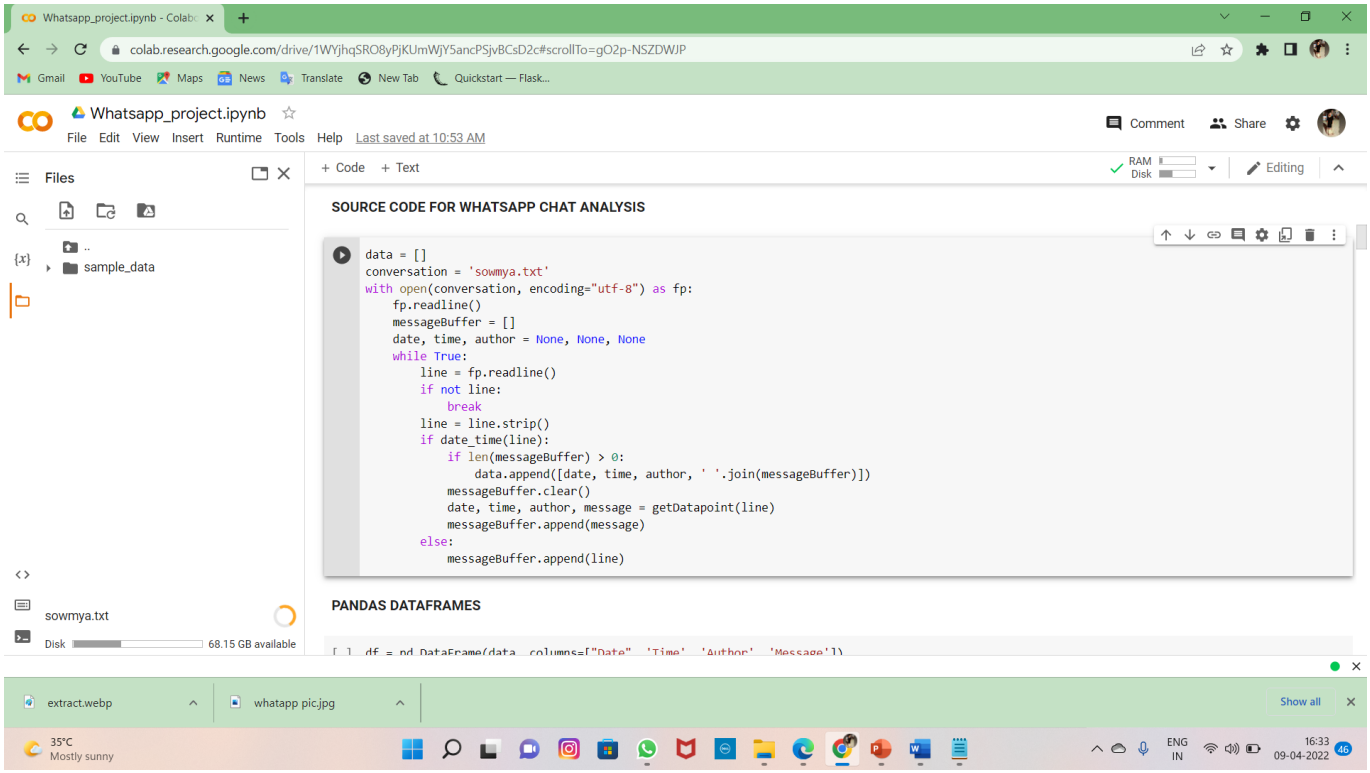
➢ Select the text file that we are downloaded .



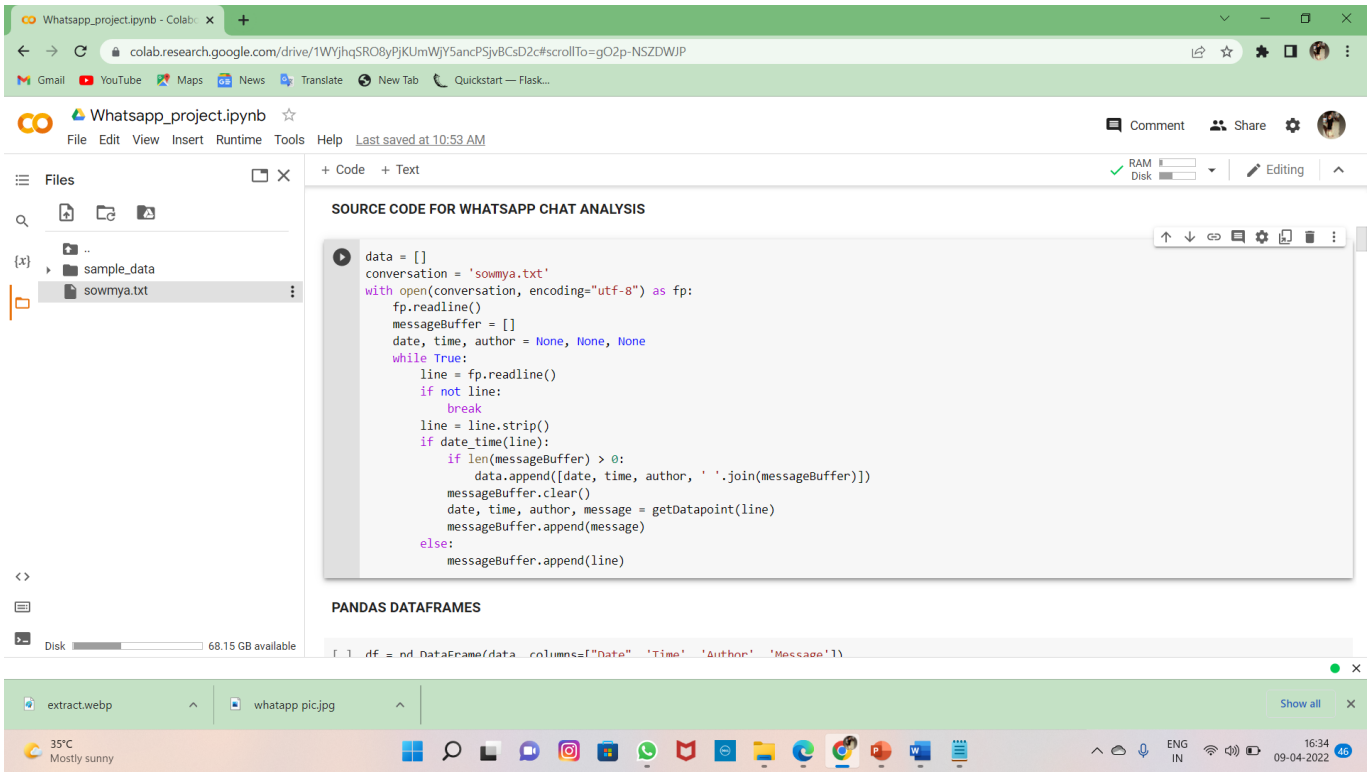
➢ **Hit on Open**



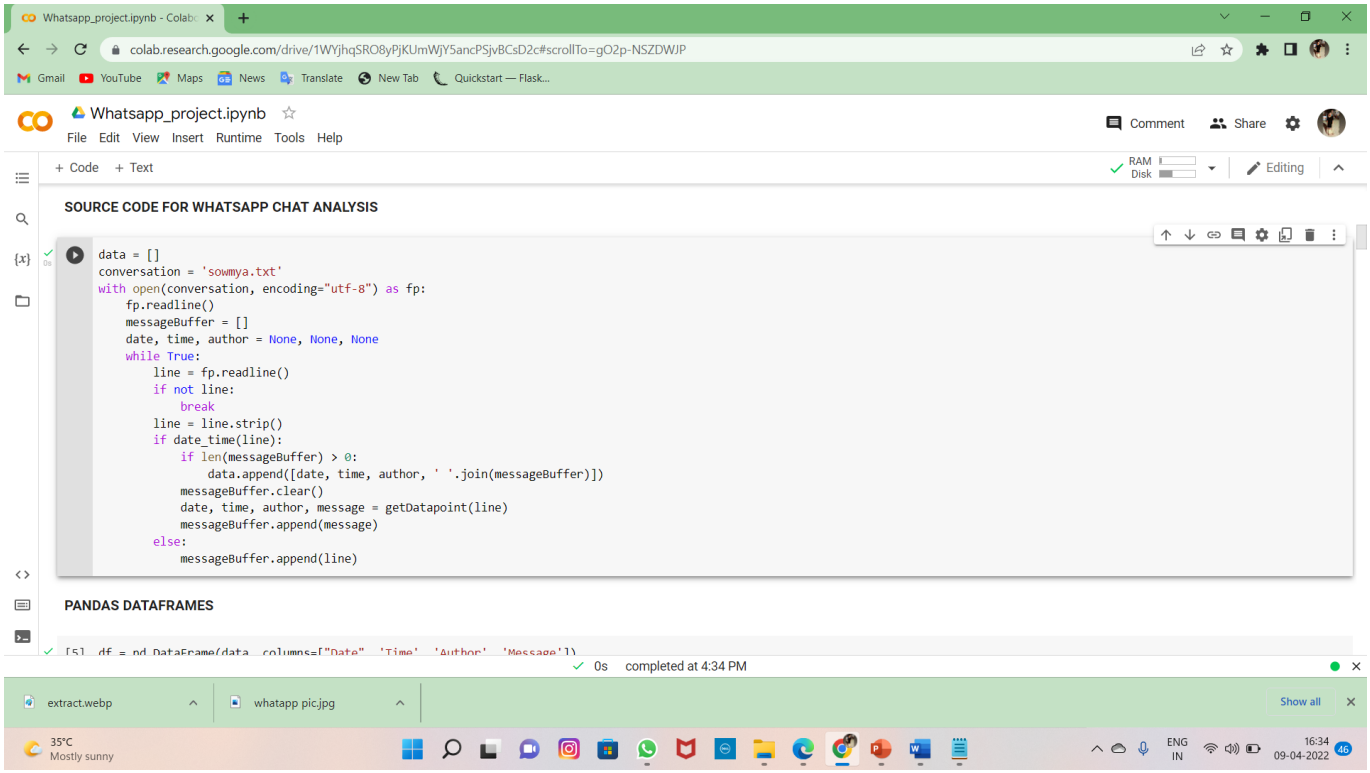
➢ **Now the file is Downloading at the bottom of the screen.**



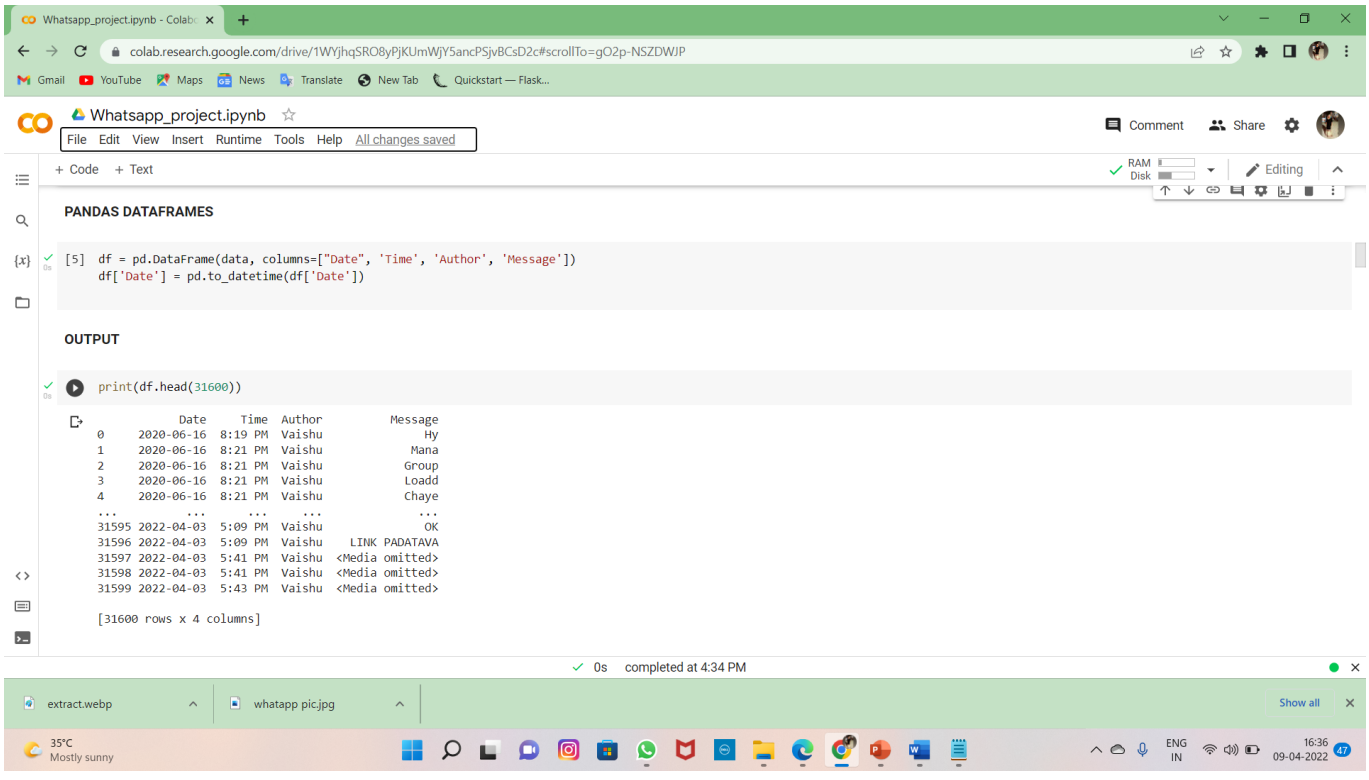
➢ **Now the file is uploaded**



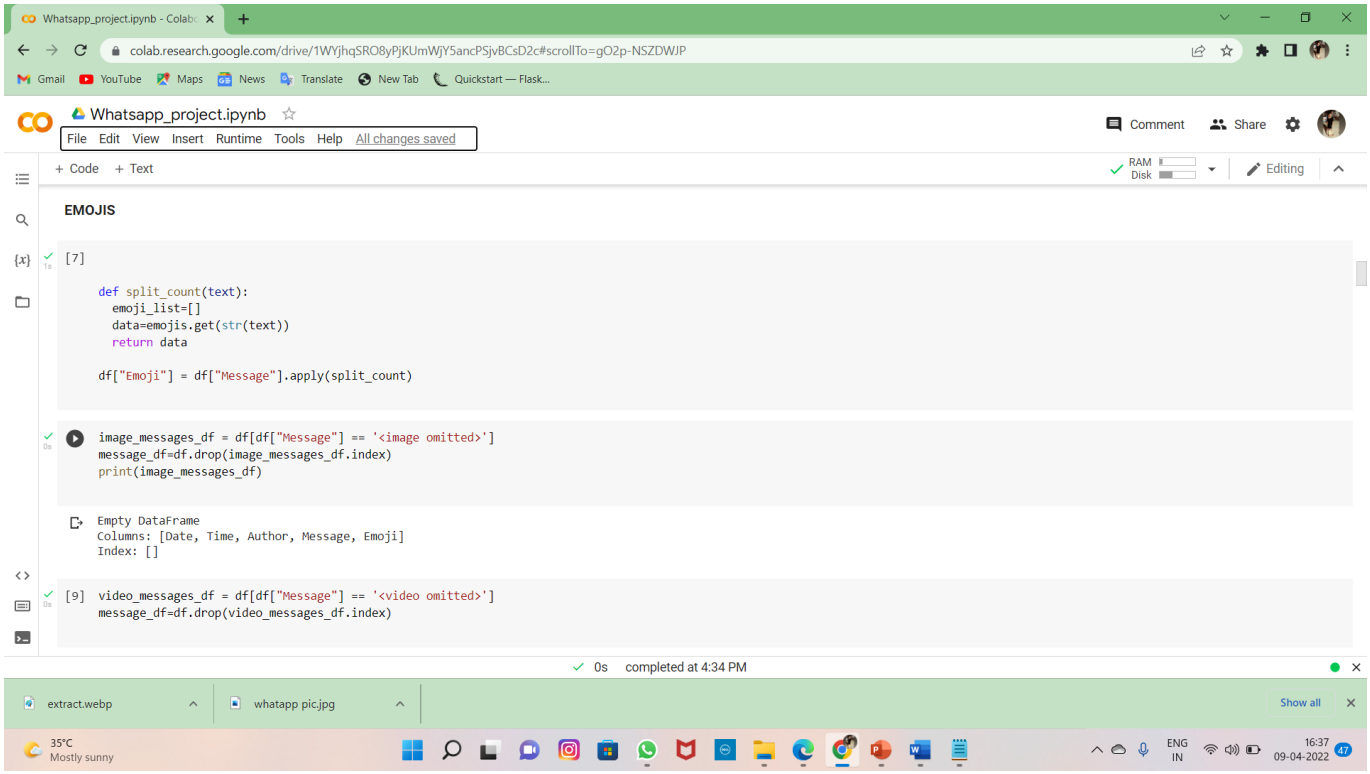
➢ **Source code for file opening and analysing the data**



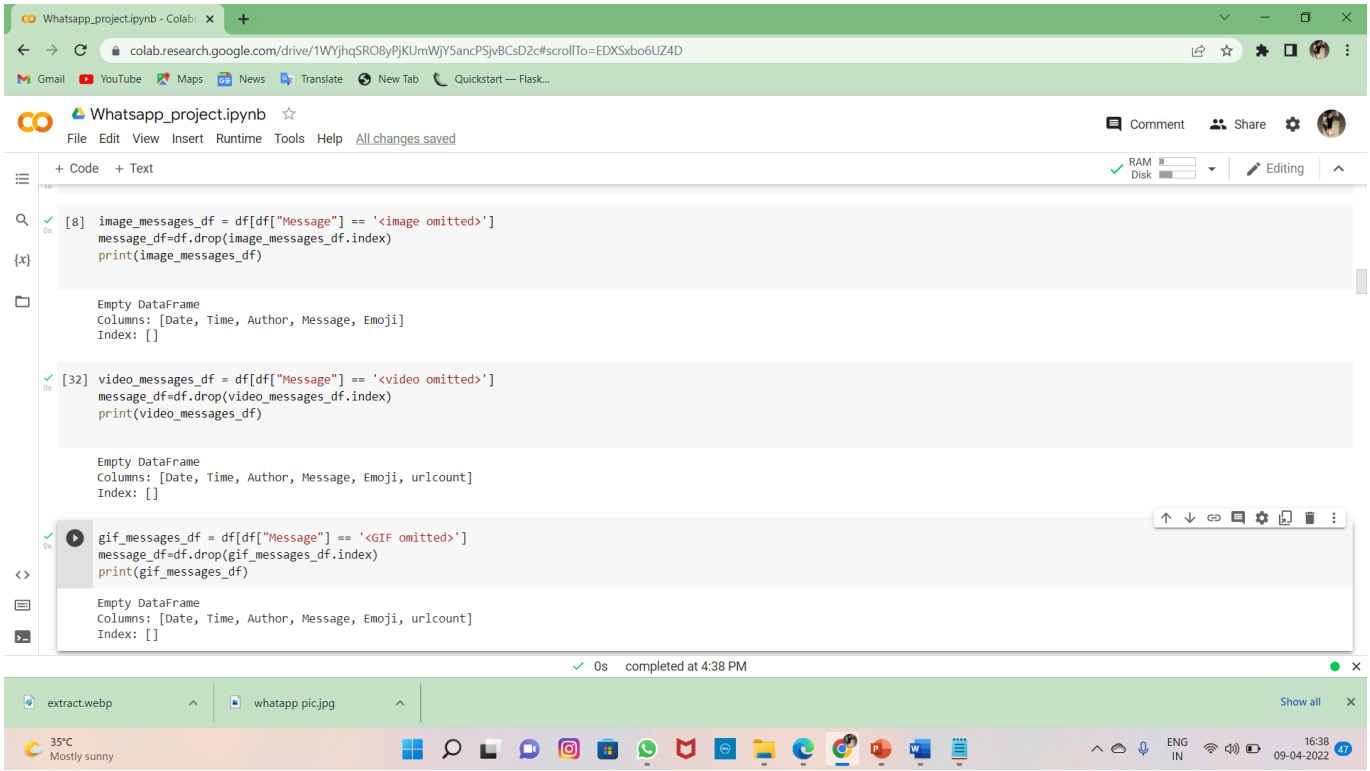
➢ Pandas DataFrames with Output



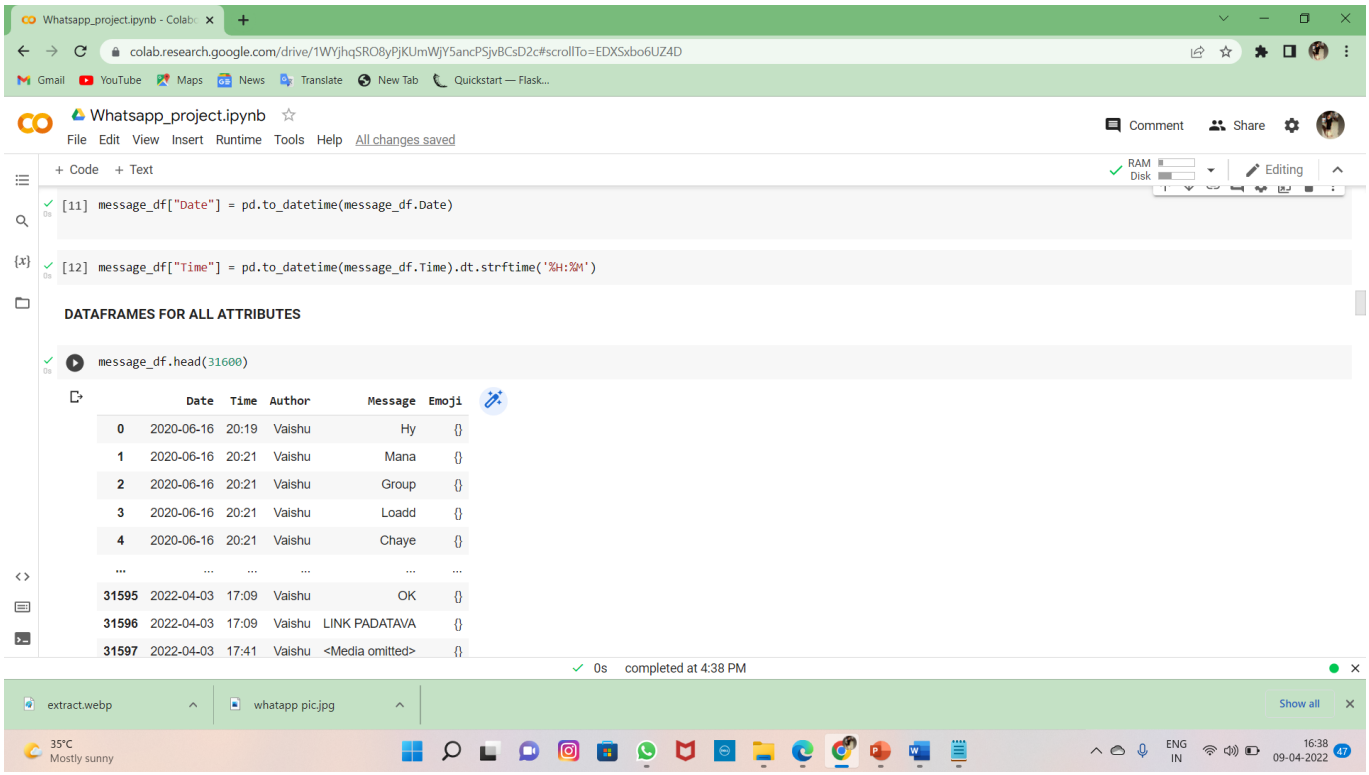
➢ Analysing the Emojis



➢ Code to show that the media and vedio and gift was in the file or not



➢ DataFrame added with emojis



```
[11] message_df["Date"] = pd.to_datetime(message_df.Date)

[12] message_df["Time"] = pd.to_datetime(message_df.Time).dt.strftime('%H:%M')
```

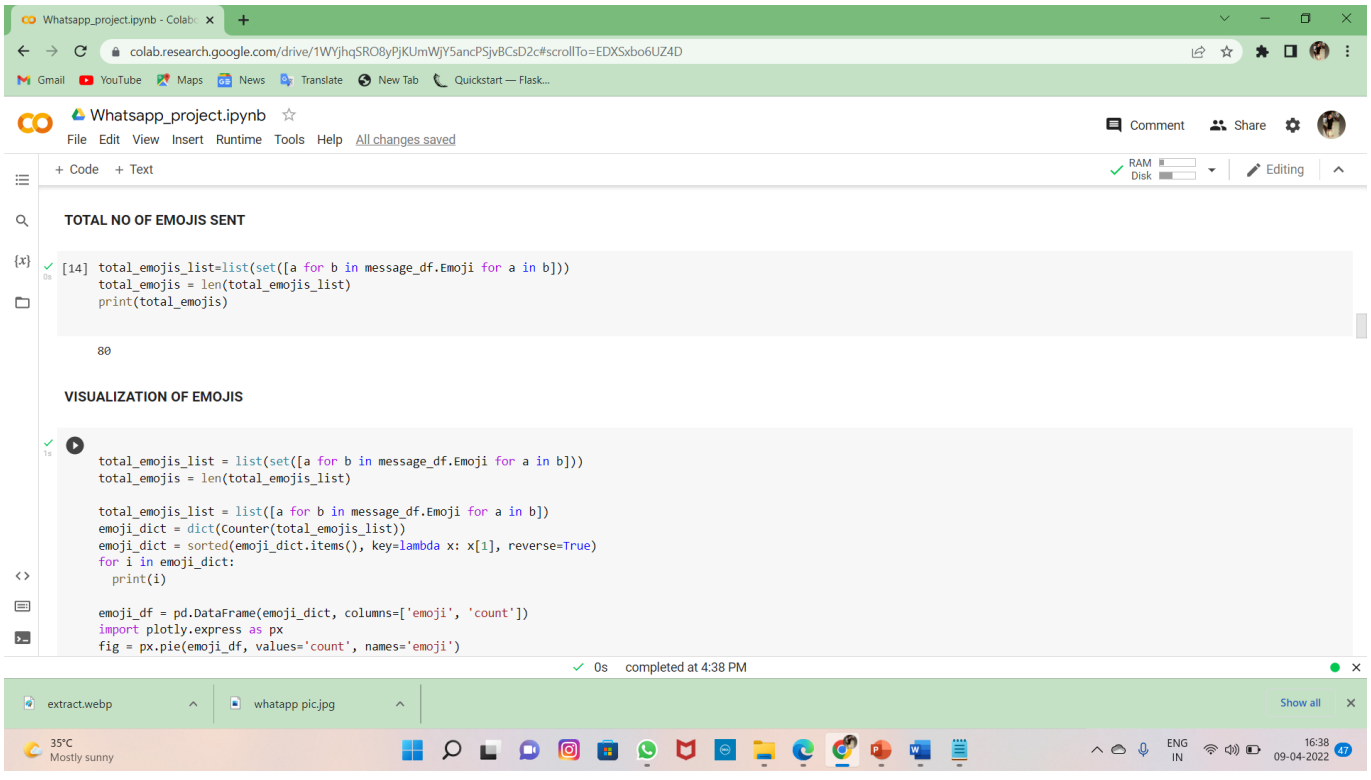
DATAFRAMES FOR ALL ATTRIBUTES

```
message_df.head(31600)
```

	Date	Time	Author	Message	Emoji
0	2020-06-16	20:19	Vaishu	Hy	{}
1	2020-06-16	20:21	Vaishu	Mana	{}
2	2020-06-16	20:21	Vaishu	Group	{}
3	2020-06-16	20:21	Vaishu	Loadd	{}
4	2020-06-16	20:21	Vaishu	Chaye	{}
...	...	...	...	...	...
31595	2022-04-03	17:09	Vaishu	OK	{}
31596	2022-04-03	17:09	Vaishu	LINK PADATAVA	{}
31597	2022-04-03	17:41	Vaishu	<Media omitted>	{}

0s completed at 4:38 PM

➢ Total no of emojis present in chat



TOTAL NO OF EMOJIS SENT

```
[14] total_emojis_list=list(set([a for b in message_df.Emoji for a in b]))
     total_emojis = len(total_emojis_list)
     print(total_emojis)
```

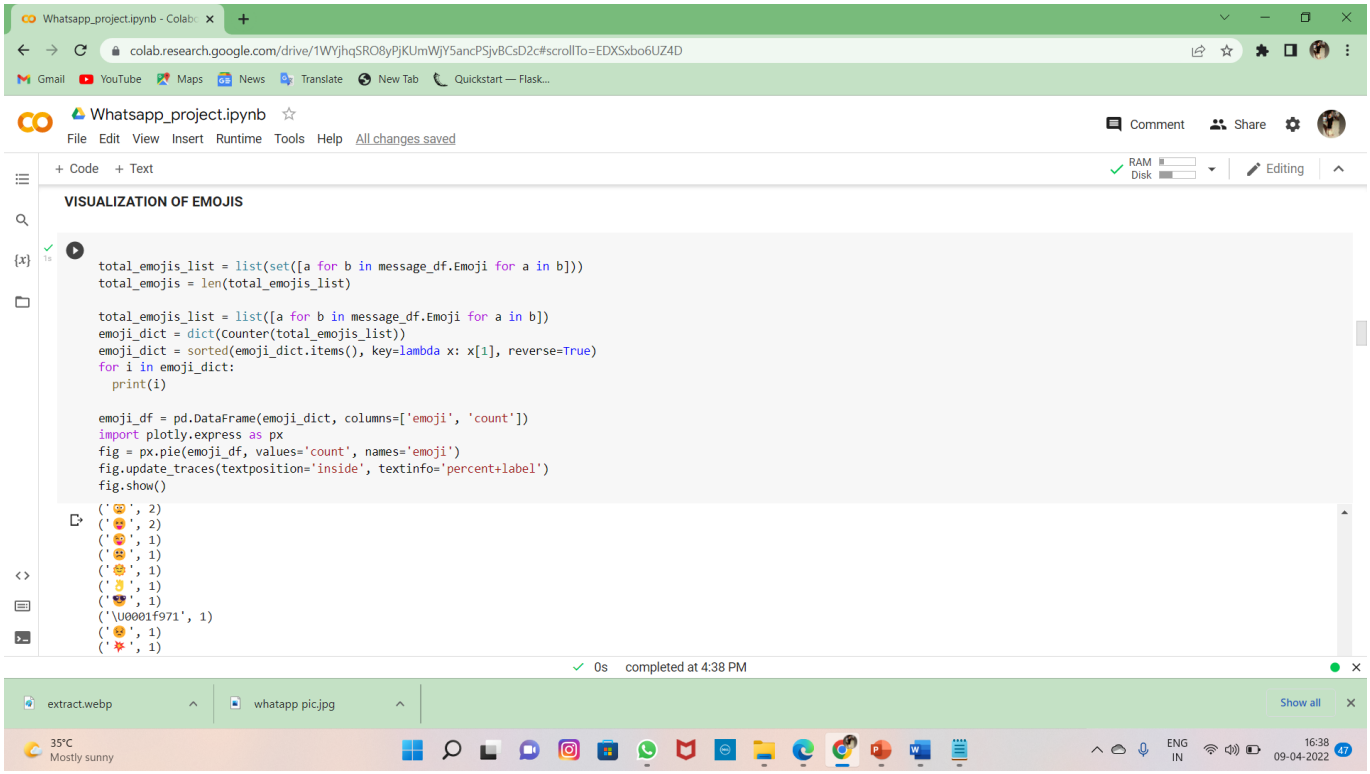
```
80
```

VISUALIZATION OF EMOJIS

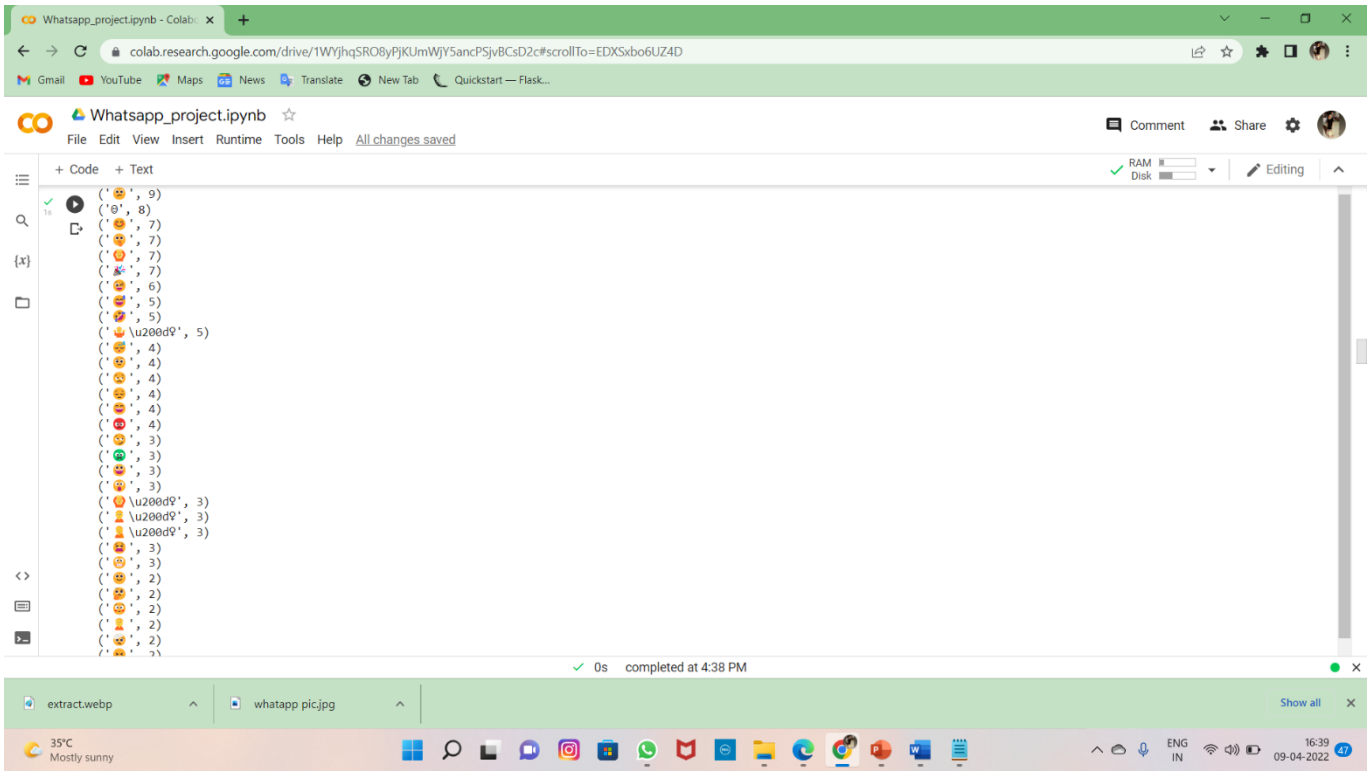
```
total_emojis_list = list(set([a for b in message_df.Emoji for a in b]))
total_emojis = len(total_emojis_list)

total_emojis_list = list([a for b in message_df.Emoji for a in b])
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)

emoji_df = pd.DataFrame(emoji_dict, columns=['emoji', 'count'])
import plotly.express as px
fig = px.pie(emoji_df, values='count', names='emoji')
```

0s completed at 4:38 PM

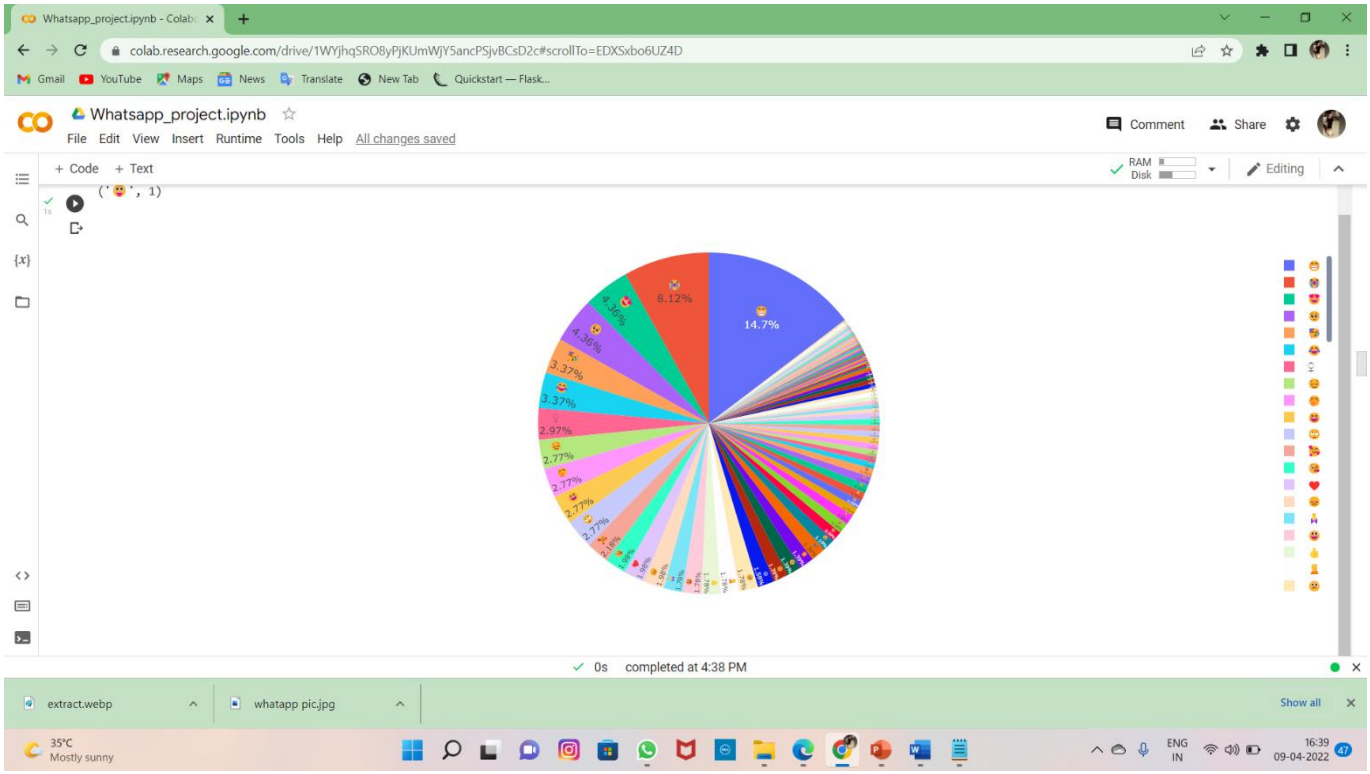
➢ Visualization of emojis



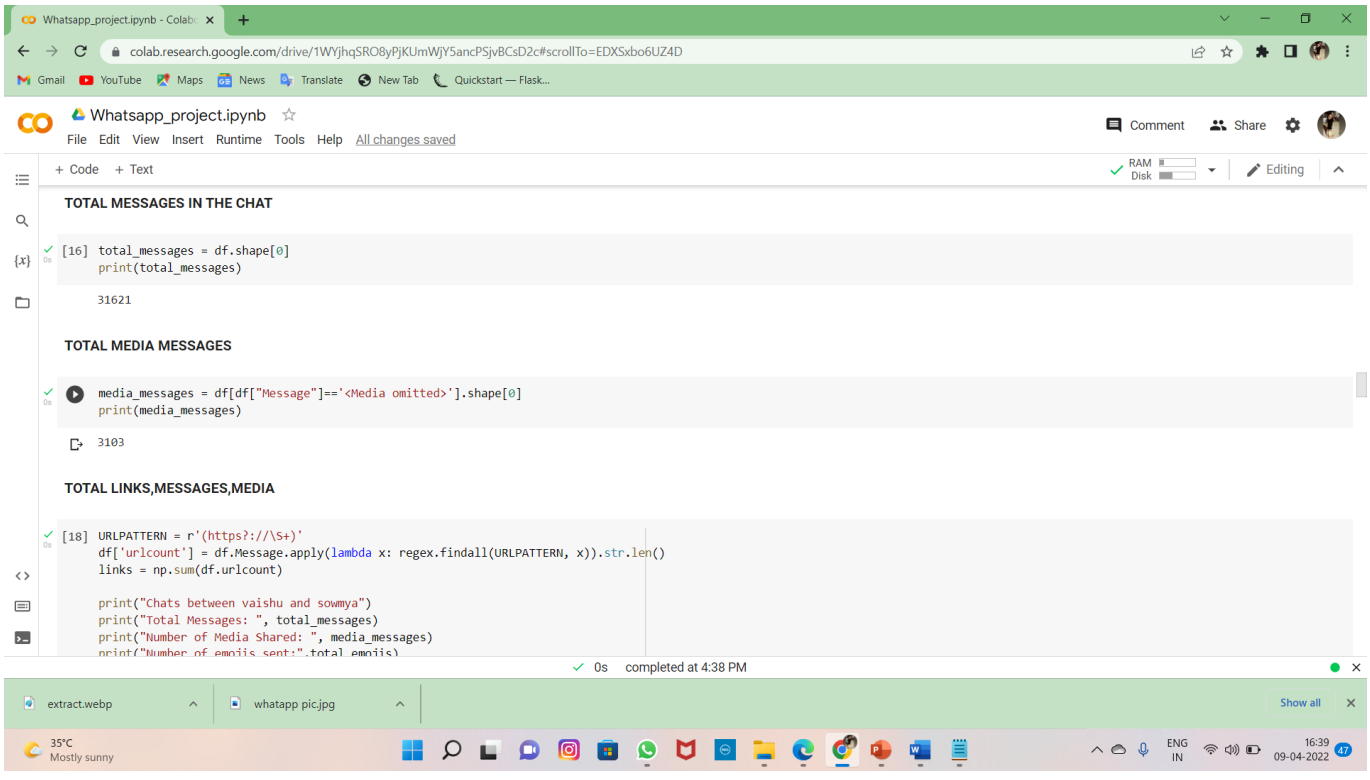
➢ No of emojis sent



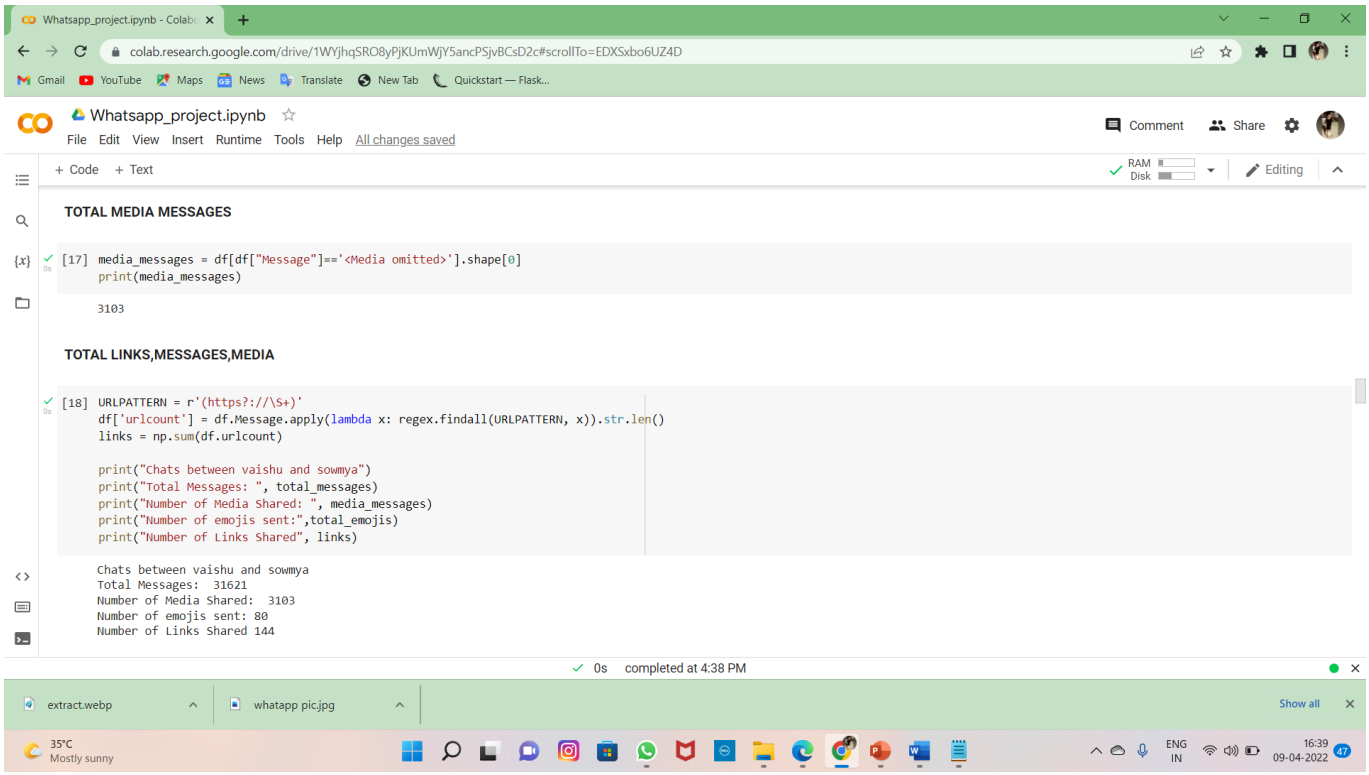
➢ Emojis visualization in pie chart



➢ Total messages and media messages in the chat



➢ Total no of links in the chat



TOTAL MEDIA MESSAGES

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

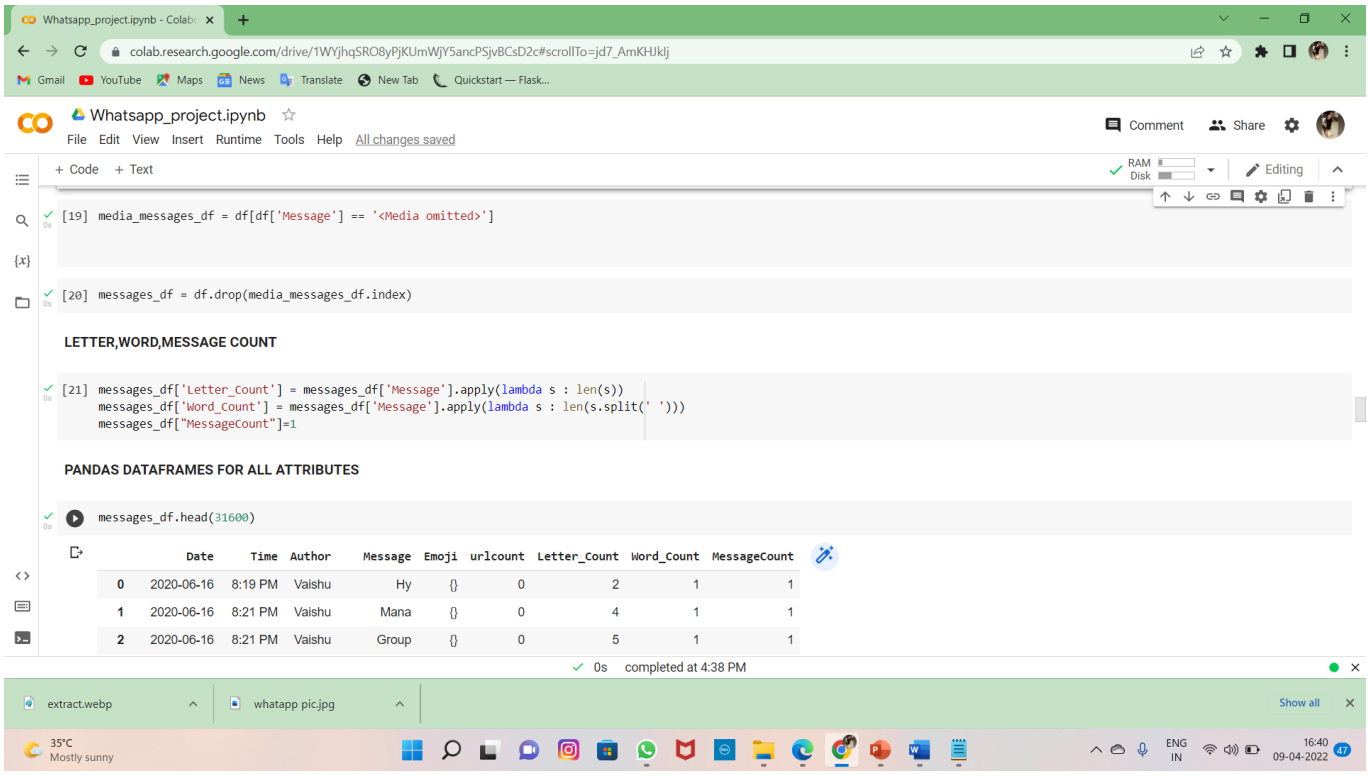
```
3103
```

TOTAL LINKS,MESSAGES,MEDIA

```
[18] URLPATTERN = r'(https?://\S+)'
     df['urlcount'] = df.Message.apply(lambda x: regex.findall(URLPATTERN, x)).str.len()
     links = np.sum(df.urlcount)

     print("Chats between vaishu and sowmya")
     print("Total Messages: ", total_messages)
     print("Number of Media Shared: ", media_messages)
     print("Number of emojis sent:",total_emojis)
     print("Number of Links Shared", links)
```

```
Chats between vaishu and sowmya
Total Messages:  31621
Number of Media Shared:  3103
Number of emojis sent: 80
Number of Links Shared 144
```

➢ Letter ,word and message count analyzation



LETTER,WORD,MESSAGE COUNT

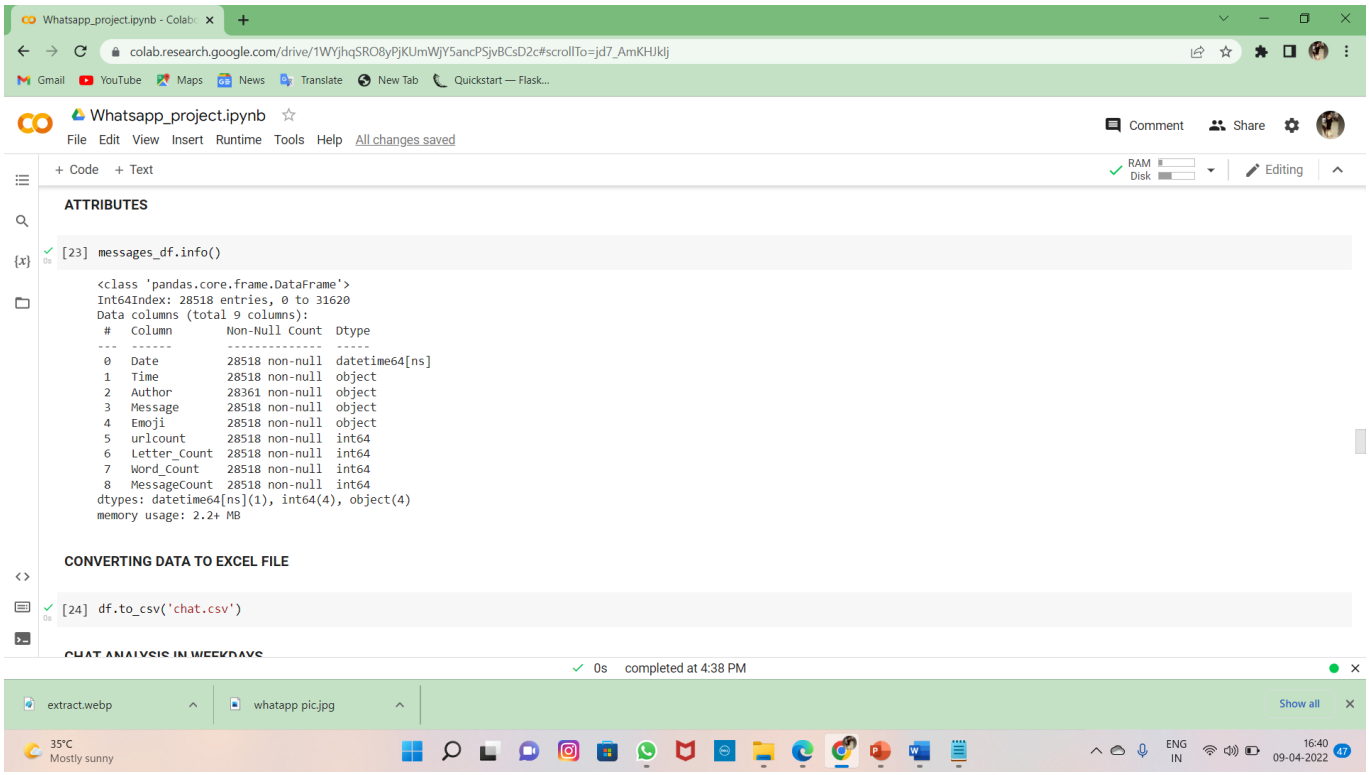
```
[21] messages_df['Letter_Count'] = messages_df['Message'].apply(lambda s : len(s))
     messages_df['Word_Count'] = messages_df['Message'].apply(lambda s : len(s.split(' ')))
     messages_df["MessageCount"]=1
```

PANDAS DATAFRAMES FOR ALL ATTRIBUTES

```
messages_df.head(31600)
```

	Date	Time	Author	Message	Emoji	urlcount	Letter_Count	Word_Count	MessageCount
0	2020-06-16	8:19 PM	Vaishu	Hy	{}	0	2	1	1
1	2020-06-16	8:21 PM	Vaishu	Mana	{}	0	4	1	1
2	2020-06-16	8:21 PM	Vaishu	Group	{}	0	5	1	1

➢ Keys in DataFrame



Whatsapp_project.ipynb - Colab

colab.research.google.com/drive/1WYjhqSRO8yPjKUmWjY5ancPSjvBCsD2c#scrollTo=jd7_AmKHJklj

Gmail YouTube Maps News Translate New Tab Quickstart — Flask...

Whatsapp_project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

ATTRIBUTES

```
[23] messages_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28518 entries, 0 to 31620
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype         
---  ------        --------------  -----         
 0   Date          28518 non-null  datetime64[ns]
 1   Time          28518 non-null  object        
 2   Author        28361 non-null  object        
 3   Message       28518 non-null  object        
 4   Emoji         28518 non-null  object        
 5   urlcount      28518 non-null  int64         
 6   Letter_Count  28518 non-null  int64         
 7   Word_Count    28518 non-null  int64         
 8   MessageCount  28518 non-null  int64         
dtypes: datetime64[ns](1), int64(4), object(4)
memory usage: 2.2+ MB
```

CONVERTING DATA TO EXCEL FILE

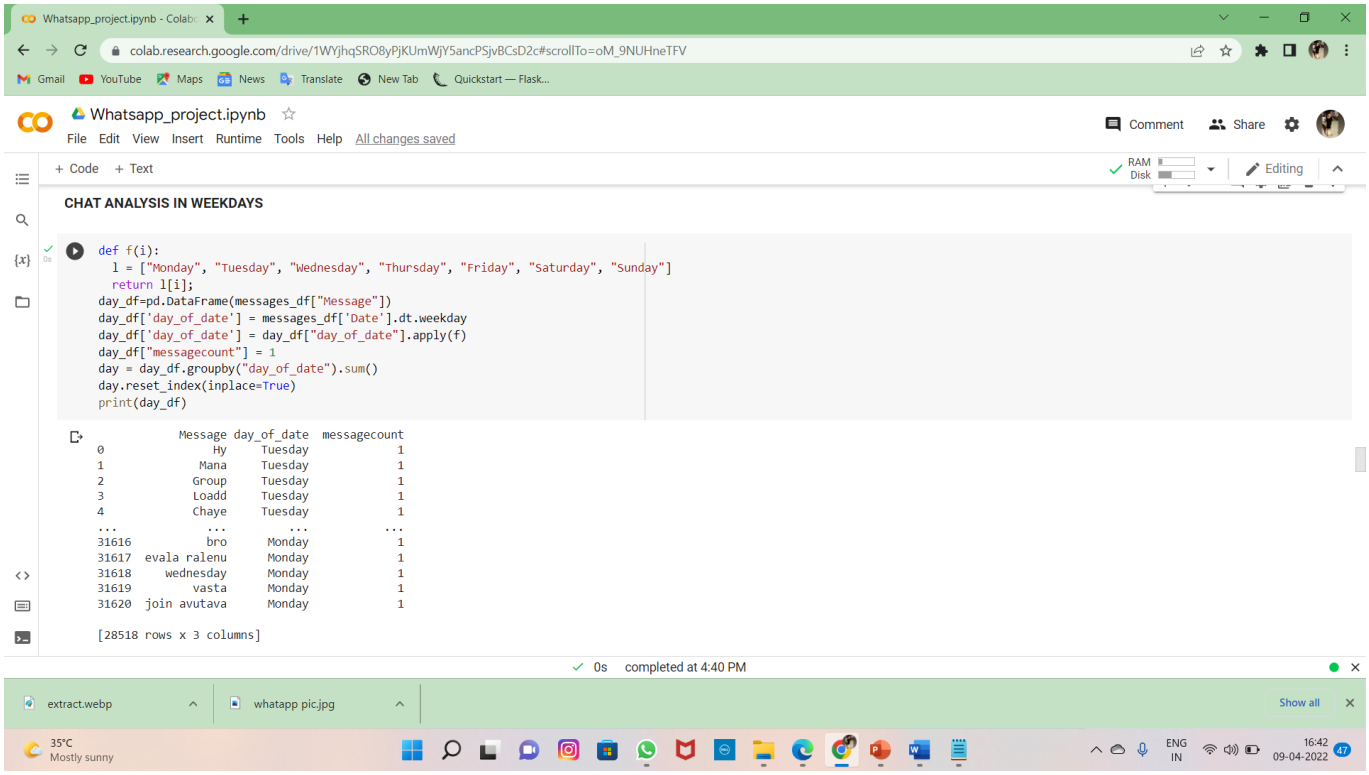
```
[24] df.to_csv('chat.csv')
```

0s completed at 4:38 PM

extract.webp whatapp pic.jpg Show all

35°C Mostly sunny ENG IN 16:40 09-04-2022

➢ Chat analyzation in weekdays



Whatsapp_project.ipynb - Colab

colab.research.google.com/drive/1WYjhqSRO8yPjKUmWjY5ancPSjvBCsD2c#scrollTo=oM_9NUHneTFV

Gmail YouTube Maps News Translate New Tab Quickstart — Flask...

Whatsapp_project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

CHAT ANALYSIS IN WEEKDAYS

```
def f(i):
  l = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]
  return l[i];
day_df=pd.DataFrame(messages_df["Message"])
day_df['day_of_date'] = messages_df['Date'].dt.weekday
day_df['day_of_date'] = day_df["day_of_date"].apply(f)
day_df["messagecount"] = 1
day = day_df.groupby("day_of_date").sum()
day.reset_index(inplace=True)
print(day_df)
```

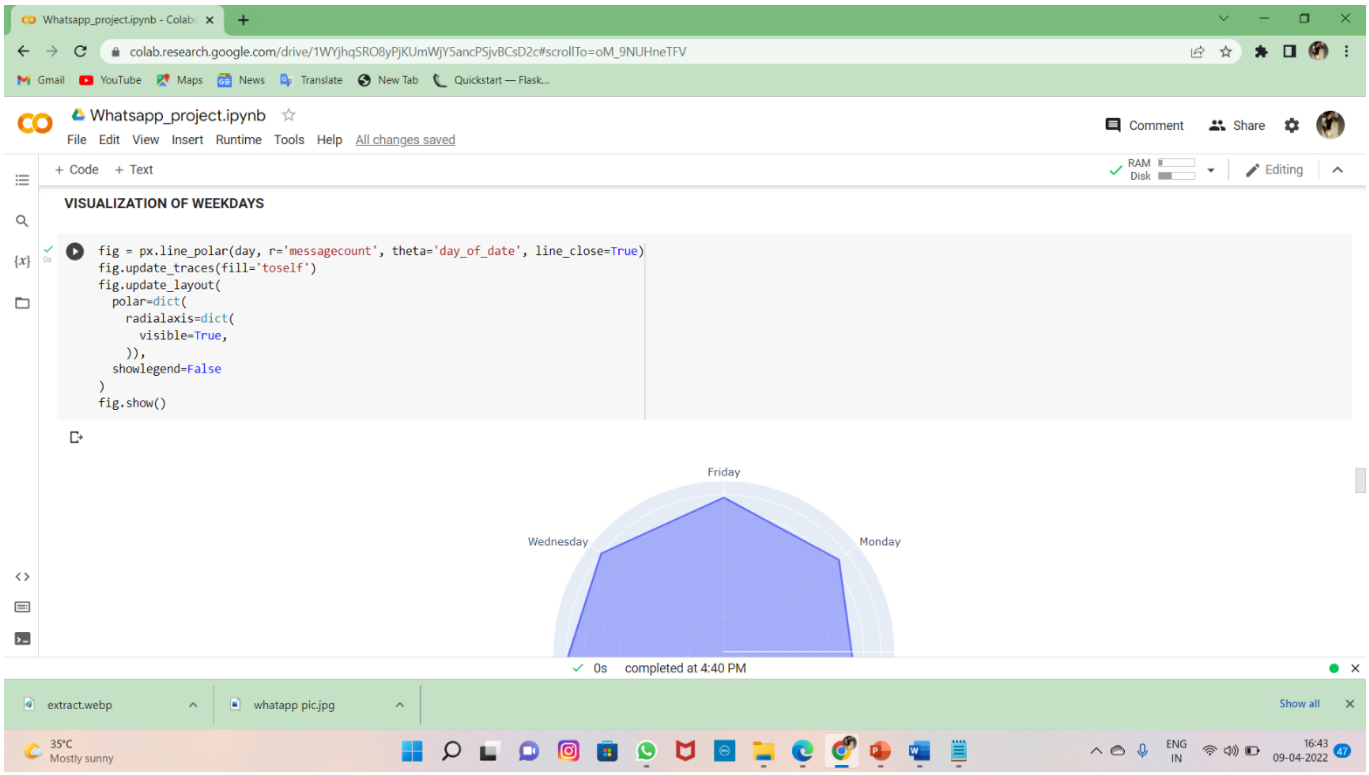
```
             Message day_of_date  messagecount
0                 Hy     Tuesday             1
1              Mana      Tuesday             1
2             Group      Tuesday             1
3             Loadd      Tuesday             1
4             Chaye      Tuesday             1
...             ...          ...           ...
31616           bro       Monday             1
31617  evala ralenu       Monday             1
31618     wednesday       Monday             1
31619         vasta       Monday             1
31620  join avutava       Monday             1

[28518 rows x 3 columns]
```

0s completed at 4:40 PM

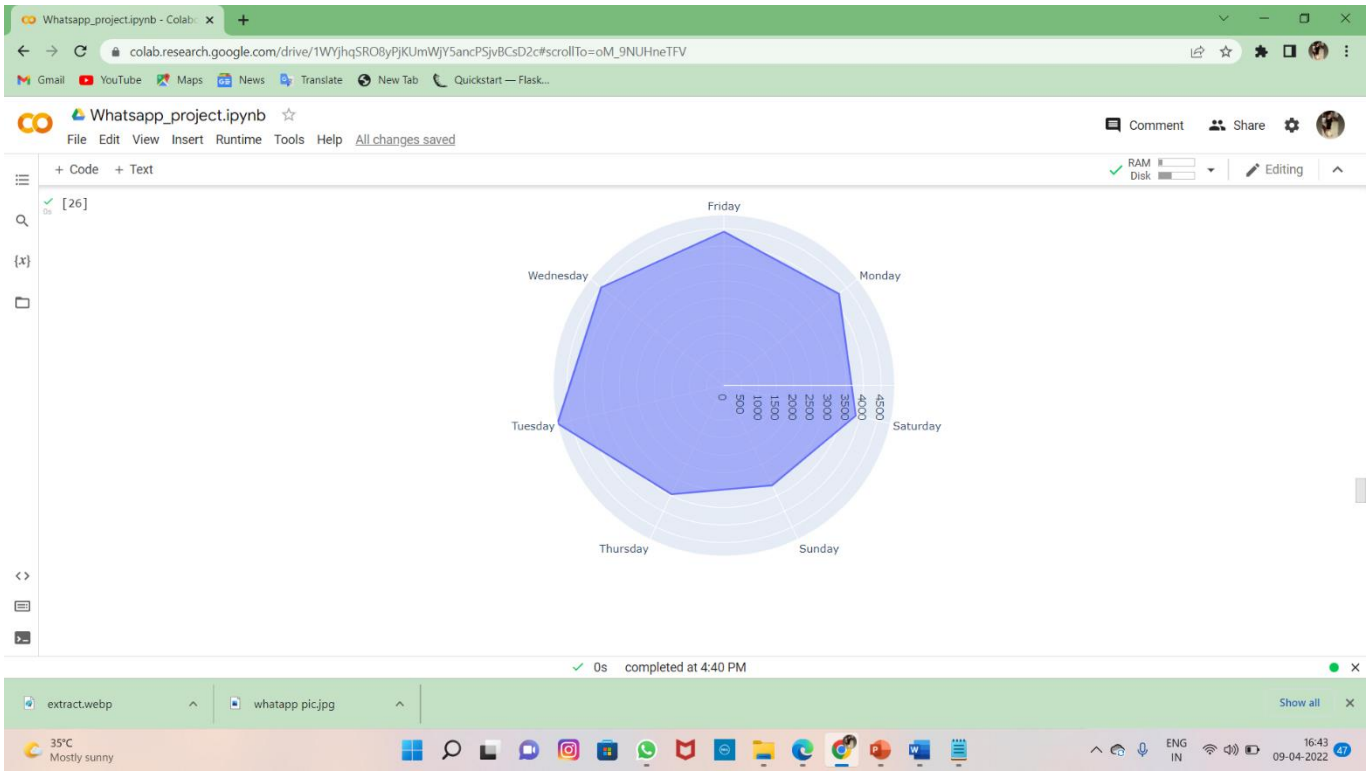
extract.webp whatapp pic.jpg Show all

35°C Mostly sunny ENG IN 16:42 09-04-2022

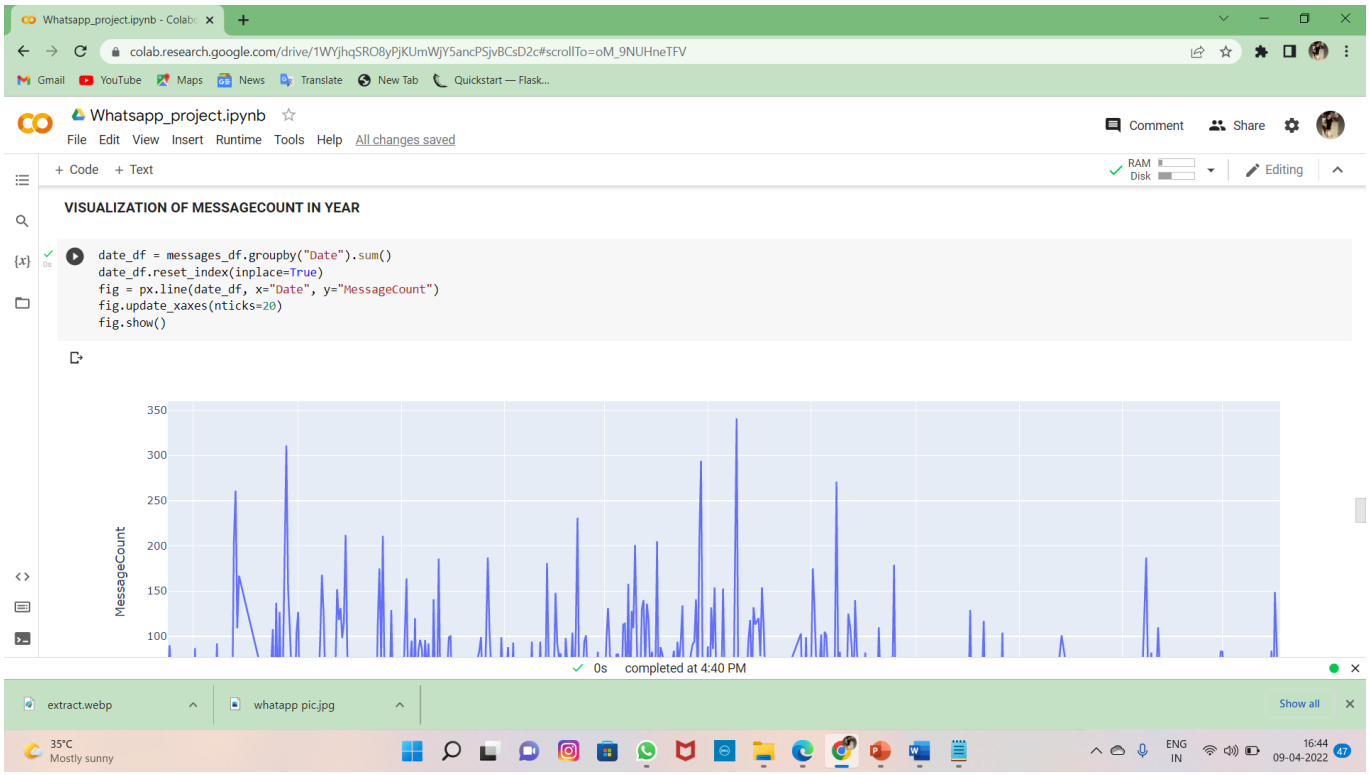
➢ code for chat in weekdays



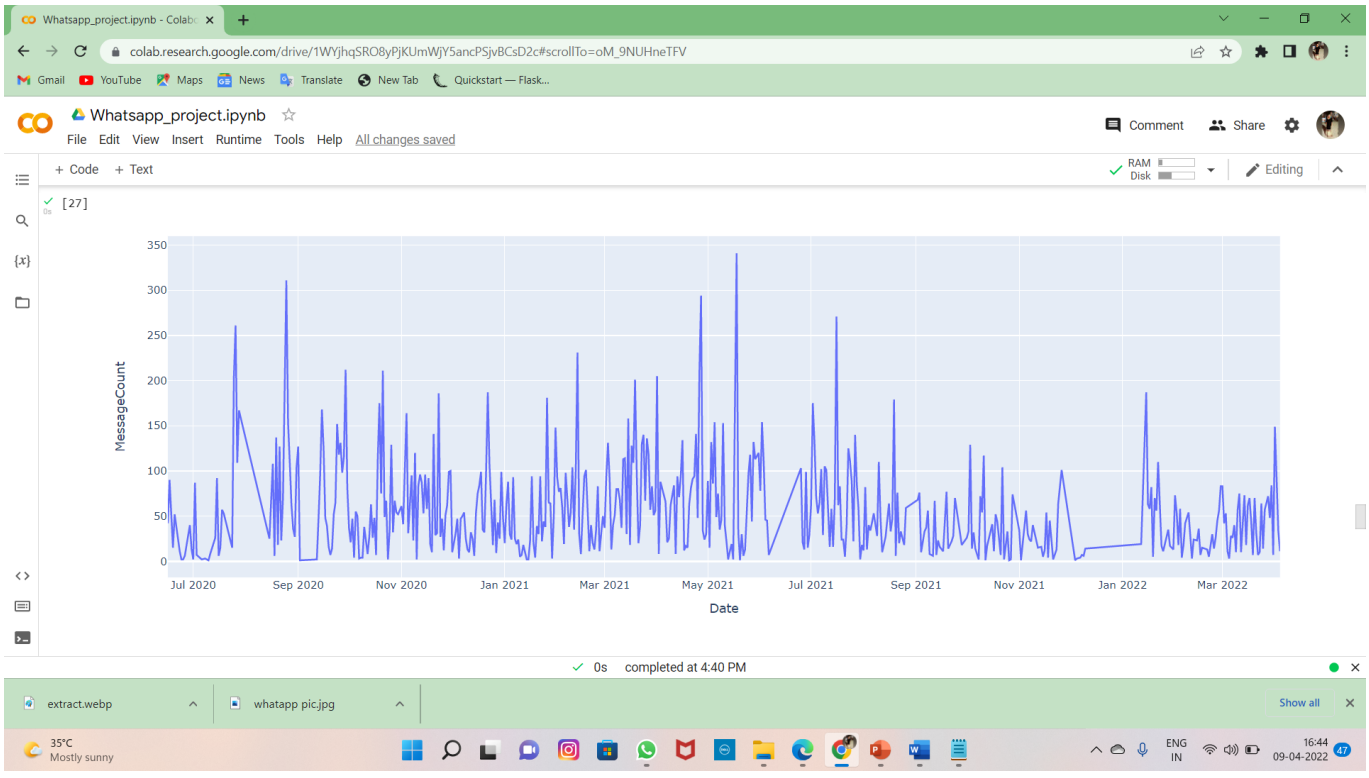
➢ Visualization of chat in weekdays



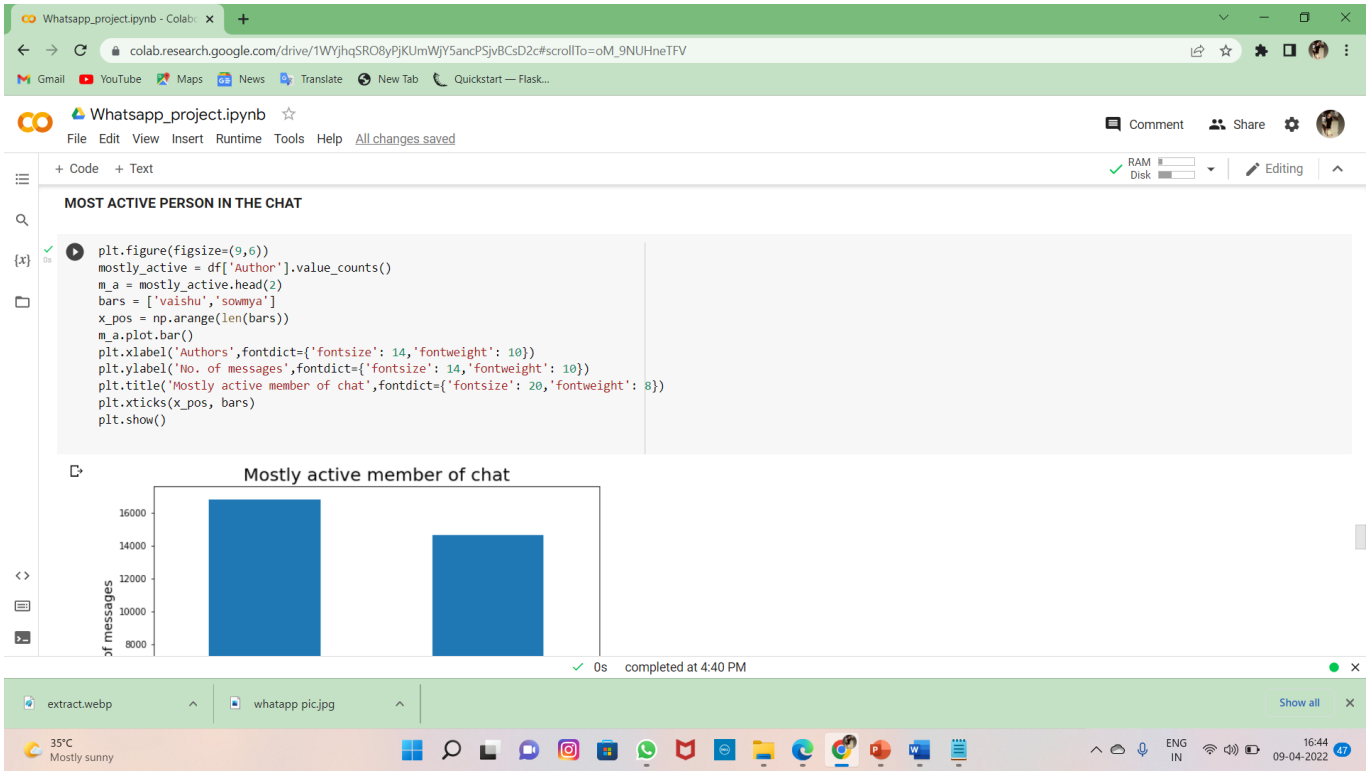
➢ Sourcecode for messagecount in years



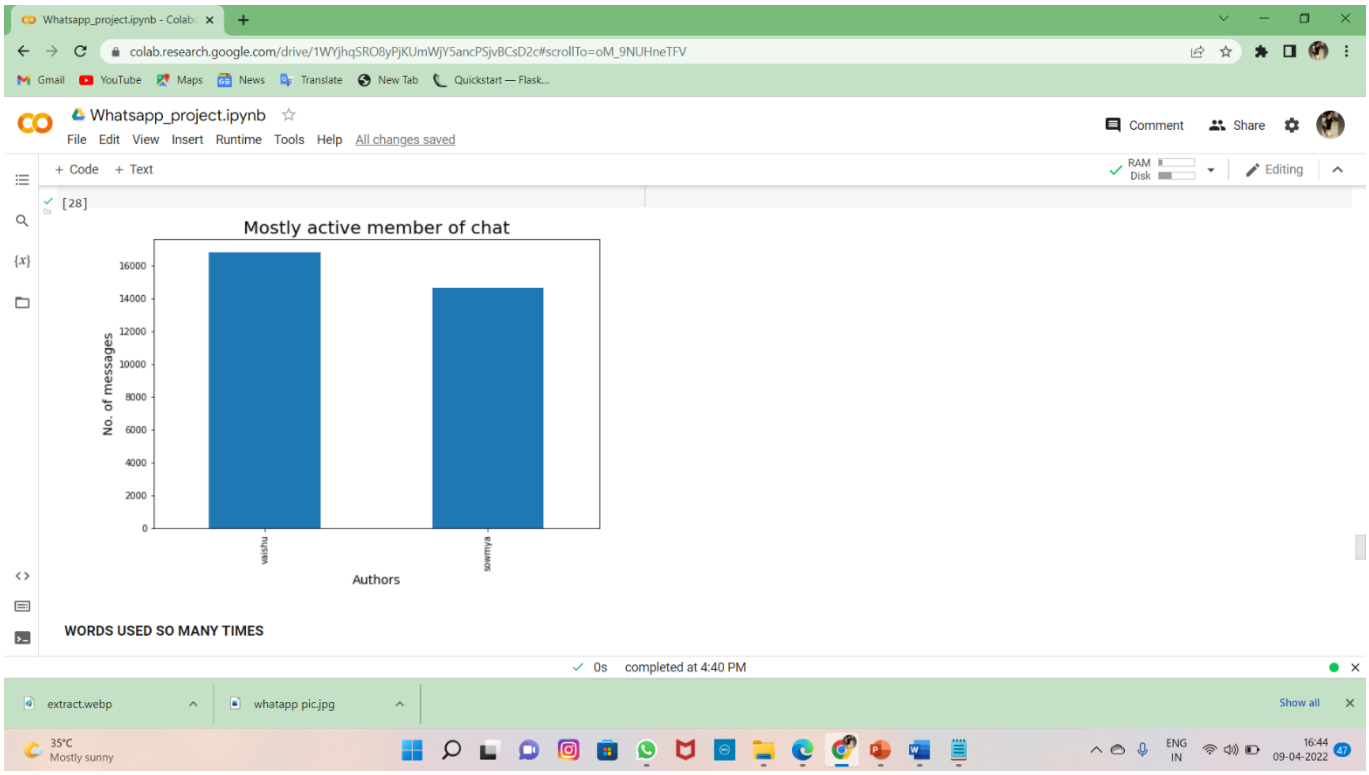
➢ Visualization of messagecount in years



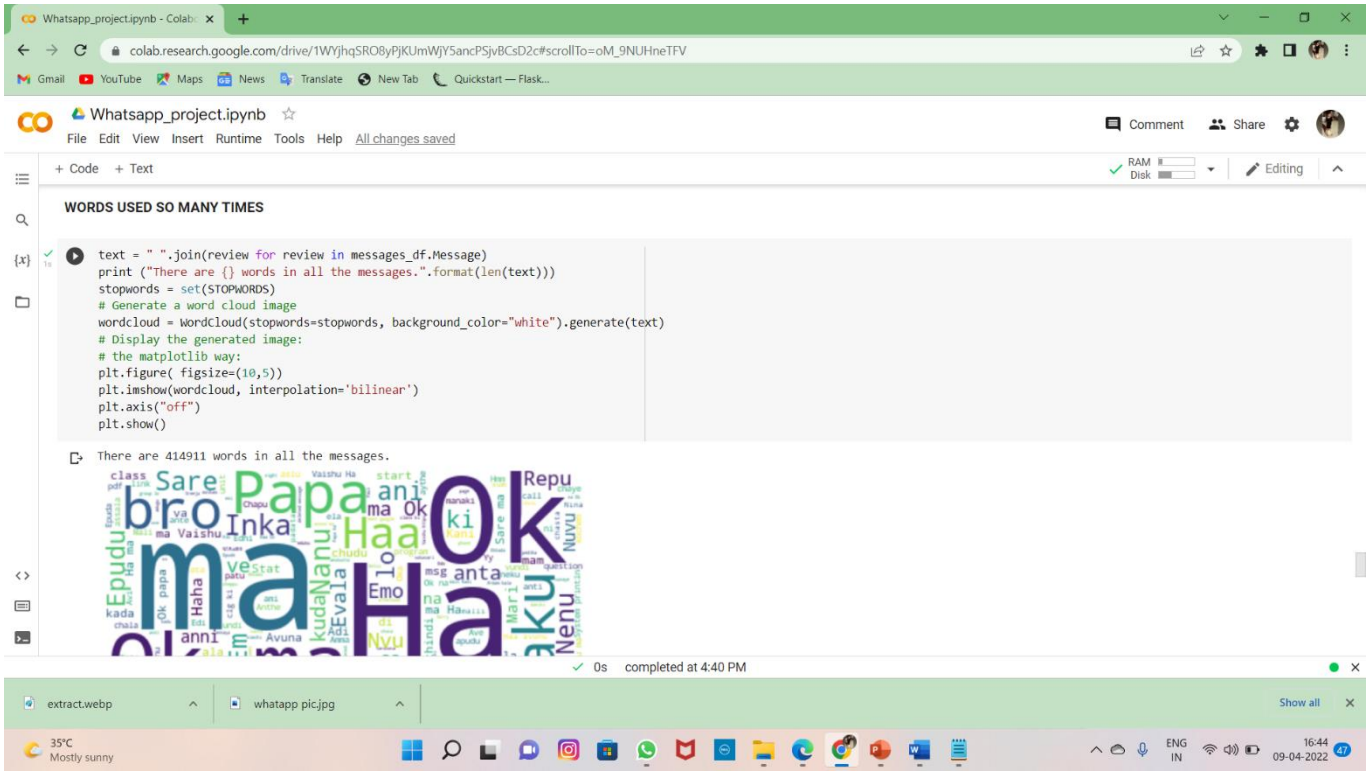
➢ Source code for the active person in the chart



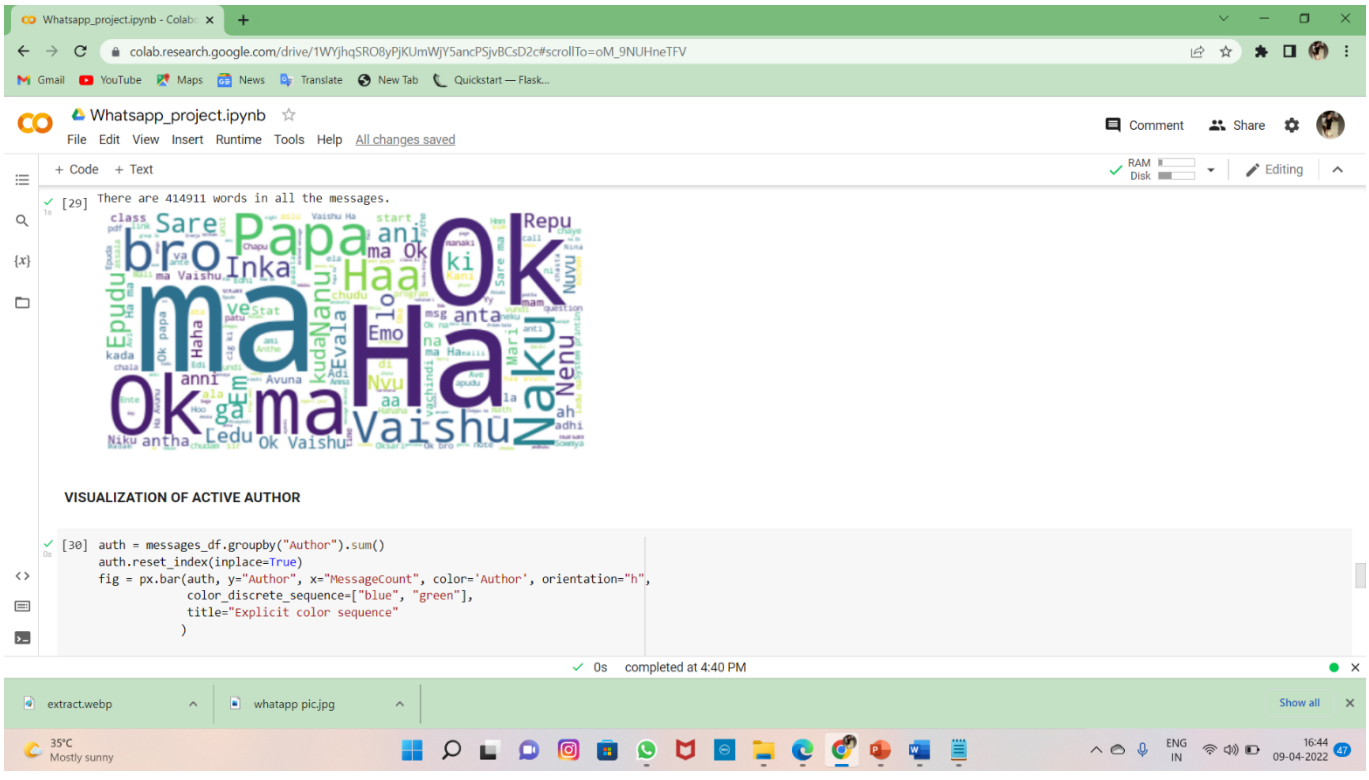
➢ Visualization of most active in the chat



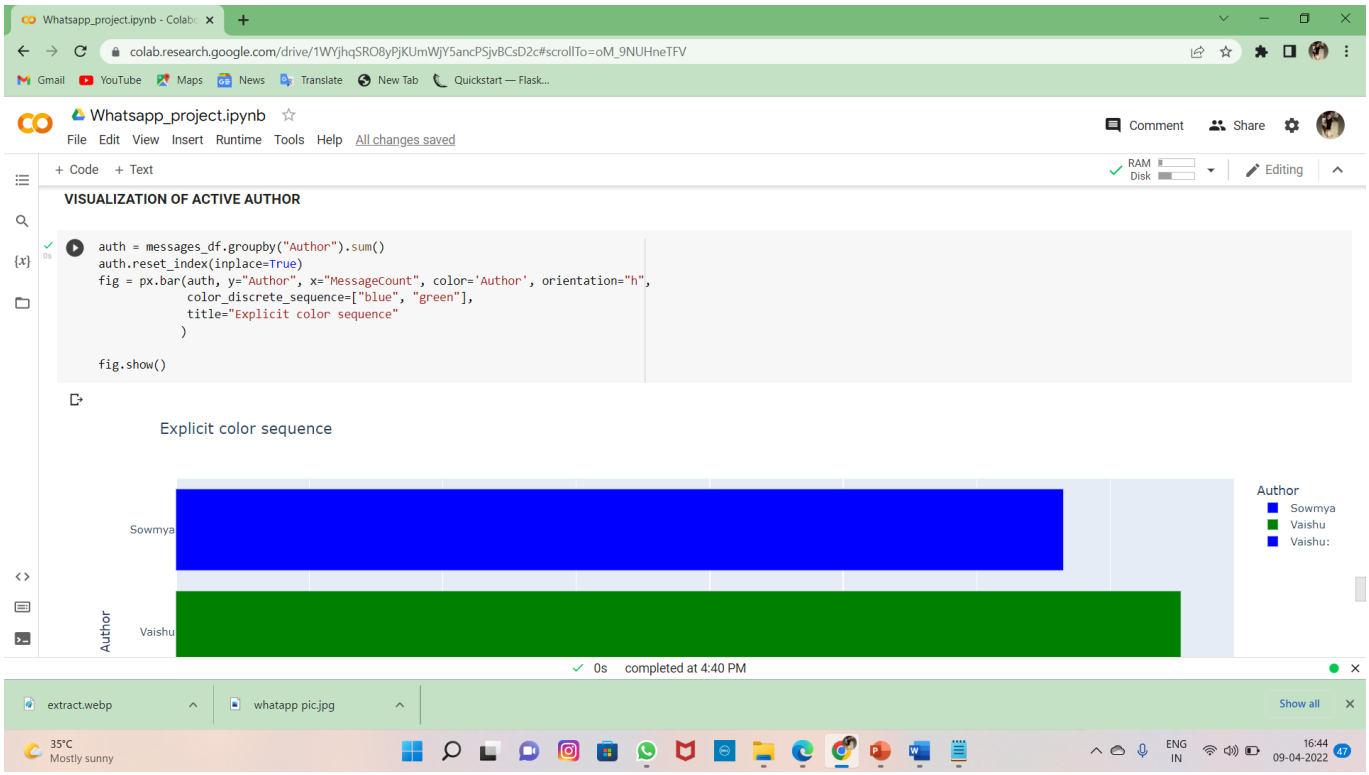
➢ Sourcecode for words used manytimes in the chat



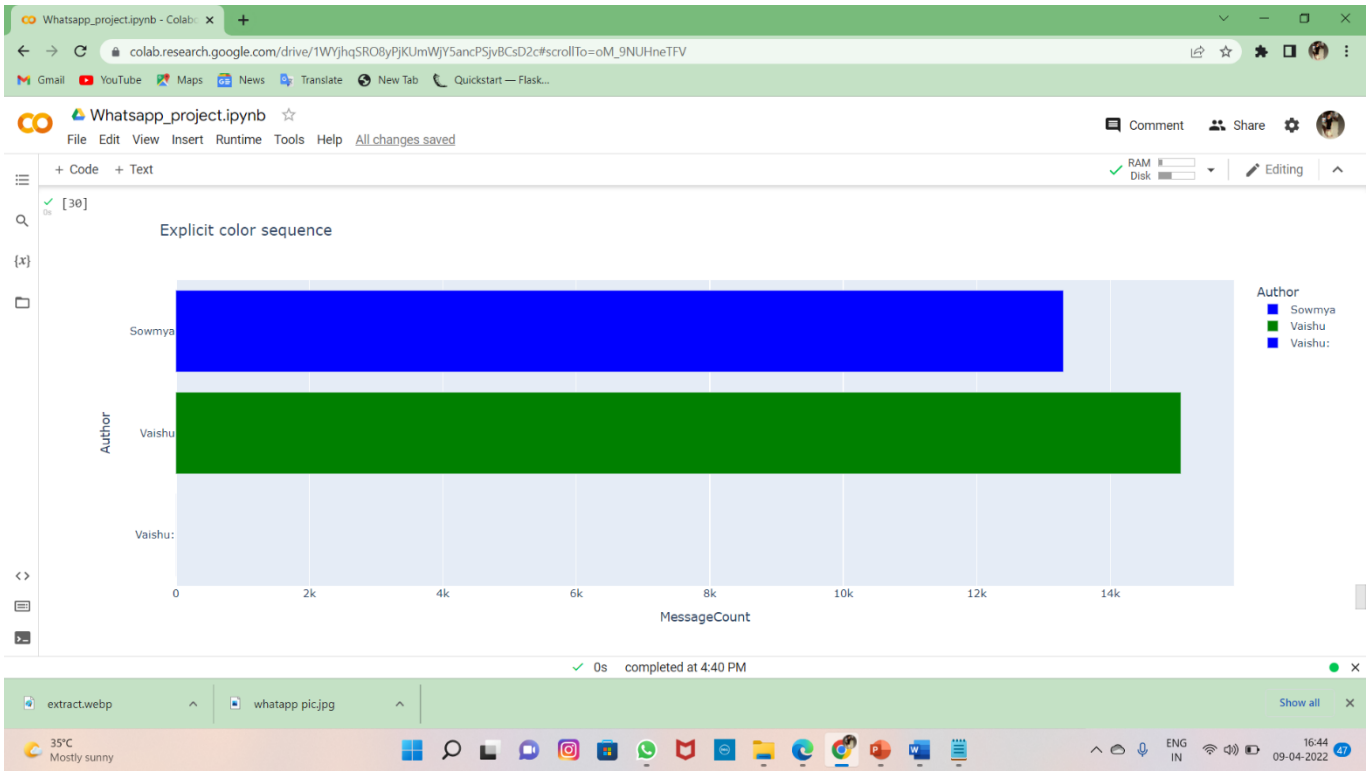
➢ Visualization of word mostly used



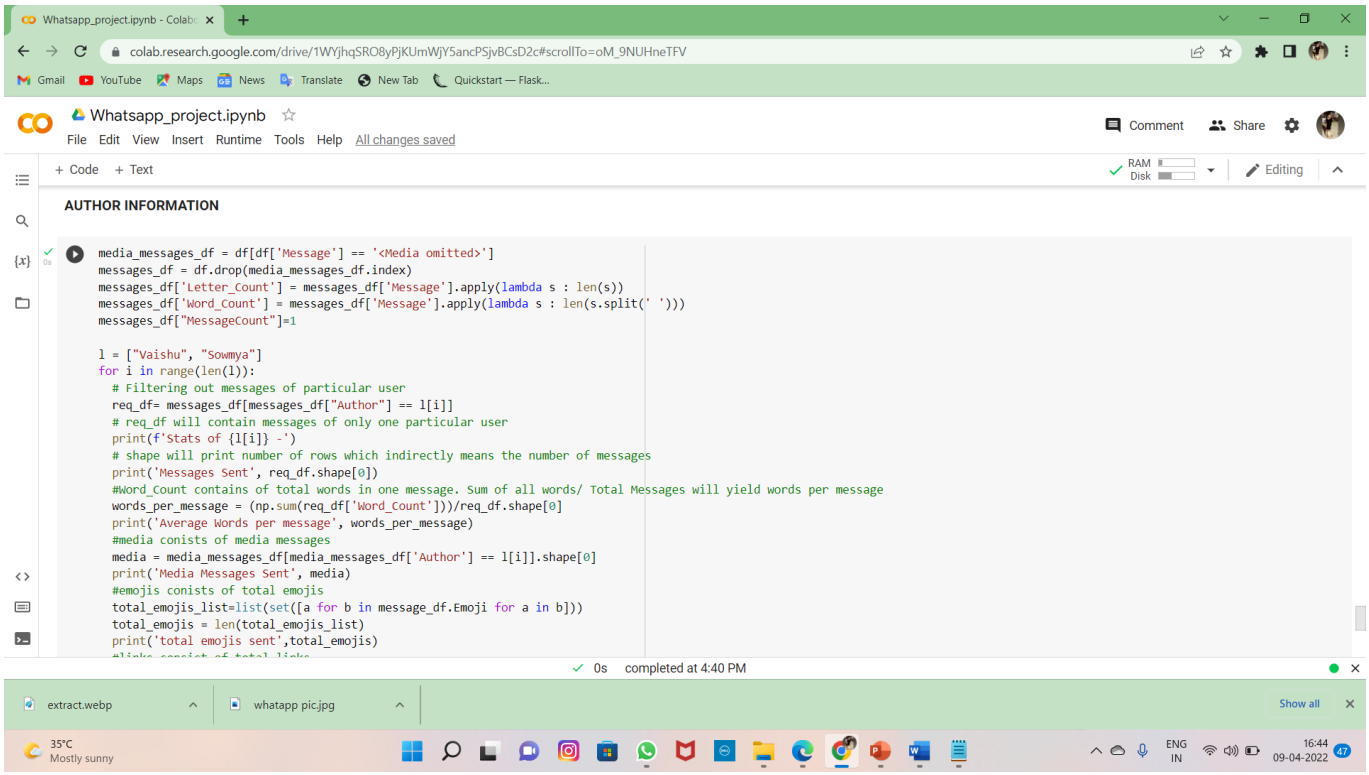
➢ Source code for active author with colors



➢ Visualization of active author with colors

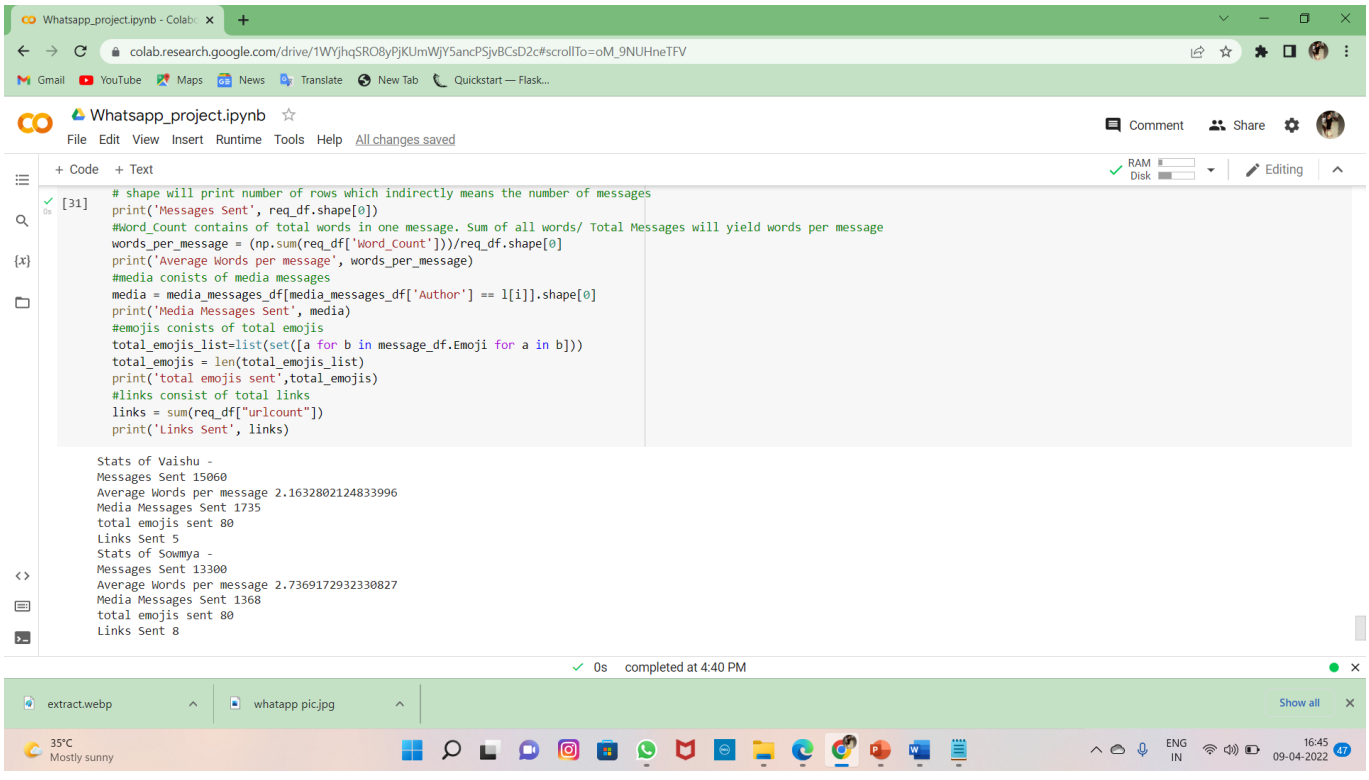


➢ Sourcecode for author information



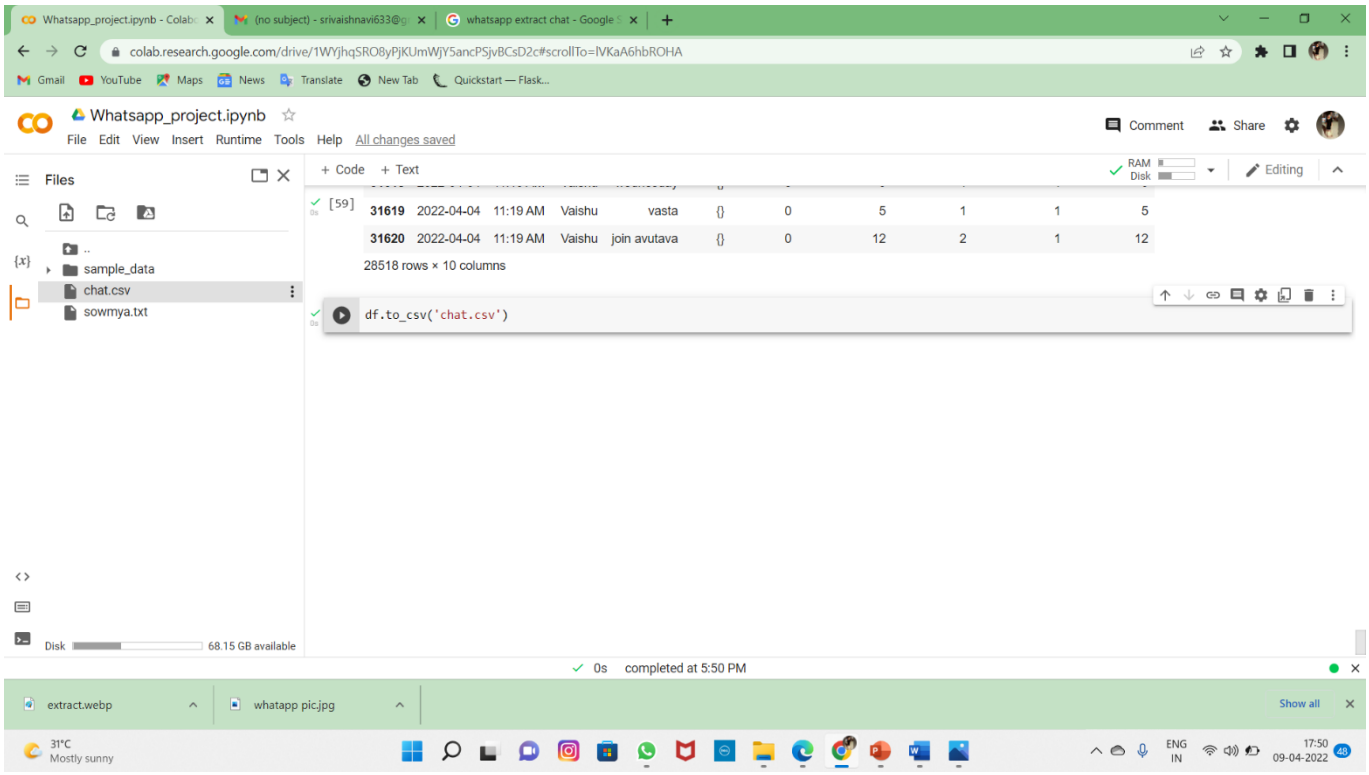
```
media_messages_df = df[df['Message'] == '<Media omitted>']
messages_df = df.drop(media_messages_df.index)
messages_df['Letter_Count'] = messages_df['Message'].apply(lambda s : len(s))
messages_df['Word_Count'] = messages_df['Message'].apply(lambda s : len(s.split(' ')))
messages_df["MessageCount"]=1

l = ["Vaishu", "Sowmya"]
for i in range(len(l)):
  # Filtering out messages of particular user
  req_df= messages_df[messages_df["Author"] == l[i]]
  # req_df will contain messages of only one particular user
  print(f'Stats of {l[i]} -')
  # shape will print number of rows which indirectly means the number of messages
  print('Messages Sent', req_df.shape[0])
  #Word_Count contains of total words in one message. Sum of all words/ Total Messages will yield words per message
  words_per_message = (np.sum(req_df['Word_Count']))/req_df.shape[0]
  print('Average Words per message', words_per_message)
  #media conists of media messages
  media = media_messages_df[media_messages_df['Author'] == l[i]].shape[0]
  print('Media Messages Sent', media)
  #emojis conists of total emojis
  total_emojis_list=list(set([a for b in message_df.Emoji for a in b]))
  total_emojis = len(total_emojis_list)
  print('total emojis sent',total_emojis)
```

➢ Output for the authors with analyzation

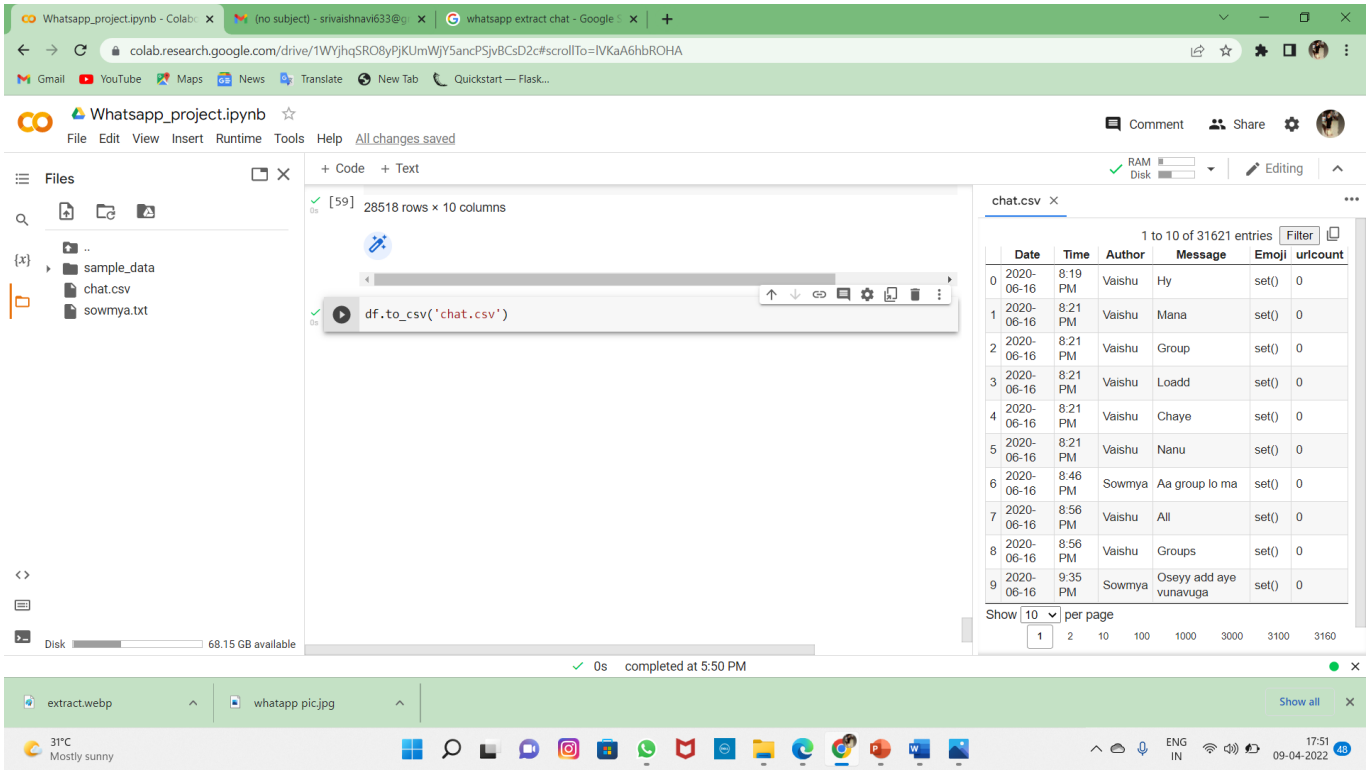


```
[31]  # shape will print number of rows which indirectly means the number of messages
      print('Messages Sent', req_df.shape[0])
      #Word_Count contains of total words in one message. Sum of all words/ Total Messages will yield words per message
      words_per_message = (np.sum(req_df['Word_Count']))/req_df.shape[0]
      print('Average Words per message', words_per_message)
      #media conists of media messages
      media = media_messages_df[media_messages_df['Author'] == l[i]].shape[0]
      print('Media Messages Sent', media)
      #emojis conists of total emojis
      total_emojis_list=list(set([a for b in message_df.Emoji for a in b]))
      total_emojis = len(total_emojis_list)
      print('total emojis sent',total_emojis)
      #links consist of total links
      links = sum(req_df["urlcount"])
      print('Links Sent', links)

Stats of Vaishu -
Messages Sent 15060
Average Words per message 2.1632802124833996
Media Messages Sent 1735
total emojis sent 80
Links Sent 5
Stats of Sowmya -
Messages Sent 13300
Average Words per message 2.7369172932330827
Media Messages Sent 1368
total emojis sent 80
Links Sent 8
```

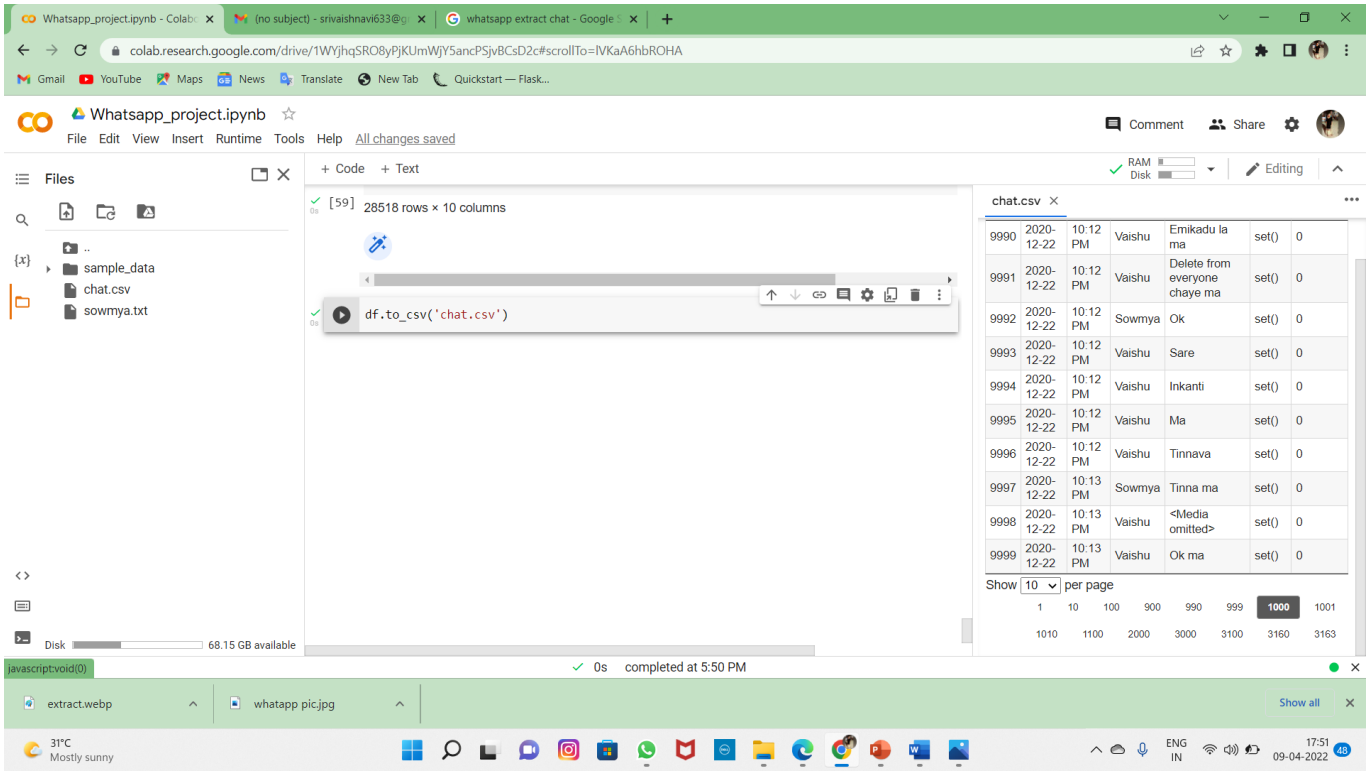
➢ Converting the DataFrame to Excel file



➢ Now the data is converted to excel file .we can see at the right of the screen



➢ Now the chat is simple and understanding



9990	2020-12-22	10:12 PM	Vaishu	Emikadu la ma	set()	0
9991	2020-12-22	10:12 PM	Vaishu	Delete from everyone chaye ma	set()	0
9992	2020-12-22	10:12 PM	Sowmya	Ok	set()	0
9993	2020-12-22	10:12 PM	Vaishu	Sare	set()	0
9994	2020-12-22	10:12 PM	Vaishu	Inkanti	set()	0
9995	2020-12-22	10:12 PM	Vaishu	Ma	set()	0
9996	2020-12-22	10:12 PM	Vaishu	Tinnava	set()	0
9997	2020-12-22	10:13 PM	Sowmya	Tinna ma	set()	0
9998	2020-12-22	10:13 PM	Vaishu	<Media omitted>	set()	0
9999	2020-12-22	10:13 PM	Vaishu	Ok ma	set()	0

CONCLUSION

Summary:

In conclusion, it can be said that the capabilities of the WhatsApp application and the power of the python programming language in implementing whatever network data analysis intended, cannot be overemphasized. This work was able to discuss the WhatsApp application and its libraries, to create an analysis of a WhatsApp group chat and visually represent the chat between the two members. A pseudocode of the plot was given and at the end, visual representation of the plot was implemented. Also, an analysis of 2menbers in the chat were done. The system was done with python, and the python libraries that were implemented includes, NumPy, Pandas, Matplotlib. At the end of the work expected results were obtained and the analysis was able to show the level of participation of the various individuals on the given WhatsApp chat. On serious note this system has the ability to analyze any WhatsApp chat data input into it.

BIBLIOGRAPHY

[1] Available from: http://www. statista.com/statistics/260819/numberof-monthly-active-WhatsApp-users. Number of monthly active WhatsApp users worldwide from April 2013 to February 2016(in millions).

[2] Ahmed, I., Fiaz, T., "Mobile phone to youngsters: Necessity or addiction", African Journal of Business Management Vol.5 (32), pp. 12512-12519, Aijaz, K. (2011).

[3] Aharony, N., T., G., The Importance of the WhatsApp Family Group: An Exploratory Analysis. "Aslib Journal of Information Management, Vol. 68, Issue 2, pp.1-37" (2016).

[4] Access Data Corporation. FTK Imager, 2013. Available at http://www.accessdata.com/support/product-downloads.

[5] D.Radha, R. Jayaparvathy, D. Yamini, "Analysis on Social Media Addiction using Data Mining Technique", International Journal of Computer Applications (0975 – 8887) Volume 139 – No.7, pp. 23- 26, April 2016.

[6] Jessica Ho, Ping Ji, Weifang Chen, Raymond Hsieh, "Identifying google talk", IEEE International Conference on Intelligence and Security Informatics, ISI '09, pp. 285-290, 2009.

[7] Mike Dickson, "An examination into AOL instant messenger 5.5 contact identification.", Digital Investigation, ScienceDirect, vol. 3, issue 4, pp. 227-237, 2006.

[8] Mike Dickson, "An examination into yahoo messenger 7.0 contact identification", Digital Investigation, ScienceDirect, vol. 3, issue 3, pp. 159-165, 2006.

21st March 2022
Vijayawada

CERTIFICATE OF INTERNSHIP

TO WHOMSOEVER IT MAY CONCERN

This is to certify Ms. N. Sri Vaishnavi has successfully completed her Internship Program as a "Python Intern" with Codegnan IT Solutions (OPC) Pvt Ltd, Vijayawada from January 21st, 2022 to March 21st, 2022.

As a part of her internship, she efficiently contributed to the work and was found to be hard-working, keen to learn, and ready to accept responsibilities.

We wish her all the best in future endeavors.

For Codegnan IT Solutions Pvt Ltd.,

Saikiran Tarigopula Taopula
HR - Manager

40-5-19/16, Prasad Naidu Complex,
P.B.Siddhartha Busstop, Moghalrajpuram
Vijayawada-520010

21st March 2022
Vijayawada

CERTIFICATE OF INTERNSHIP

TO WHOMSOEVER IT MAY CONCERN

This is to certify Mr.Y. Soma Sekhar Dinesh has successfully completed her Internship Program as a "Python Intern" with Codegnan IT Solutions (OPC) Pvt Ltd, Vijayawada from January 21st, 2022 to March 21st, 2022.

As a part of her internship, she efficiently contributed to the work and was found to be hard-working, keen to learn, and ready to accept responsibilities.

We wish her all the best in future endeavors.

For Codegnan IT Solutions Pvt Ltd.,

Saikiran Taopula
HR - Manager

40-5-19/16, Prasad Naidu Complex,
P.B.Siddhartha Busstop, Moghalrajpuram
Vijayawada-520010