

**PROJECT REPORT**

**Twitter Search Application**

**“Chirp Search”**

**Prepared by**

Vishnuram Jatin Bangaru(vb440)

Ankeeta Priyam(ap2213)

Noopur Singh(ns1482)

Ganesh Raj Kyatham(grk62)

Table of Contents

[**INTRODUCTION** 3](#_Toc132218008)

[**PROJECT ARCHITECTURE** 4](#_Toc132218009)

[**PROCESSING TWEETS FOR STORING IN DATASTORES** 6](#_Toc132218010)

[**SEARCH APPLICATION DESIGN** 7](#_Toc132218011)

[**RESULTS** 8](#_Toc132218012)

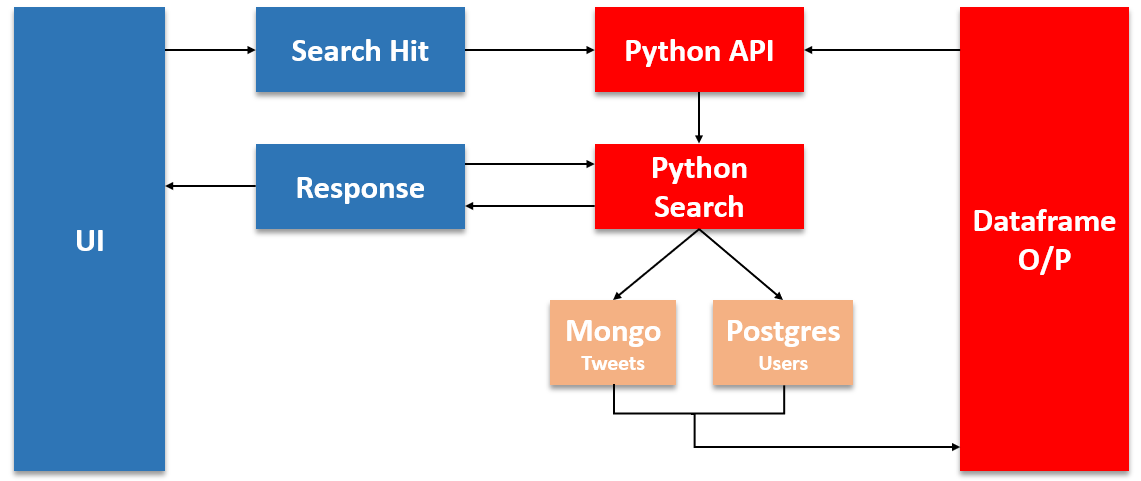
[**REFERENCES** 10](#_Toc132218013)

# **INTRODUCTION**

Twitter is a popular social media platform that enables users to share short messages, known as tweets, with a wide audience. With over 330 million monthly active users, Twitter has become a critical communication tool for individuals, businesses, and organizations worldwide. The platform's real-time nature and broad reach makes it a valuable source of information for breaking news, trending topics, and public opinion. Overall, Twitter plays a significant role in shaping public discourse and connecting people from diverse backgrounds and perspectives.

As part of the MSc Data Science program at Rutgers University, State University of New Jersey, course 954:694:01, the project is started that involves developing a search application utilizing Twitter data. The objective is to design and store the information in multiple data stores to enable rapid access via Python. Additionally, the project involves implementing a caching system for the top search results to minimize the need for repeated visits to the data stores.

# **PROJECT ARCHITECTURE**



*Fig. 1*

The project architecture displayed above depicts the flow of information. Firstly, the UI receives input from the user and sends it to the Python API.

The Python API then directs the control to the search function in the Python source code. Within the search function, a query string is formulated based on the input received from the user, and it is sent to the mongo dB context, where the Tweets table contains the relevant tweets.

The output of this search is returned to the Search Function, which generates an output Dataframe that includes the tweet, username, and the number of retweets for that tweet. The Python API then receives this result and sends it back to the UI, which displays the response to the user.

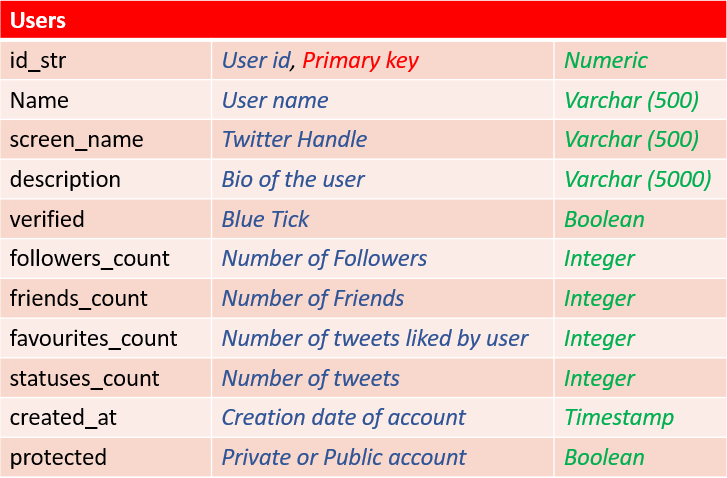
If the user clicks on the username or number of tweets, the Python API is called again, and depending on the selection, either the Users table in PostgresSQL (for user click) or the Tweets table in MongoDB (for tweet click) is queried. If the user clicks on the username, the details of the user are returned. If the user clicks on the retweet count, the retweeted tweets are returned.

**PERSISTED DATA MODEL AND DATASTORES**

The project incorporates two different types of databases - Relational and Non-Relational. For Relational database management, PostgresSQL is employed, and for Non-Relational database management, MongoDB is utilized.

The tweet dataset is distributed between these databases. User level details are stored in the Users table in PostgresSQL, while tweet-related details are stored in the Tweets table in MongoDB.

The Users table's architecture and schema are displayed in the figure 2 and The Tweets table's architecture and schema are illustrated in the figure 3.



*Fig. 2*

# **PROCESSING TWEETS FOR STORING IN DATASTORES**

Make sure that you processed the tweets and stored the information (users, tweets) in the datastores one at a time (DO NOT load all the data into a dataframe and then load the dataframe into the datastore)

# **SEARCH APPLICATION DESIGN**

What types of searches are allowed? What kinds of drill-downs are allowed?

How were search queries translated into queries for the datastores?

What was your notion of relevance (i.e. how did you order the results of the search queries)?

How is the cache used?

# **RESULTS**

Show the results of each type of query.

Timings of your test search queries (make sure you are hitting cached and non cached data)

**CONCLUSIONS**

# **REFERENCES**