RUTGERS

# Twitter Search Application

## REPORT SPRING 2023


**Presented By**

N Kiran Reddy – 220008729
Dileep Kumar P – 218008143
Manish M – 219007841
Sai Charan K V – 220007887

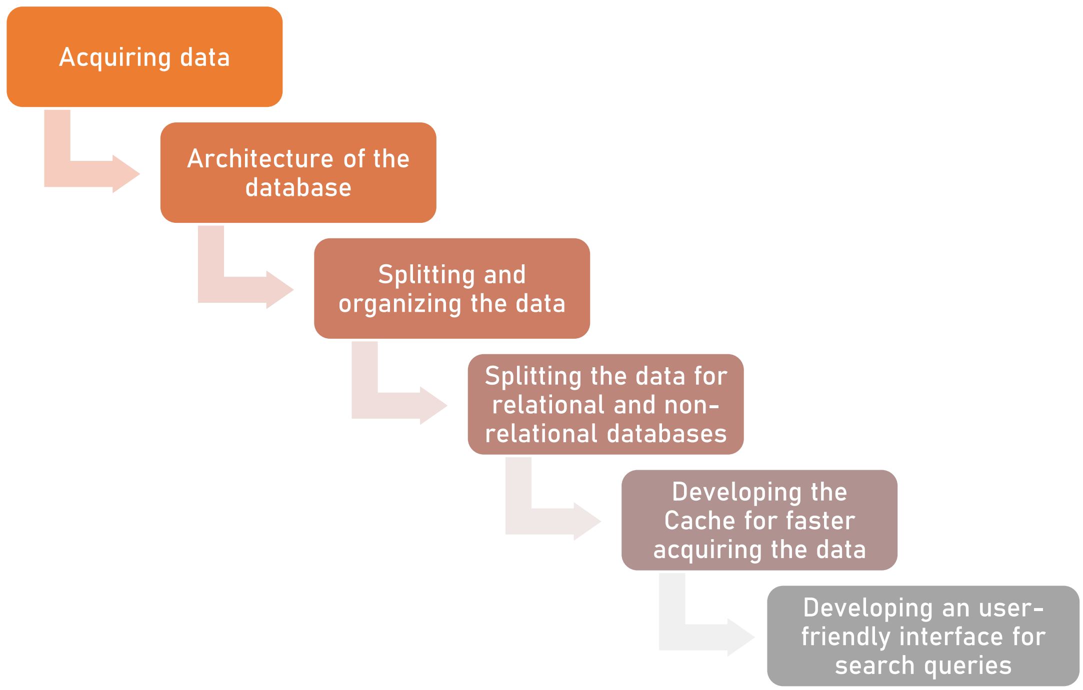

**GitHub Link –**
*https://github.com/dileepPDK123/DBMS-694-Team03-2023.git*

**Contributors –**
*Kiranreddy-n*
*dileepPDK123*
*Manish533005*
*CharanK33*

# Executive Summary

## The milestone graph

The graph illustrates the step-by-step approach that the project will follow. The Twitter dataset that will be used in this project contains a significant amount of data in the form of JSON objects. To create a database, this data must be decoded first. Once the data has been decoded, it will be processed in accordance with the following stages.




# Introduction

## Background information

The goal of this project is to create a search application that can analyze Twitter data using both relational and non-relational data storages. The project will involve storing information about tweets and users in different databases to optimize the efficiency of data access. Additionally, the application will implement a cache for frequently accessed data to further enhance performance. The search interface will allow users to query tweets by various parameters, including string, hashtag, user, time range and many more. The application will also provide drill-down features, such as displaying the author of the tweet, when it was tweeted, and the number of retweets. Python will be the primary programming language used for the project, including loading data into the data stores using mysql integrated with python, implementing the cache using valid dictionaries, and developing the UI using python TKinter.

## Methodology

### Relational Database

For this project, MySQL integrated with Python has been chosen as the database management system to store the Twitter dataset. The dataset contains a large amount of data in the form of JSON objects, which will be imported into the database. Python code will be implemented to insert the data from each JSON object into their respective fields in the database one line at a time. This will simulate the processing of data arriving in a stream.

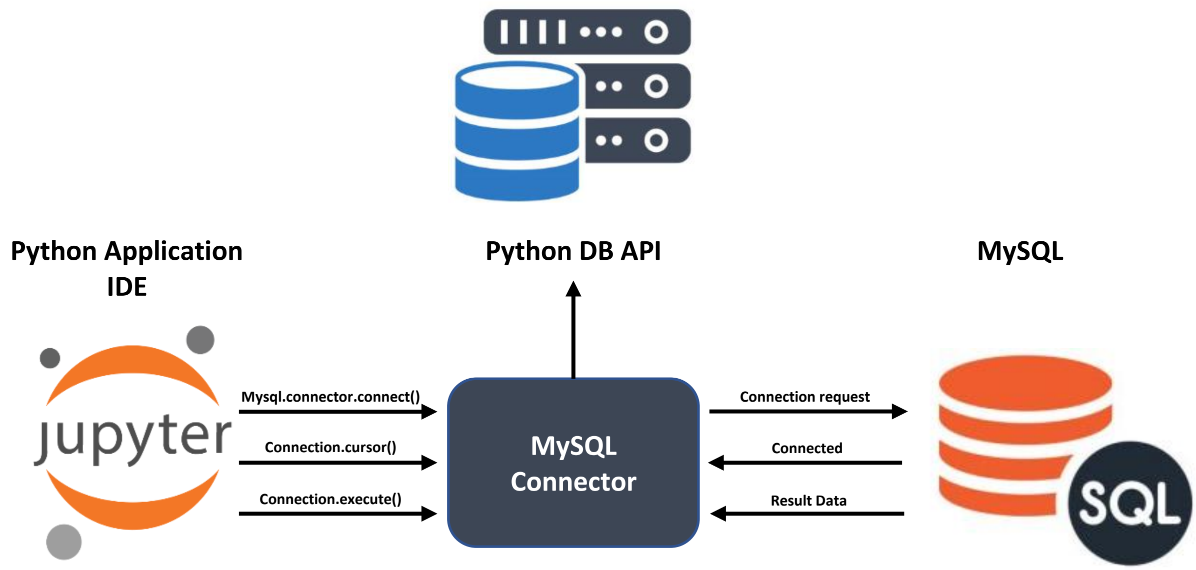



***figure i*** *– Representation of MySQL connection with Python*

MySQL is a widely used and popular relational database management system known for its robustness, performance, and scalability. It offers various features such as transaction support, foreign keys, and stored procedures, making it suitable for handling complex datasets. The integration of MySQL with Python provides an efficient and easy-to-use interface for managing data in the database from within Python code.

### Non – Relational database

To store the non-relational twitter data for the project, MongoDB will be used. Since MongoDB's document-oriented data model is well suited to JSON, the tweets' format makes sense. Using Python code, which analyzes each tweet individually and adds it into the database, the data will be imported into MongoDB. Since there is no need for pricey joins when using a non-relational database like MongoDB, data querying is likewise more effective. This is crucial for the search application, which must swiftly fetch tweets using search parameters like string, hashtag, and user.

**Graphical User Interface**

To build the search interface for the project, the team has decided to use Python's built-in GUI framework, Tkinter. Tkinter is a widely used and popular GUI toolkit for Python that provides a simple way to create desktop applications. It is a cross-platform toolkit that can be used on different operating systems such as Windows, Linux, and macOS.

One of the main advantage of Tkinter is its ease of integration with both MySQL and MongoDB. Python has libraries such as PyMySQL and PyMongo that allow developers to connect to these databases easily. Tkinter also provides features that enable the creation of dynamic and interactive user interfaces, such as buttons, input fields, and list boxes, which are necessary for building the search application. Another advantage of Tkinter is that it comes pre-installed with Python, which means developers do not need to install any additional packages or libraries to use it. This makes it easy to develop and distribute the search application to different users without worrying about compatibility issues.

## Pipeline of the project model

The project follows a clear pipeline that starts with the tkinter GUI, where users select their desired search options. From there, the relevant Python files are accessed to execute the search. The search begins by looking into the cache for frequently accessed data. If the data is not available in the cache, the program accesses the MySQL database to retrieve the data. In case the query requires access to the non-relational MongoDB database, the relevant Python file is accessed. The data is then displayed on the GUI. This pipeline ensures that the search is performed efficiently and the relevant data is accessed quickly, providing a seamless experience to the users.

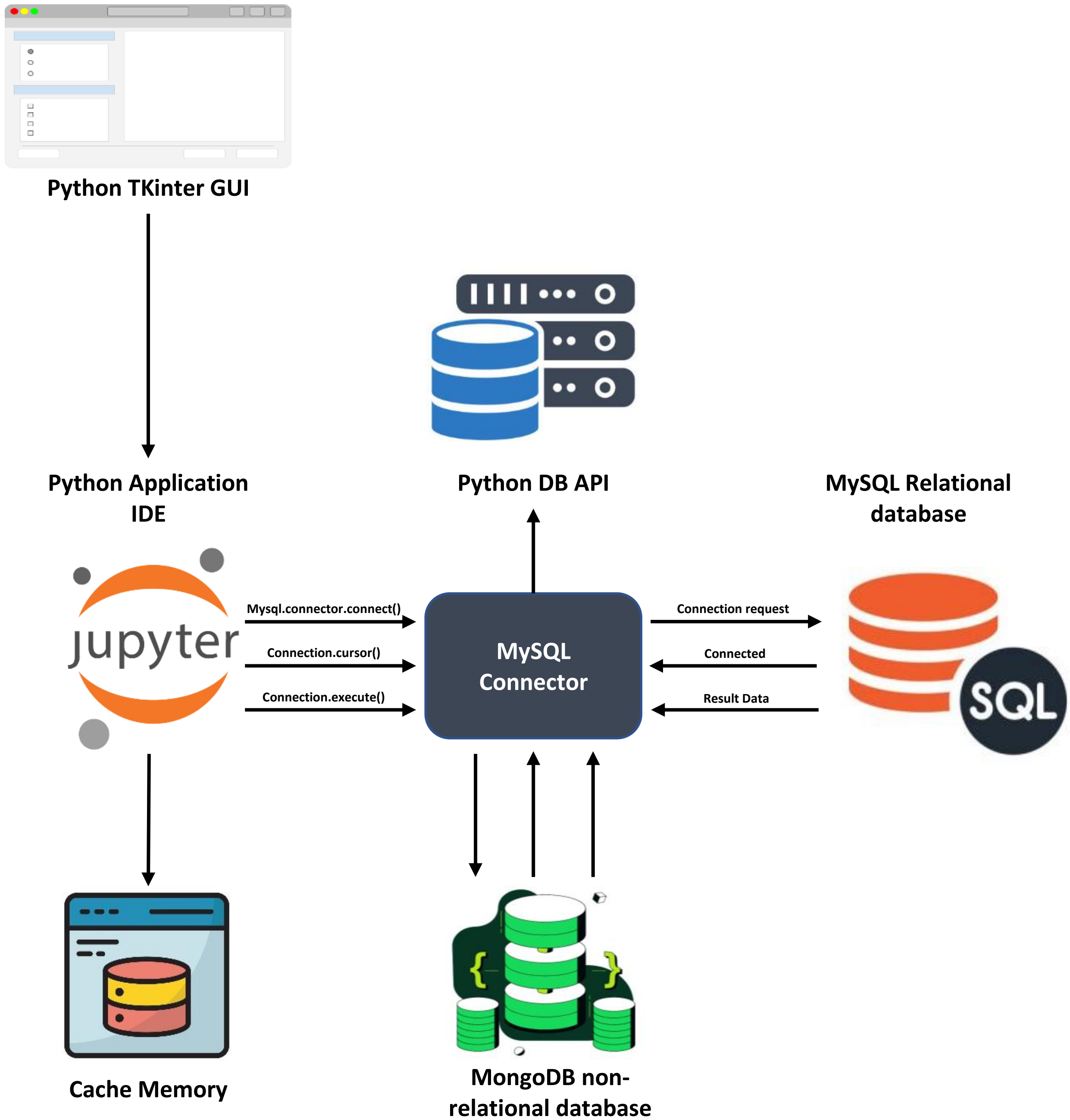



***figure ii** – Project pipeline*

# Project Process

## Relational Database

The relational database utilized in this project comprises of five tables: user data, tweet data, retweet data, tweet hashtag data, and tweets string data. Each table contains distinct sets of data that are essential for the efficient processing of search queries. To better understand the content present in each table, please refer to the picture below:

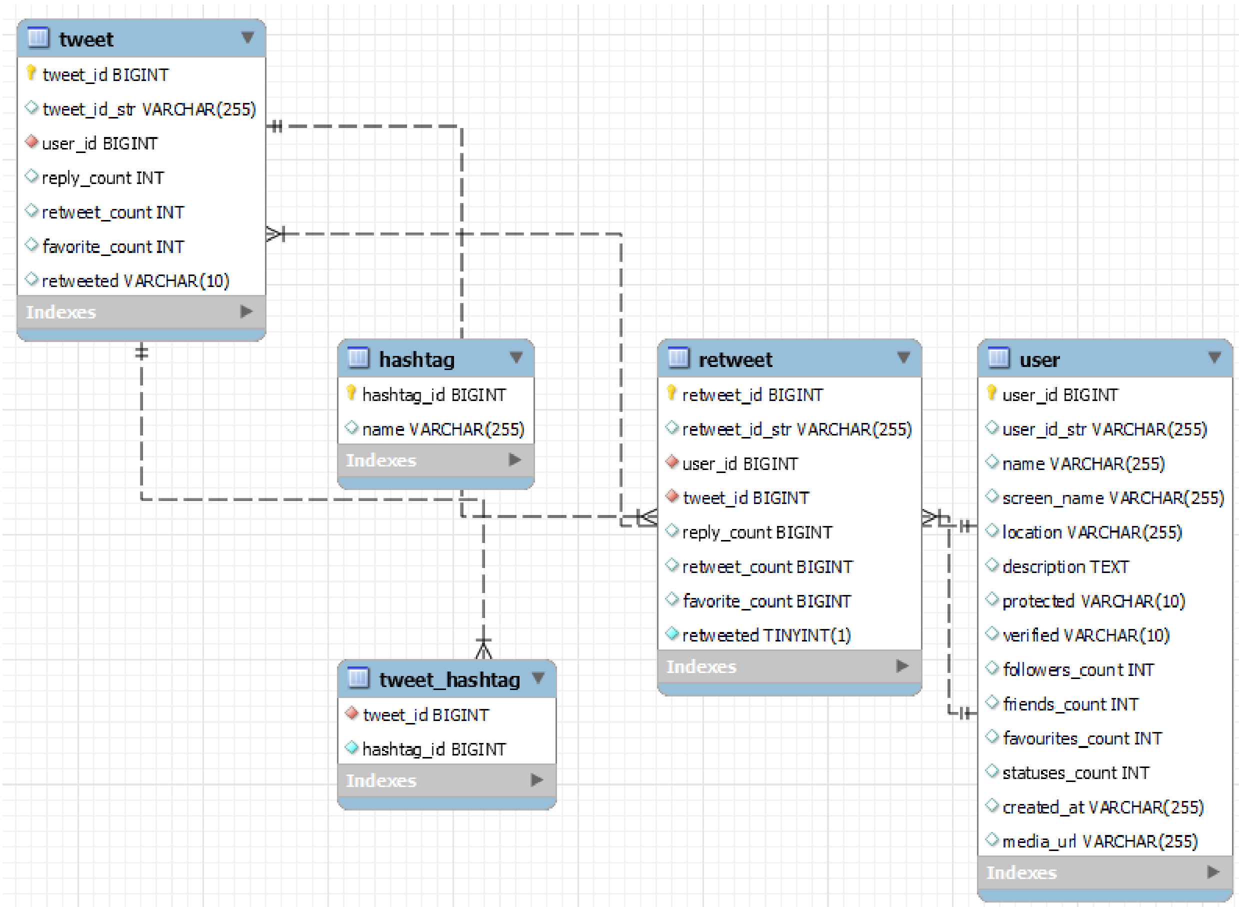



***figure iii*** *– A diagram illustrating the structure and content of each table in the relational database used for this project*

The dataset is inserted into the database one at a time. The JSON dataset is read using the readlines() function and stored as a list of strings, which is used for future insertions. Each element in the data is then iterated through, and useful information is extracted. If the user is not already in the database, it is added to the database. If the text starts with 'RT' (i.e., if it is a retweet), the retweeted_status field is checked, and the user data from the retweeted_status is inserted in the same way the user data is inserted into the database.

Storing user data in a separate SQL database table has multiple advantages. One of the significant benefits is that it optimizes storage space by reducing redundancy as user information is not embedded in every tweet. Additionally, it allows for easy maintenance of user data as modifications can be made in one place, rather than modifying every tweet. Establishing relationships between users based on following or friends can also be easily achieved in the future. Furthermore, it makes searching for user-specific information much more efficient. However, a tradeoff is that the search application has to query the MySQL database every time a tweet is displayed with user information since the data is stored separately from tweet data.

**Insering Tweet and Retweet Information**

Tweet and retweet information, such as tweetid, retweetid, reply count, etc., are stored in two different tables, excluding the text. Storing them separately has several advantages. Since MySQL is a relational database designed for structured data, it's easier to query and analyze the data due to the clear relationship between tweet and retweet data. For complex queries, MySQL's SQL language and advanced indexing capabilities can make it easier to write and more performant.

For the search application, only relevant fields from the User data are selected. A table called 'user' is created in the MySQL database with fields and datatypes shown in the figure. Tweet and retweet information is inserted into the database by iterating through the data file one at a time. If the tweetid is not in the database, it is added. If the text starts with 'RT', it is considered a retweet, and the tweet information is inserted into the retweet table. If the tweet information in retweeted status already exists in the database, it is inserted into the tweet table after cross-checking.

**Insering Hashtag and tweet_hashtag table**

In the Twitter dataset, multiple hashtags exist, and most tweets have repetitive hashtags like #corona. Unique hashtags are stored in the hashtag table with an auto-increment hashtagid. Before inserting the hashtag into the table, the presence of the hashtag is checked, and if it is not present, it is added.

To link hashtags with their respective tweets, another table called tweet hashtag is created. It serves as a composite or junction table, connecting the "tweet" and "hashtag" tables in a many-to-many relationship. It contains two columns, hashtagid and tweetid. Data insertions into both the hashtag and tweet_hashtag tables occur one line at a time.

## Non-Relational Database

In MongoDB, data is stored in collections rather than tables, and each collection consists of multiple documents. In this project, we are using two collections, one for tweet text data and the other for retweet text data.The tweet text data collection will store the actual text content of each tweet, along with some metadata such as the tweet ID and the timestamp when the tweet was created. This collection will be used to support text-based queries, such as searching for tweets containing a particular keyword or phrase.

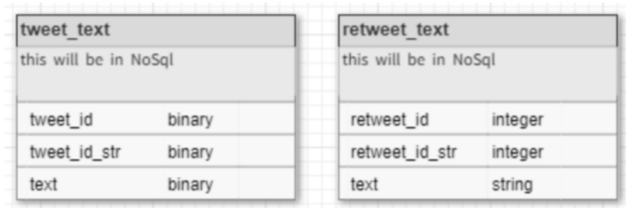



***figure iv*** *– A diagram illustrating the non-relational database tables and its content*

## Implementation of Cache

The cache is a crucial component of the search application, as it helps to improve the performance and reduce the response time of the application. By storing frequently accessed data in memory, the cache reduces the need to fetch data from the MySQL database, which can be time-consuming and resource-intensive.

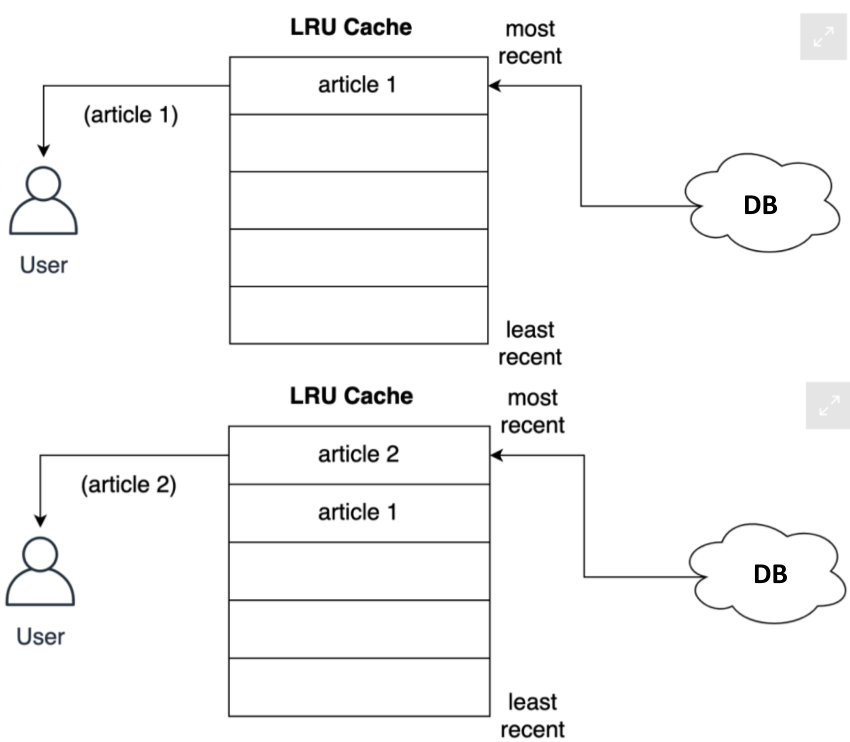



***figure v*** *– A diagram illustrating working principle of LRU*

LRU stands for Least Recently Used and is a cache eviction strategy used in computer systems. In this strategy, the cache is divided into fixed-size blocks, and when the cache is full, the least recently used block is evicted to make space for new entries. This strategy is based on the assumption that the least recently used data is the least likely to be accessed again in the near future.

To prevent data loss, the cache periodically checkpoints its data to disk, allowing it to be reloaded when the application starts up. The Least Recently Used (LRU) strategy is used to evict the least recently used entry if the cache is full.

In the context of the cache implemented using a Python dictionary in our application, LRU is used to ensure that the cache does not exceed its maximum size. When a new item is added to the cache and the maximum size is already reached, the least recently used item is evicted to make room for the new entry. This ensures that the cache is always within its specified limit and that the most recently accessed data is retained in the cache.

## Graphical User Interface – Tkinter

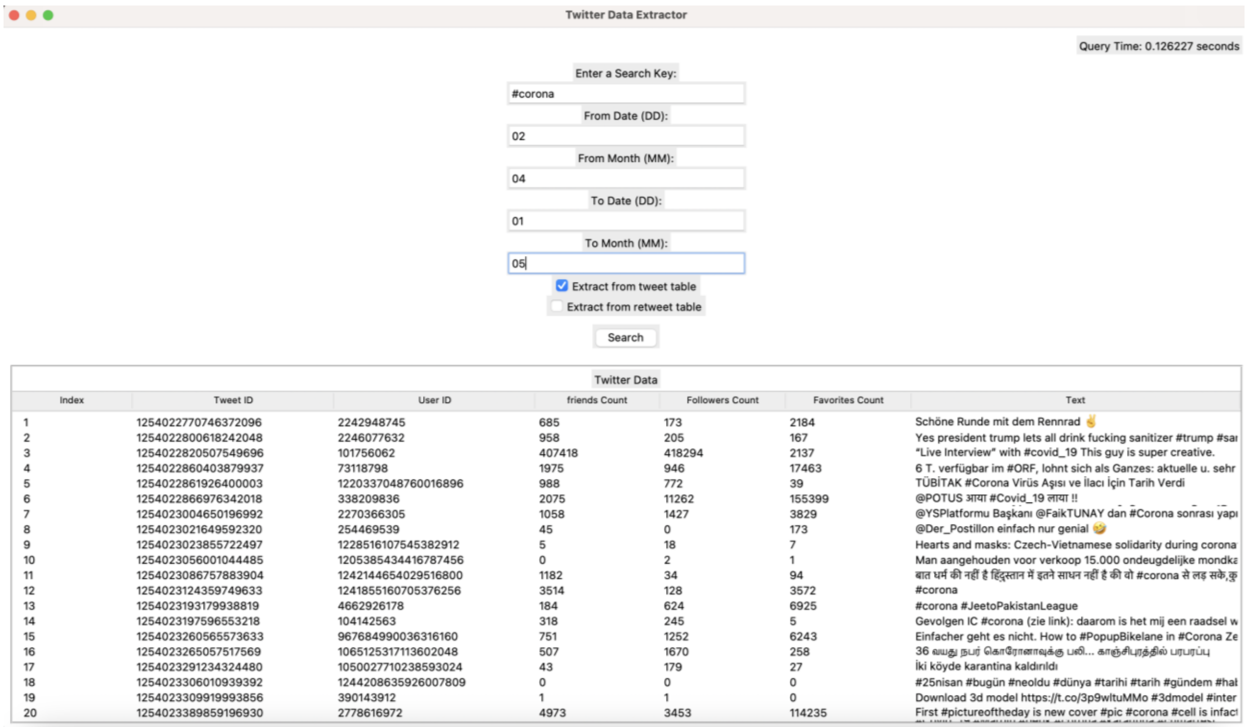
Tkinter, Python's built-in GUI framework, to develop the search interface for the project. Tkinter is a widely adopted and popular toolkit for building desktop applications with Python. It provides an easy-to-use and intuitive way to create graphical user interfaces for Python applications. One of the major benefits of Tkinter is its cross-platform compatibility, which allows the same code to be used on different operating systems such as Windows, Linux, and macOS. This can save time and effort in development, as it eliminates the need to write and maintain separate code for different platforms. Additionally, Tkinter is well-documented and has a large community of users, which can make it easier to find solutions to common problems or get help when needed. The simplicity of Tkinter's API also makes it a good choice for developers who are new to GUI programming, as it has a relatively shallow learning curve compared to some other GUI toolkits.

### Overview of the Twitter Search Application GUI

The search interface for the project is built using Python's Tkinter GUI framework. The GUI opens in a window of size 1920x1080 pixels, providing ample space for users to interact with the application. The interface consists of five search boxes for taking input as search keys, with two search boxes for the start and end dates, and two search boxes for the start and end months. In addition, there is a search box for entering the search term.

Two check boxes are provided that allow users to choose whether to search in the tweet table or the retweet table. This feature gives users more control over their searches, enabling them to search for tweets or retweets specifically. The search results are displayed in a treeview at the bottom of the interface. The treeview provides a clear and concise view of the search results, enabling users to quickly find the information they need.

At the top right corner of the interface, the query time is mentioned, which provides users with information on how long the search took. This feature is particularly useful for users who need to perform multiple searches and want to optimize their search time.

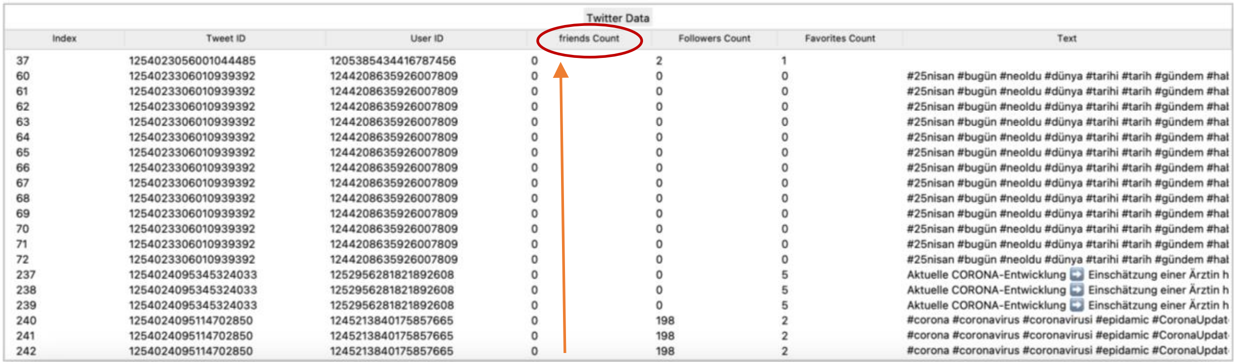



| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 1 | 1254022770746372096 | 2242948745 | 685 | 173 | 2184 | Schöne Runde mit dem Rennrad ✌️ |
| 2 | 1254022800618242048 | 2246077632 | 958 | 205 | 167 | Yes president trump lets all drink fucking sanitizer #trump #sa |
| 3 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 4 | 1254022860403879937 | 73118798 | 1975 | 946 | 17463 | 6 T. verfügbar im #ORF, lohnt sich als Ganzes: aktuelle u. sehr |
| 5 | 1254022861926400003 | 1220337048760016896 | 988 | 772 | 39 | TÜBİTAK #Corona Virüs Aşısı ve İlacı İçin Tarih Verdi |
| 6 | 1254022866976342018 | 338209836 | 2075 | 11262 | 155399 | @POTUS आया #Covid_19 लाया !! |
| 7 | 1254023004650196992 | 2270366305 | 1058 | 1427 | 3829 | @YSPlatformu Başkanı @FaikTUNAY dan #Corona sonrası yap |
| 8 | 1254023021649592320 | 254469539 | 45 | 0 | 173 | @Der_Postillon einfach nur genial 🤣 |
| 9 | 1254023023855722497 | 1228516107545382912 | 5 | 18 | 7 | Hearts and masks: Czech-Vietnamese solidarity during corona |
| 10 | 1254023056001044485 | 1205385434416787456 | 0 | 2 | 1 | Man aangehouden voor verkoop 15.000 ondeugdelijke mondka |
| 11 | 1254023086757883904 | 1242144654029516800 | 1182 | 34 | 94 | बात धर्म की नहीं है हिंदुस्तान में इतने साधन नहीं है की वो #corona से लड़ सके,कु |
| 12 | 1254023124359749633 | 1241855160705376256 | 3514 | 128 | 3572 | #corona |
| 13 | 1254023193179938819 | 4662926178 | 184 | 624 | 6925 | #corona #JeetoPakistanLeague |
| 14 | 1254023197596553218 | 104142563 | 318 | 245 | 5 | Gevolgen IC #corona (zie link): daarom is het mij een raadsel w |
| 15 | 1254023260565573633 | 967684990036316160 | 751 | 1252 | 6243 | Einfacher geht es nicht. How to #PopupBikelane in #Corona Ze |
| 16 | 1254023265057517569 | 1065125317113602048 | 507 | 1670 | 258 | 36 வயது நபர் கொரோனாவுக்கு பலி... காஞ்சிபுரத்தில் பரபரப்பு |
| 17 | 1254023291234324480 | 1050027710238593024 | 43 | 179 | 27 | İki köyde karantina kaldırıldı |
| 18 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #ha |
| 19 | 1254023309919993856 | 390143912 | 1 | 1 | 0 | Download 3d model https://t.co/3p9wltuMMo #3dmodel #inter |
| 20 | 1254023389859196930 | 2778616972 | 4973 | 3453 | 114235 | First #pictureoftheday is new cover #pic #corona #cell is infact |

**Features included**

One of the key features of the search interface is the ability to sort data based on user preferences. When the user clicks on a column name, the data in that column is sorted in either ascending or descending order. Additionally, if the first column is already sorted in descending order and the user clicks on the second column, the data is sorted in either ascending or descending order with the second column taking priority over the first column. This is known as relative sorting and helps users quickly find the data they are looking for.

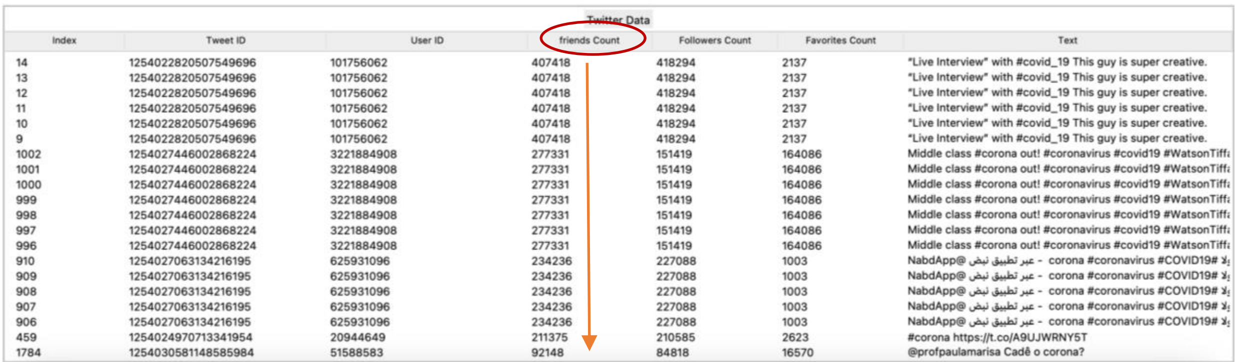
For example, if the user clicks on *friends_count* once, the data is sorted in ascending order.



Twitter Data

| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 37 | 1254023056001044485 | 1205385434416787456 | 0 | 2 | 1 | |
| 60 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 61 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 62 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 63 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 64 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 65 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 66 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 67 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 68 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 69 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 70 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 71 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 72 | 1254023306010939392 | 1244208635926007809 | 0 | 0 | 0 | #25nisan #bugün #neoldu #dünya #tarihi #tarih #gündem #hal |
| 237 | 1254024095345324033 | 1252956281821892608 | 0 | 0 | 5 | Aktuelle CORONA-Entwicklung ➡ Einschätzung einer Ärztin h |
| 238 | 1254024095345324033 | 1252956281821892608 | 0 | 0 | 5 | Aktuelle CORONA-Entwicklung ➡ Einschätzung einer Ärztin h |
| 239 | 1254024095345324033 | 1252956281821892608 | 0 | 0 | 5 | Aktuelle CORONA-Entwicklung ➡ Einschätzung einer Ärztin h |
| 240 | 1254024095114702850 | 1245213840175857665 | 0 | 198 | 2 | #corona #coronavirus #coronavirusi #epidamic #CoronaUpdat |
| 241 | 1254024095114702850 | 1245213840175857665 | 0 | 198 | 2 | #corona #coronavirus #coronavirusi #epidamic #CoronaUpdat |
| 242 | 1254024095114702850 | 1245213840175857665 | 0 | 198 | 2 | #corona #coronavirus #coronavirusi #epidamic #CoronaUpdat |

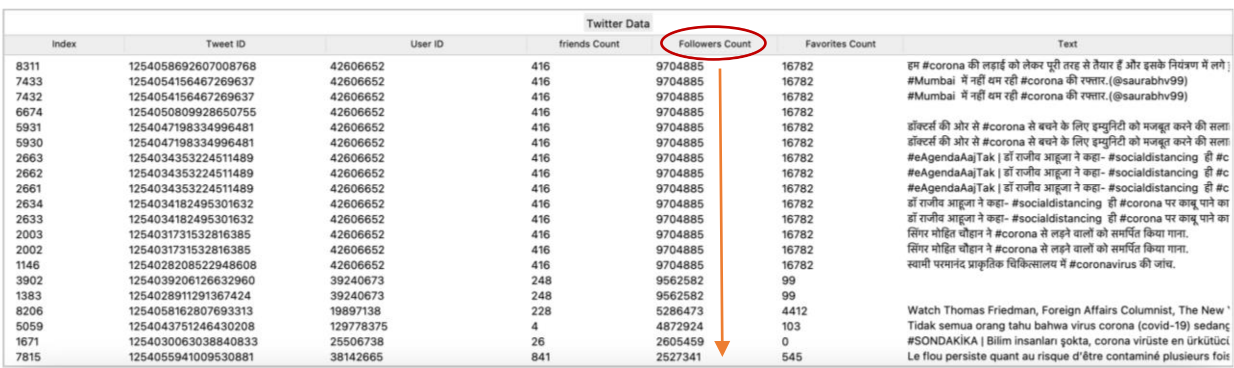
If the user clicks on the *friends_count* for the second time, the data is sorted in descending order.



Twitter Data

| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 14 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 13 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 12 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 11 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 10 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 9 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 1002 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 1001 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 1000 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 999 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 998 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 997 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 996 | 1254027446002868224 | 3221884908 | 277331 | 151419 | 164086 | Middle class #corona out! #coronavirus #covid19 #WatsonTiff |
| 910 | 1254027063134216195 | 625931096 | 234236 | 227088 | 1003 | NabdApp@ عبر تطبيق نبض - corona #coronavirus #COVID19# |
| 909 | 1254027063134216195 | 625931096 | 234236 | 227088 | 1003 | NabdApp@ عبر تطبيق نبض - corona #coronavirus #COVID19# |
| 908 | 1254027063134216195 | 625931096 | 234236 | 227088 | 1003 | NabdApp@ عبر تطبيق نبض - corona #coronavirus #COVID19# |
| 907 | 1254027063134216195 | 625931096 | 234236 | 227088 | 1003 | NabdApp@ عبر تطبيق نبض - corona #coronavirus #COVID19# |
| 906 | 1254027063134216195 | 625931096 | 234236 | 227088 | 1003 | NabdApp@ عبر تطبيق نبض - corona #coronavirus #COVID19# |
| 459 | 1254024970713341954 | 20944649 | 211375 | 210585 | 2623 | #corona https://t.co/A9UJWRNY5T |
| 1784 | 1254030581148585984 | 51588583 | 92148 | 84818 | 16570 | @profpaulamarisa Cadê o corona? |

Further, if the user clicks on *followers_count*, the data is sorted in descending order giving first priority to the selected column.



Twitter Data

| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 8311 | 1254058692607008768 | 42606652 | 416 | 9704885 | 16782 | हम #corona की लड़ाई को लेकर पूरी तरह से तैयार हैं और इसके नियंत्रण में लगे |
| 7433 | 1254054156467269637 | 42606652 | 416 | 9704885 | 16782 | #Mumbai में नहीं थम रही #corona की रफ्तार.(@saurabhv99) |
| 7432 | 1254054156467269637 | 42606652 | 416 | 9704885 | 16782 | #Mumbai में नहीं थम रही #corona की रफ्तार.(@saurabhv99) |
| 6674 | 1254050809928650755 | 42606652 | 416 | 9704885 | 16782 | |
| 5931 | 1254047198334996481 | 42606652 | 416 | 9704885 | 16782 | डॉक्टर्स की ओर से #corona से बचने के लिए इम्युनिटी को मजबूत करने की सला |
| 5930 | 1254047198334996481 | 42606652 | 416 | 9704885 | 16782 | डॉक्टर्स की ओर से #corona से बचने के लिए इम्युनिटी को मजबूत करने की सला |
| 2663 | 1254034353224511489 | 42606652 | 416 | 9704885 | 16782 | #eAgendaAajTak \| डॉ राजीव आहूजा ने कहा- #socialdistancing ही #c |
| 2662 | 1254034353224511489 | 42606652 | 416 | 9704885 | 16782 | #eAgendaAajTak \| डॉ राजीव आहूजा ने कहा- #socialdistancing ही #c |
| 2661 | 1254034353224511489 | 42606652 | 416 | 9704885 | 16782 | #eAgendaAajTak \| डॉ राजीव आहूजा ने कहा- #socialdistancing ही #c |
| 2634 | 1254034182495301632 | 42606652 | 416 | 9704885 | 16782 | डॉ राजीव आहूजा ने कहा- #socialdistancing ही #corona पर काबू पाने का |
| 2633 | 1254034182495301632 | 42606652 | 416 | 9704885 | 16782 | डॉ राजीव आहूजा ने कहा- #socialdistancing ही #corona पर काबू पाने का |
| 2003 | 1254031731532816385 | 42606652 | 416 | 9704885 | 16782 | सिंगर मोहित चौहान ने #corona से लड़ने वालों को समर्पित किया गाना. |
| 2002 | 1254031731532816385 | 42606652 | 416 | 9704885 | 16782 | सिंगर मोहित चौहान ने #corona से लड़ने वालों को समर्पित किया गाना. |
| 1146 | 1254028208522948608 | 42606652 | 416 | 9704885 | 16782 | स्वामी परमानंद प्राकृतिक चिकित्सालय में #coronavirus की जांच. |
| 3902 | 1254039206126632960 | 39240673 | 248 | 9562582 | 99 | |
| 1383 | 1254028911291367424 | 39240673 | 248 | 9562582 | 99 | |
| 8206 | 1254058162807693313 | 19897138 | 228 | 5286473 | 4412 | Watch Thomas Friedman, Foreign Affairs Columnist, The New ' |
| 5059 | 1254043751246430208 | 129778375 | 4 | 4872924 | 103 | Tidak semua orang tahu bahwa virus corona (covid-19) sedang |
| 1671 | 1254030063038840833 | 25506738 | 26 | 2605459 | 0 | #SONDAKİKA \| Bilim insanları şokta, corona virüste en ürkütüc |
| 7815 | 1254055941009530881 | 38142665 | 841 | 2527341 | 545 | Le flou persiste quant au risque d'être contaminé plusieurs fois |

This search interface provides a powerful and intuitive way to search and analyze Twitter data. With its advanced sorting and filtering capabilities, users can quickly and easily find the data they need, while the detailed information view allows them to dive deeper into individual data points.

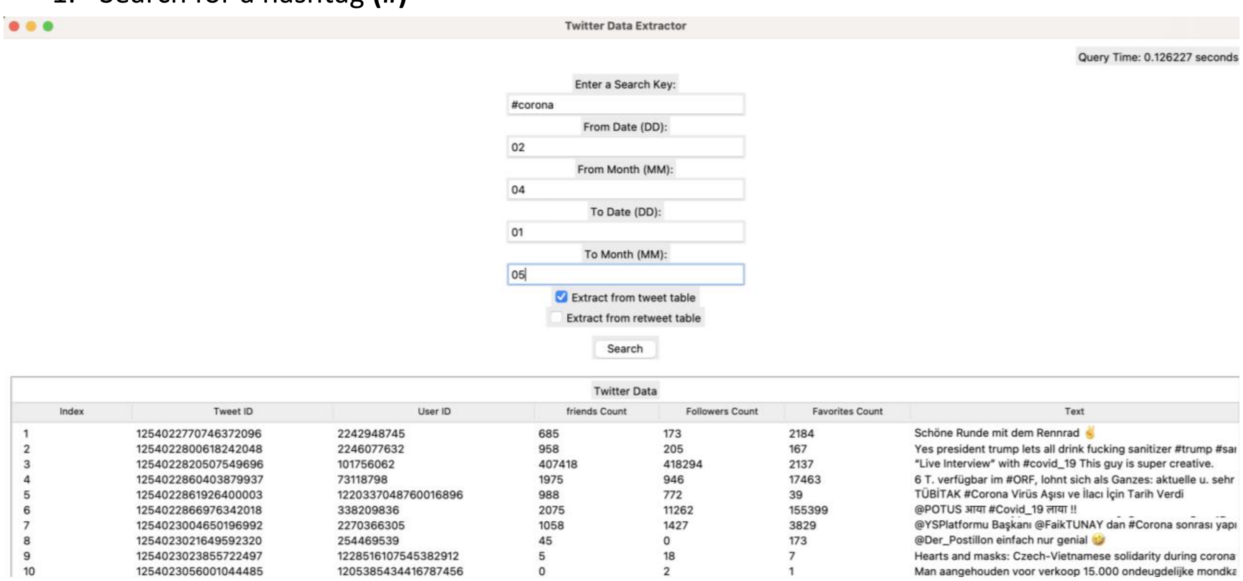
Another important feature is the ability to view detailed information about a particular row of data. When the user clicks on a row, a new window pops up displaying additional details about the user, such as their profile picture, bio, and the number of tweets or retweets they have posted. This feature allows users to quickly get more information about a particular user without having to perform additional searches or queries. (refer to the following screenshot)


| | |
|---|---|
| Screen Name | tellfabian |
| Location | Quebec |
| Description | sci-fi, |
| Followers Count | 205 |
| Friends Count | 958 |
| Text | Yes president trump lets all drink sanitizer #trump #sanitizer #lol #corona |

### Types of searches allowed

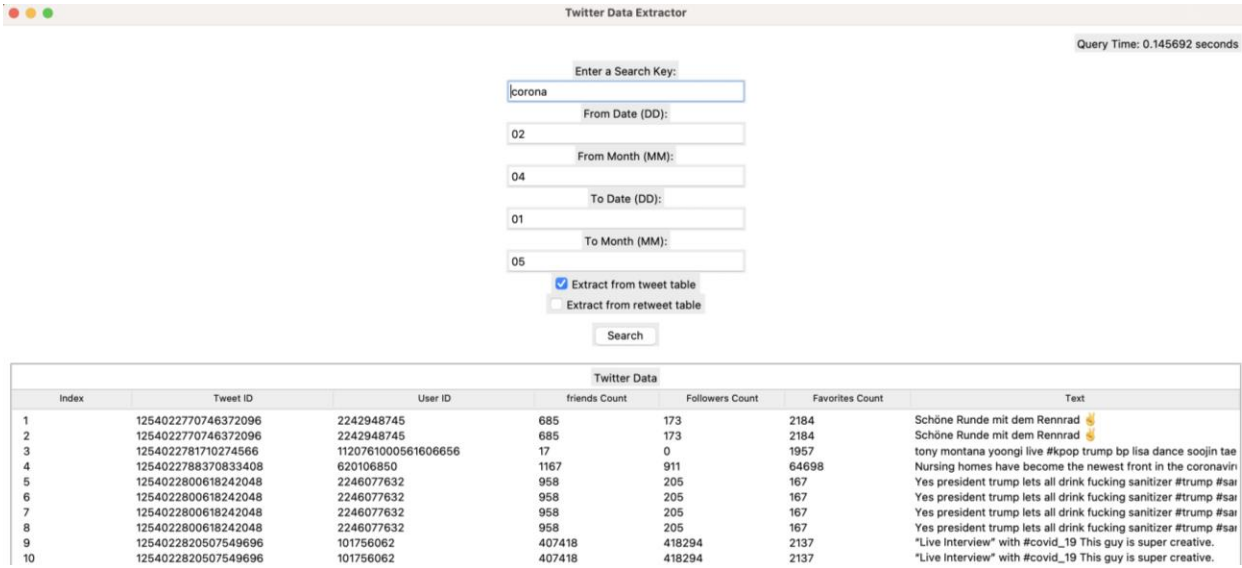
The search application allows three types of searches. If the user inputs a search key starting with the # symbol, the search application interprets it as a hashtag and searches for tweets or retweets containing that particular hashtag. Similarly, if the user inputs a search key starting with the @ symbol, the search application interprets it as a screen name input and searches for user names containing the given string. If the user inputs just the string without any symbol, the search application takes it as an input string and performs searches in the tweets or retweets table containing that string in the text available. These search types provide flexibility to the user to search for the desired data based on specific criteria.

1. Search for a hashtag **(#)**



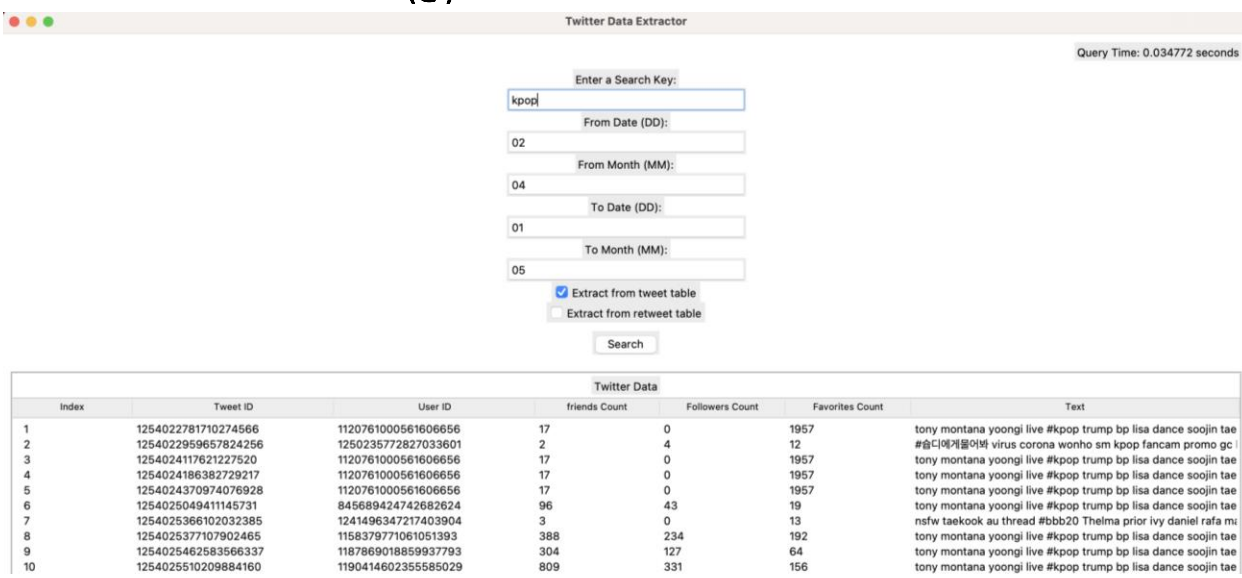

| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 1 | 1254022770746372096 | 2242948745 | 685 | 173 | 2184 | Schöne Runde mit dem Rennrad ✌️ |
| 2 | 1254022800618242048 | 2246077632 | 958 | 205 | 167 | Yes president trump lets all drink fucking sanitizer #trump #sai |
| 3 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 4 | 1254022860403879937 | 73118798 | 1975 | 946 | 17463 | 6 T. verfügbar im #ORF, lohnt sich als Ganzes: aktuelle u. sehr |
| 5 | 1254022861926400003 | 1220337048760016896 | 988 | 772 | 39 | TÜBİTAK #Corona Virüs Aşısı ve İlacı İçin Tarih Verdi |
| 6 | 1254022866976342018 | 338209836 | 2075 | 11262 | 155399 | @POTUS आया #Covid_19 लाया !! |
| 7 | 1254023004650196992 | 2270366305 | 1058 | 1427 | 3829 | @YSPlatformu Başkanı @FaikTUNAY dan #Corona sonrası yapı |
| 8 | 1254023021649592320 | 254469539 | 45 | 0 | 173 | @Der_Postillon einfach nur genial 🤪 |
| 9 | 1254023023855722497 | 1228516107545382912 | 5 | 18 | 7 | Hearts and masks: Czech-Vietnamese solidarity during corona |
| 10 | 1254023056001044485 | 1205385434416787456 | 0 | 2 | 1 | Man aangehouden voor verkoop 15.000 ondeugdelijke mondka |

2. Search for a string




| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 1 | 1254022770746372096 | 2242948745 | 685 | 173 | 2184 | Schöne Runde mit dem Rennrad ✌ |
| 2 | 1254022770746372096 | 2242948745 | 685 | 173 | 2184 | Schöne Runde mit dem Rennrad ✌ |
| 3 | 1254022781710274566 | 1120761000561606656 | 17 | 0 | 1957 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 4 | 1254022788370833408 | 620106850 | 1167 | 911 | 64698 | Nursing homes have become the newest front in the coronavir |
| 5 | 1254022800618242048 | 2246077632 | 958 | 205 | 167 | Yes president trump lets all drink fucking sanitizer #trump #sar |
| 6 | 1254022800618242048 | 2246077632 | 958 | 205 | 167 | Yes president trump lets all drink fucking sanitizer #trump #sar |
| 7 | 1254022800618242048 | 2246077632 | 958 | 205 | 167 | Yes president trump lets all drink fucking sanitizer #trump #sar |
| 8 | 1254022800618242048 | 2246077632 | 958 | 205 | 167 | Yes president trump lets all drink fucking sanitizer #trump #sar |
| 9 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |
| 10 | 1254022820507549696 | 101756062 | 407418 | 418294 | 2137 | "Live Interview" with #covid_19 This guy is super creative. |

3. Search for a user name **(@)**



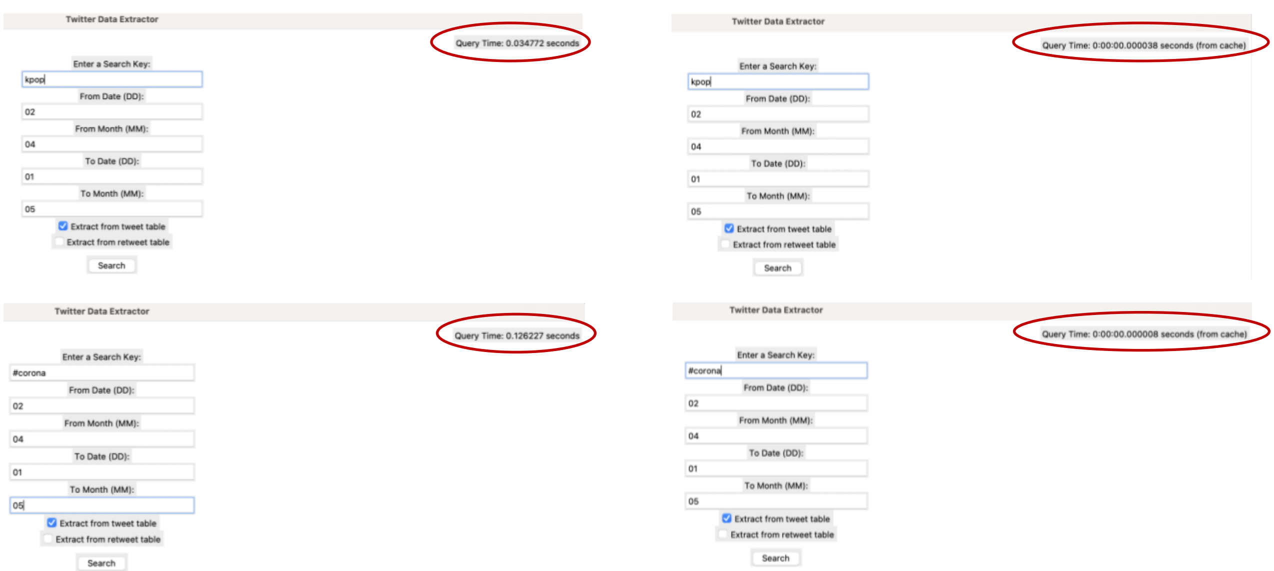

| Index | Tweet ID | User ID | friends Count | Followers Count | Favorites Count | Text |
|---|---|---|---|---|---|---|
| 1 | 1254022781710274566 | 1120761000561606656 | 17 | 0 | 1957 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 2 | 1254022959657824256 | 1250235772827033601 | 2 | 4 | 12 | #숩디에게물어봐 virus corona wonho sm kpop fancam promo gc |
| 3 | 1254024117621227520 | 1120761000561606656 | 17 | 0 | 1957 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 4 | 1254024186382729217 | 1120761000561606656 | 17 | 0 | 1957 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 5 | 1254024370974076928 | 1120761000561606656 | 17 | 0 | 1957 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 6 | 1254025049411145731 | 845689424742682624 | 96 | 43 | 19 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 7 | 1254025366102032385 | 1241496347217403904 | 3 | 0 | 13 | nsfw taekook au thread #bbb20 Thelma prior ivy daniel rafa ma |
| 8 | 1254025377107902465 | 1158379771061051393 | 388 | 234 | 192 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 9 | 1254025462583566337 | 1187869018859937793 | 304 | 127 | 64 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |
| 10 | 1254025510209884160 | 1190414602355585029 | 809 | 331 | 156 | tony montana yoongi live #kpop trump bp lisa dance soojin tae |

## Query time optimization

The cache is a crucial component of the search application, as it helps to improve the performance and reduce the response time of the application. By storing frequently accessed data in memory, the cache reduces the need to fetch data from the MySQL database, which can be time-consuming and resource-intensive. LRU stands for Least Recently Used and is a cache eviction strategy used in computer systems. In this strategy, the cache is divided into fixed-size blocks, and when the cache is full, the least recently used block is evicted to make space for new entries. This strategy is based on the assumption that the least recently used data is the least likely to be accessed again in the near future.

To prevent data loss, the cache periodically checkpoints its data to disk, allowing it to be reloaded when the application starts up. The Least Recently Used (LRU) strategy is used to evict the least recently used entry if the cache is full. In the context of the cache implemented using a Python dictionary in our application, LRU is used to ensure that the cache does not exceed its maximum size. When a new item is added to the cache and the maximum size is already reached, the least recently used item is evicted to make room for the new entry. This ensures that the cache is always within its specified limit and that the most recently accessed data is retained in the cache.

**An example of the implementation of the cache is as follows:**
Suppose a user searches for tweets containing the hashtag **"#corona", "corona"** and **"@kpop"** multiple times within a short time period. Without a cache, the search application would need to query the database for each search, resulting in multiple database queries and slower response times. However, with the cache implemented, the first search result would be stored in the cache with the hashtag "#corona" as the key and the corresponding tweets as the value. The next time the user searches for tweets containing the same hashtag, the application would first check the cache to see if the results are already available. If they are, the application would retrieve the results from the cache instead of querying the database, resulting in faster response times. The cache would continue to store the search results until the TTL value for the cache entry expires, at which point it would be evicted from the cache. This implementation not only improves the performance of the search application but also reduces the number of database queries, making it more efficient.



## Conclusion

The success of the project can be attributed to the effective use of various technologies and techniques for efficient data handling and processing. The team's use of a relational database management system, MySQL, allowed for efficient storage, retrieval, and management of large amounts of data. The implementation of a cache using Python's dictionary data structure further improved the performance of the system by reducing the response time for frequently accessed data. The use of a GUI framework, Tkinter, provided a user-friendly interface that enhanced the overall user experience.

The project's success also emphasizes the importance of proper design and planning to achieve a robust and scalable system. The team's use of a layered architecture and modular design approach allowed for easy maintenance, testing, and scalability of the system. The team's focus on designing a system that can handle large amounts of data without compromising on performance or user experience was critical to the project's success.

**The following are the milestones achieved in the project development process:**

- Conducted a thorough analysis of the dataset and identified the relevant parameters to be considered for the search query.
- Created a MySQL database with appropriately named tables to store the data.
- Successfully implemented the insertion of data into the MySQL database using Python.
- Created a MongoDB database with appropriately named collections to store the data.
- Successfully implemented the insertion of data into the MongoDB database using Python.
- Successfully designed a working cache to store the data after filtering it, to control the data flow efficiently.
- Designed the visual structure of the user interface and developed the programming part to implement the search query and display the results on the UI.
- The number of records that has been inserted into the relational data base system, contains User table – 88063, Tweet table – 87865, Retweets table – 14841 and Hashtags – 44628.

## Distribution of Work among team members

Dileep Kumar Pothala – Relational Database

Manish Maddimsetty – Cache Storage

Venkata Sai Charan Kornu – Non Relational Database

Nemalidinne Kiran Reddy – Search Application design & reports