Design Document: "Mini Project 2 (group project)"

1. General Overview

   This Python script is an essential component of the Tweet System, a comprehensive application for user authentication, tweet management, and database interactions. Initially, a companion script, load-json.py, efficiently loads JSON data into MongoDB, showcasing functionalities for connecting, creating/accessing databases and collections, dropping an existing 'tweets' collection, and batch-loading JSON data.Thereafter , users can log in, create accounts, compose tweets, and perform various actions, facilitated by modules such as compose_tweet, list_top_tweets, list_top_users, search_users, and search_tweets.

2. User Guide:

   First we start processing the database using load-json.py and get the data sorted.

   Start by executing the main script using the command: python main.py <port_number>. Upon successful execution, users can interact with the system through a menu offering options such as searching tweets, users, listing top tweets and users, composing tweets, and exiting. When searching for tweets or users, enter the keywords or criteria when prompted. When listing the top tweets or users, enter the number of top tweets or users to display. When composing a tweet, enter the content of the tweet. You can quit the search or listing at any time by entering ‘q’.

3. Technologies Used:
   The project is developed using Python and MongoDB queries, leveraging the pymongo library. MongoDB compass is utilized as the database for efficient storage and retrieval of data.

4. Software Design:
   The software design of the system is modular, with each functionality implemented in a separate Python module. Here’s a detailed description of each module:

   - **Search Tweets**: Users can search for tweets based on specific keywords.
   - **Search Users**: Users can search for other users based on specific criteria.
   - **List Top Tweets**: The system can display the top tweets based on certain metrics.
   - **List Top Users**: The system can display the top users based on certain metrics.
   - **Compose a Tweet**: Users can compose and post a new tweet.

5. Algorithms

   The system uses MongoDB’s built-in operations for searching, inserting, and retrieving data. **It also creates indexes on the tweet data to speed up searches**. The specific algorithms for each functionality are implemented in separate Python files (search_tweets.py, search_users.py, list_top_tweets.py, list_top_users.py, compose_tweet.py).
6. Testing Strategy

   Unit Testing
   - Each component of the system, such as the search tweets, search users, list top tweets, list top users, and compose tweet functions, were individually tested to ensure their correctness. This

involved running individual parts of each function and checking the output for accuracy and proper formatting.

Integration Testing

- The integration of session handling with database interactions was carefully tested to ensure the system functions seamlessly as a whole.

Scenarios Tested

- Several scenarios were tested to ensure the robustness of the system, including:
- Searching for tweets with single and multiple keywords.
- Searching for users based on specific criteria.
- Listing the top tweets based on retweet count, like count, or quote count.
- Listing the top users based on follower count.
- Composing and posting a new tweet with different content lengths.

Test Case Coverage

- Test cases were designed to cover a wide range of scenarios for each major function, ensuring comprehensive testing of the system. Special emphasis was placed on boundary conditions and erroneous inputs for session and database interactions.

Bug Statistics

- No major bugs were found during the testing process. Any minor issues identified in edge cases such as multiple keywords in searching functions and taking port numbers in the terminal etc. were promptly addressed, ensuring the robust functionality of the system.

7. Group Work Breakdown

Work Distribution:

- Pranav: Handled search_users.py, search_tweets.py, managed testing
- Amaan: Handled load.json and search_users.py, list_top_users.py, documentation inputs, testing
- Ashwin: Handled list_top_users, list_top_tweets, managed testing, fixed bugs in nearly all functions.
- Rishi: Managed list_top_tweets.py and compose_tweets.py., documentation

Time Allocation:

- Each member spent an estimated 22-24 hours on their respective functionalities and testing procedures.

Progress Updates:

- Progress updates were shared among team members approximately every day to track work completion and resolve any issues.

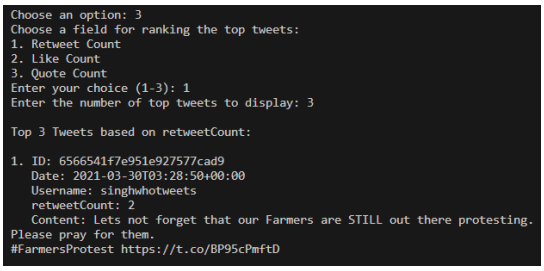
Coordination Method:

- Utilized GitHub features for sharing progress, keeping track of updates through changes committed, VSCode live sharing and Google Collab.

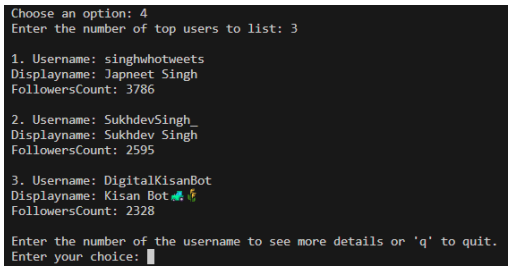
8. Assumptions:

- MongoDB Connection: The system assumes that a MongoDB instance is running on ‘localhost’ and can be accessed using the provided port number.
- Database and Collection: The system assumes that the tweet and user data are stored in a collection named ‘tweets’ in the database ‘291db’.
- Keyword Matching: The system assumes that multiple keywords in a tweet are separated by spaces or punctuations.
- User Identity: When composing a tweet, the system assumes the username to be “291user”.

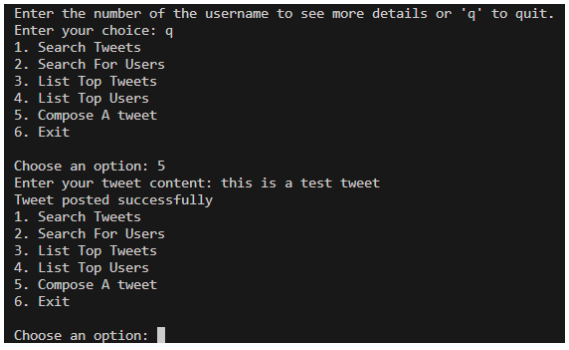
9. Limitations:
    - User Authentication: The system currently does not include functionalities for user authentication and account creation. Therefore, it does not support multiple users or personalized user experiences.
    - Session Management: The system does not maintain state between different runs of the main.py script. Each run of the script constitutes a separate session.
    - Case Insensitivity: The system assumes case-insensitive matching for keywords. This might not be desirable in all scenarios.
    - Limited Functionalities: The system provides a limited set of functionalities (searching for tweets/users, listing top tweets/users, and composing a tweet). Other potential functionalities like updating or deleting tweets, following or unfollowing users, etc., are not supported.

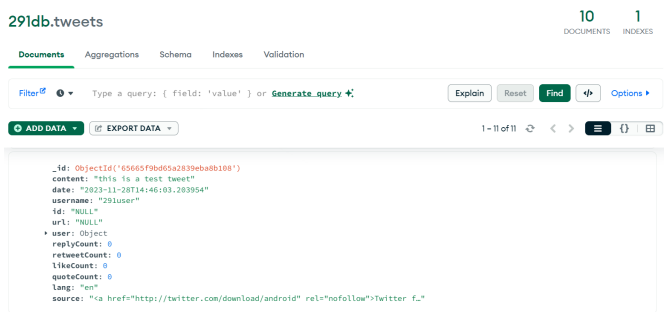


"Implementation of list_top_tweets: a user is given 3 options, retweet count like count and quote count all cases have been tested and work. An example of retweet count have been given where we asked to give 3 top tweets and as shown in the example the first output is shown in the image"



"Implementation of list_top_users: a user is given a choice, to display the n users where the user chooses the value of n, all cases have been tested and work. An example has been given where we asked to give 3 top users and the first output has been shown in the image, additionally there is a nested input asking the user to see more details of a specific user."





"Implementation of compose_tweets, where a user is asked for an input to post a tweet to the database. The tweet content is recorded with the url and precise timestamp."