General Overview

The program takes 2 additional arguments to run, the first being the .json file being used and the second being the port number. The user is then presented with 6 options, selectable by typing a number from 1 to 6: 1 is searching for tweets, 2 is searching for users, 3 is listing the top tweets, 4 is listing the top users, 5 is composing a new tweet, and 6 is closing the program. Options 1-5 call a function from its respective file in the folder.

If the user chooses to search for tweets, they are asked to provide their search keywords separated by commas. The program then returns a maximum of 5 tweets which contain that keyword with their id, the username of the poster, and the content of the tweet. The user is then asked if they want to see more fields of a tweet, then asked which tweet to select based on their order in the menu. Then, they can specify which field they want to see, repeating until they choose to not view any more fields, sending them back to the prompt for if they want to view more fields of a tweet. If the user does not want to view any more tweets, the user will be sent back to the first menu.

If the user chooses to search for other users, they are then asked to provide one search keyword. If the keyword is “exit”, the program will return back to the first menu. All users which match that keyword will be displayed, and the user is asked to either select a user based on their order in the list or type 0 to try another search keyword. If the user selects another user, more details of that user will be displayed, then the user will be sent back to the searching prompt.

If the user chooses to list the top tweets, they are then asked to choose whether they would like the list to be sorted by retweet count, quote count, or like count by typing a number from 1 to 3. They are then asked how many of the top tweets they would like to see. Then, that number of tweets are displayed, sorted as per the user’s request, with their id, date, content, and poster username. Tweets can be selected by their order in the list, which will show all of the tweet fields, and the user can type “x” to go back to the first menu.

If the user chooses to compose a new tweet, they are prompted for the tweet content, after which the tweet will be posted and added to the database, and the user will be sent back to the first menu.

Algorithm Details

Building document store

- Takes in json file and port number as command line arguments
- creates a database called ‘291db’
- if there is a collection of tweets already existing, then drop it.
- create a new tweet collection.
- loads batches of tweets in size of 1000 using for loop.

Search tweets

- Takes keywords separated by commas
- Iterates through all fetched documents
- Compares entire list of keywords and checks if they lie inside the document content
- Lower() used for making data case-insensitive
- Adds accepted document to 2D array which is used to print the tweets
- Offers availability to see any chosen field of tweets from all possible fields
- Allows this feature until user wishes to exit
- Accounts for invalid inputs

Search users

- Prompts the user to input a keyword for searching users or type “exit” to quit.
- Searches the tweets collection for users whose display name or location matches the keyword (case-insensitive)—group results to remove duplicates.
- It shows a numbered list of matching users (username, display name, location). If there are no matches, inform the user.
- It lets the user pick a user by number to view full details or return to the search.
- Loops the process until the user types “exit”

List top tweets

- Prompt the user to input the sorting type. Loop until they select a valid type.
- Prompt the user on how many tweets they want to see
- Shows the tweet information
- Ask the user if they want to see all fields of a tweet. Loop until they press ‘x’

List top users

- Creates a dictionary of dictionaries of users with no duplicates.
- sorts the dictionary according to the count of followers.
- if n is a valid positive number or not, prints the n top users
- else print error message
- while loop runs until user types ‘n’
- if user types ‘y’, checks if username is found in tweets using a boolean flag ‘found’,
- if found, prints out full information of users
- if not found, print “There is no such user” message.

Compose tweet

- Obtains the current date and time and converts it to ISO 8641 format
- Collects input from the user for the “content” field of the tweet
- Inserts a new tweet into the database with all values null except for current date, content, and user as “291user”

Testing Strategy

For functions 1 and 2, we tested with keywords with varying cases to test the case-insensitivity. Based on the usernames available and content of tweets, we tested specific terms that would trigger only certain test cases. For example, for function 1, we tested keywords with special characters like underscore (e.g. “Desh_Ka_Gaddar_RSS”), apostrophes (e.g. “farmer’s”) and more that are important in distinguishing tweets.

For function 3, we tested with both json files, made sure that less than 1 tweet selected works, and made sure it sorted properly with every sorting type.

For function 4 list top users, we tested with the given 10.json and 100.json files, added in extra rows to account for if there are duplicate users with different information. Added a count variable just to test whether we are getting the correct number of top users.

For function 5, we created test tweets, then used the search tweet function to test if all fields were added correctly.

Source Code Quality

Our program uses separate files with their own functions to help separate each part of the program. Additionally, we have explanatory comments throughout the code to assist with readability, and our variable names are descriptive enough to avoid confusion.