CMPUT 291 Mini Project 2 Report

a) General Overview of the System with a Small User Guide

TweetInsightDB we designed a MongoDB-based application for managing and querying tweet data. Thus, system contains two main components: Phase 1: for building the document store and Phase 2: for operating on it.

User Guide:

Phase 1 (load-json.py): This will load JSON data into MongoDB. Run load-json.py <json_file> <mongo_port> to create a 291db database and a tweets collection.

Phase 2 (mongo_operations.py): If you want to operate on the database. Run mongo_operations.py <mongo_port> to access features like searching tweets/users, listing top tweets/users, and composing tweets.

b) Detailed Design of Software (major functions and responsibility):

Phase 1: load-json.py

Loads JSON data into MongoDB.

We used the batch insertion for time efficiency.

We created necessary indexes for optimized search in Phase 2.

connect_to_mongo(port): Establishes connection to MongoDB.

load_json_to_mongodb(json_file, db): Loads data from a JSON file to MongoDB in batches, creating indexes for efficient searches.

mongo_operations.py

Phase 2: Tweets Operations

Main Menu (mongo_operations.py): Facilitates user navigation.

- main_menu (db): Provides the main interface for user interaction, routing to various functionalities.

Tweets (tweets_operations.py): Uses regex for keyword matching.

- search_tweets (db, keywords): Implements search functionality for tweets using regex for keyword matching.
- show_tweet_details (db, tweet_id) (show tweet details): Fetches and displays detailed information of a selected tweet.
- list_top_tweets (db): Lists top tweets based on user-specified criteria retweetCount, likeCount, and quoteCount.

- compose_tweet (db): Allows users to compose and insert a new tweet with the current date.

Users (users_operations.py): Searches based on display names or locations.

- search_users (db, keyword): Implements user search functionality using keyword matching similar to search_tweets.
- list_top_users (db): Lists top users based on follower's count, retweet count, like count
- show_user_details (db, keyword) (Show User Details): Displays detailed information for a selected user.

c) Testing Strategy

Search Functionality: We Tested with various keywords, blank inputs to make sure case-insensitive matching and correct retrieval.

Listing Top Entities: We verified accuracy with predefined datasets, ensuring correct sorting and duplicate handling.

Tweet Composition: We tested insertion function and made sure it only shows correct results. We also made sure all operations met the time constraints specified in the rubric.

Edge-Cases Testing: Test 'Enter' or use whitespace to ensure nothing will show up and test emoji cases.

d) Group Work Break-Down Strategy

Falak Sethi (fsethi): Worked on Phase 1 and Phase 2 development, JSON loading, indexing, user and tweets operations, main menu integration. Approximately 20 hours spent.

Suhansh Patel (suhanshk): Contributed to Phase 1 and Phase 2 development based on other group members' work, primarily on user and tweets search functionalities. Contributed around 20 hours.

Aaryan Singh (ajsingh): Worked on debugging and solving issues for testing. Focused on composing ad storing of tweets in database. Solved the issue for search term matching different fields. Involved for about 20 hours.

Zhiyuan Li (zhiyua15): Documented the design report, LLM file and pushed initial work for compose tweet feature. Also worked on listing top users. Figure out the issues by testing the final outcomes. Dedicated around 20 hours.

Coordination: It was done via **weekly in-person meeting. Google meet and Discord meetings** if needed. As well as **GitHub for version control** and task tracking.

e) Assumptions and Limitations

Assumptions:

1. Our program has only English version but can search up in other languages (even emoji) by keyword matching.
2. Our data size management should not be large that takes time for loading and queries.

Limitations:

1. Our search functionality may not be able to handle extremely large datasets beyond the scope of the project's test cases but we still did our best to prevent that.
2. The application is command-line-based, and also no encapsulations, which needs our user guide to make it works for non-technical users or clients.

f) Code Quality

Readability: We have a clear name of responsibility for each file and consistent code structure.

Maintenance: Functions are well-organized that can be easy to maintain.

Error Handling: Robust error handling in place to manage exceptions and user input errors.

Overall, our project, "TweetInsightDB", is designed to manage and query large-scale tweet data, which shows a practical application of MongoDB in processing and analyzing social media data.