CMPUT 291 Mini Project 1 – Fall 2024

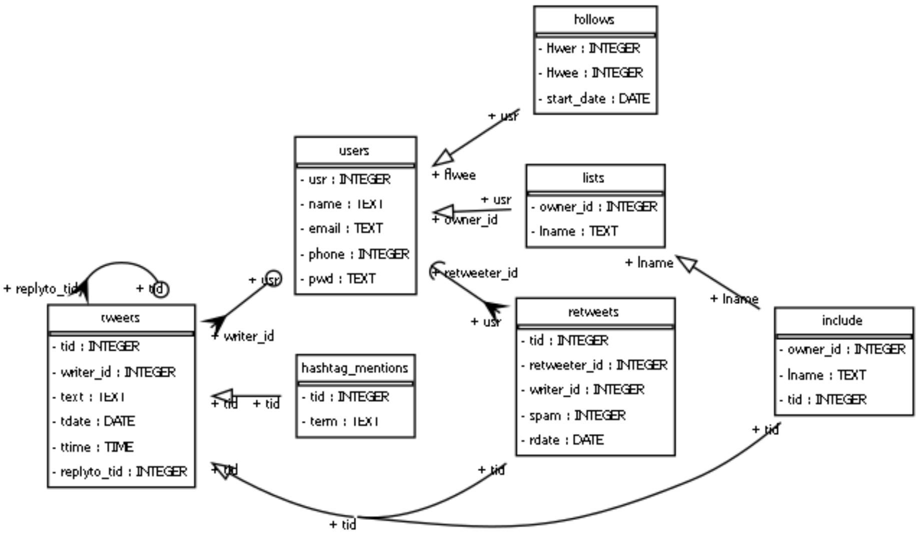
Team Member: Darlene Nguyen, Khanh Bui, Norman Wong, Lucy Jing

DESIGN DOCUMENT

(a) a general overview of your system with a small user guide

We run the program with the command line `python3 main.py` on the lab machine. Once the program runs, we prompt the program to ask if the user has had an account in the system. If the user is new to the system, we prompt the user to enter all of their info, in order to create an account. After logging in, the user can get access to all functionalities that we have in our system by entering what they decide to do (`'search tweet', 'search user', 'tweet', 'reply', 'retweet', 'list followers', 'log out', 'exit'`). Once an action word is entered, the functionality corresponding to that input will run. After the result is display/ the tweet is successfully posted, the system is prompted to ask user what they are want to do once again. The program will eventually stop after it receive the input of `'exit'` or `CTRL+C` on the lab machine.

(b) a detailed design of your software with a focus on the components required to deliver the major functions of your application



1. Login Screen:

- Purpose: The LogInPage function serves as the primary entry point for user interaction, handling both login and registration processes. It prompts the user to either log in or register, validates their input, and interacts with the database to perform necessary operations.
- Return Value: Returns the user ID and login state.
- How it works:
    - LogInPage: Prompts the user to choose between login and registration. Checks login against the database. If user has a valid login, returns user id and sets the login_state flag to True if login is successful. If user needs a new account, function will ask for name, email, phone number, and password. The system assigns a unique

user ID to each new user. Inserts new user information into the users table. Sets the login_state flag to True after successful registration. Calls call_db to establish a database connection.

  - call_db: Establishes a connection to the SQLite database test.db. Returns the connection and cursor objects.
  - get_user_length: Queries the users table to determine the number of existing users. Returns the count of users.
  - show_users: Queries the users table to retrieve all user information. Returns the fetched data.

2. Search for tweets

- Purpose: Enables users to search the SQLite database for tweets using keywords, including hashtags.
- How it Works:
  - Accepts a keyword input and queries the tweets and hashtag_mentions tables to find matching results.
  - Differentiates between hashtag searches (e.g., #keyword) and regular text searches, providing relevant results based on the input.
- Testing:
  - Run main.py and use the command line input "search tweet" to initiate the search feature.
  - Enter keywords (including hashtags) to verify that the search returns accurate tweet results.

3. Search for users

- Purpose: search_user: This function takes a keyword as input and searches the users table for matching usernames. It displays the search results and allows the user to view more results or select a specific user for detailed information.
- Parameters: connection: An active SQLite database connection. cursor: A cursor object associated with the database connection. keyword: The keyword to search for in usernames. current_user_id: The ID of the currently logged-in user. offset: The starting offset for pagination of search results or tweets. limit: The number of results or tweets to display per page.
- Return Value: Both functions do not return a value. They just interact with the user and the database to display information and perform actions.
- How it Works:
  - search_user: Executes an SQL query to search for usernames matching the keyword. Displays the search results, paginated with a limit of 5 results per page. Prompts the user to view more results or select a user. If the user chooses to view more users, calls itself recursively with an increased offset. If a user is selected, calls the show_user_info function to display details.
  - show_user_info: Executes SQL queries to retrieve the user's tweet count, following count, follower count, and recent tweets. Displays the user's information, including recent tweets. Prompts the user to follow the user, view more tweets, or exit. If the user chooses to follow, inserts a new record into the follows table. If the user chooses to view more tweets, calls itself recursively with an increased offset.

4. Compose a tweet

- Purpose: Prompt to let user post a tweet, retweet, or reply to another tweet
- Return: True always
- There are two functions: compose_tweet(connection, cursor, user_id, replyto_tid = None) and compose_retweet(connection, cursor, user_id, tid)
  - compose_tweet(connection, cursor, user_id, replyto_tid = None):
    - If it is a tweet, the replyto_id will be defaulted to be None

        - Creates a new tweet or a reply, ensuring unique hashtags.
        - Gets tweet text and validates hashtags.
        - Records unique hashtags in hashtag_mentions.
        - Handle input error such as invalid tweet or duplicate hashtags.
    - compose_retweet(connection, cursor, user_id, tid):
        - Allows users to retweet an existing tweet.
        - Confirms retweet action and checks for spam.
        - Updates retweets table with retweet details.
- Both functions interact with tweets, hashtag_mentions, and retweets tables for data insertion and validation.

5. List followers

There are 4 functions in the functionality 4: list_followers, show_follower_details, parse_datetime and follow_user.

- The function list_followers retrieves data from the tables to get the id, and the name of the followers who followed our user. Then, the function will print out the follower's information based on the data that was just retrieved. While printing the followers' information, only 5 followers are displayed at a time, and the user has a chance to choose to view more. After displaying all the followers, or if the user chooses not to view more, the function will ask if the user wants to view more details about one follower. Next, the function will search through all the followers in the list to seek a matching case. After finding a matching case, the function will call the next function show_follower_details.
- The function show_follower_details will first retrieve data about the number of tweets the selected follower has posted, the number of followers the user has, and the number of users that the follower followed. Then, the function will display that information. After displaying the above information, the function will select all the tweets, including retweets and convert the date and time in SQL to datetime in Python using the function 3. parse_datetime. After the date and time the follower posts the tweets are converted to datetime, the function sorts the list of tweets by descending order of time. Then, the function will display the follower's tweets. After displaying all the tweets or the user chooses not to view more, the program asks if the user wants to follow back. If the user wants to follow back, the program calls the function follow_user.
- The function follow_user updates the follows.

6. Logout

- Purpose: Allows the user to end the current session and return to the login page.
- How it Works:
    - When the user enters "logout", the system sets the isLogin flag to False and clears user_id, effectively logging out the user.
    - Displays a confirmation message: "You have been logged out."
    - Returns control to the main loop, allowing the user to log in again without exiting the program.
    - Ensures a secure exit from the current session without closing the database connection.
    - Provides the user with the option to re-login or exit the application entirely.
- NOTE: Initially, we planned to implement the logout as a separate function. However, we found it simpler for the programmer to handle the logout option directly within the main loop. This approach retained the original idea but made the loop in main.py more flexible.

7. Main function (main.py)

- Purpose: Acts as the entry point for the application, managing the overall workflow and interactions with the SQLite database.
- Structure:
  - Initializes the database connection and cursor.
  - Handles user login and logout functionality in a loop, allowing users to re-login after logging out.
  - Provides options for various actions: searching tweets, composing tweets, retweeting, replying, listing followers, and searching users.
  - Connects each action to its corresponding module for a modular and maintainable code structure.
- Testing: To test the application, simply run main.py and follow the prompts to interact with each feature. Ensure that the SQLite database (test.db) is properly set up with the required tables before running the tests.

(c) your testing strategy

- Sample testing: Sample Data-Generated by AI The design of the sample data takes some edge cases into consideration, such as duplicate user names, multiple hashtags under one post, and spam retweets.
- In order to test the functionality easier, we also created a function named `print_all_data()`,so that we can view all tables in our system.
- All members in our group test the program on their lab machines to make sure it works smoothly as expected.
- We create test cases by reading the mini project description, eclass discussion forum and TA's announcements about updating information.

(d) your group work break-down strategy

- We agreed with the work divided among everyone, and communicated effectively on Discord.
- When any member detected the bugs, that member would fix it on their own files first before pushing it into Github.
- We arranged 4 meetings in total (each last at least 4 hours):
  - Introduction and Dividing
  - Checking on the progress of each functionality
  - Combining and Test the main function
  - Revising the code to meet all the requirements of the mini project 1
- Each person spent at least 10 - 15 hours to complete their part.
- Communication was effective and straight-forward via Discord group message, and in-person meetings.
- We always notified each other whenever there are any changes in codes due to bugs, displaying, etc.
- Work division:
  - Darlene Nguyen: Compose Tweet, Login Page (i.e. structure, email checking), Displaying, testing, dividing work
  - Norman Wong: Search for Users, Login Page (i.e structure), Testing
  - Khanh Bui: Search for Tweets, Login Page (i.e structure), main.py structure
  - Lucy Jing: List Followers, Database Sample, Testing