**PROJECT**

**DSA-II**

**PROJECT STATEMENT**

A simple way to look at the Project would be looking it like a MOCK TWITTER Platform. It is just a model, basic framework of how Twitter works.

* Users can follow each other.

That is how users follow each other to get their tweets and updates. This requires use of Graph Data Structure in bulk.

* A timeline for each user with his own tweets.
* One can include #tags in a tweet, which can later be searched. Upon searching a #tag all the tweets with that #tag are displayed using Hashing.

It is just the casual #tags which make topics trends.

We have stored data for this using File Handling. Data structures have been used in all functionalities of the project. Login system is based on Hashing including Forgot Password feature.

**PROJECT DESIGN**

Project has been designed in a way that:

* Users can create new accounts. The details are stored in a text file.
* Existing users can sign in. Forgot password feature can be used in case the user forgets his password.

This login system is mainly based on Hashing, where user\_name will be the key for the Hash Function and for collision handling, Chaining has been used.

* Once a user is signed in he given option to

1. Tweet
2. Timeline
3. Follow
4. Trending Topics
5. Log Out

* **Tweet**: The user can express herself or himself within a word limit of 140 words (same as twitter). The tweets are stored in a text file.
* **Follow**: The user has an option of following all the users existing in the data base. Once a user starts following a user the second user also follows back, using Undirected Graph data structure.

It has search by Name feature in which user can search other users using their name and once the Name is found they have an option of following them or not.

* **Timeline**: All the tweets of the user and the users he/she follows are displayed in timeline using File Handling.
* **Logout**: The user is signed out of his account.
* While tweeting when a user creates a #tag they are saved in a Hash table and can be later searched using Hashing, for collision handling Chaining had been used.

Upon searching an existing #tag all the tweets having that #tag are displayed.

**CODE SKELETON**

The code has been divided into three modules

1. PART I
2. PART II
3. PART III

**PART I LOGIN SYSTEM**

Part 1 has mainly two functions based on Hashing.

Sign up–It takes user information and prints in a text file. Using Hashing we give an error if username already exists.

Login–It takes username and password, calls search\_hash\_tbl function to verify the credentials. If password is not correct the search\_hash\_tbl function is again called and answers for the asked question for Forgot password are verified.

**PART II NETWORK OF USERS**

This part has the main use of Graph Data structure with following functions:

create\_network – This is used to initialize Graph, which will be used to connect users and help the Follow function. Adjacency list are being used not the matrix implementation.

connect\_network: This plays the vital role of connect users together i.e. follow each other. Adjacency list is populated here after adding edges between the users who want to follow other user.

search\_#tags: Hashing is used to read the tweets from text file and #tags are inserted into them. After searching that #tag all the tweets having that #tag are displayed.

**PART III**

Finally, where does the data come to the various functions after first execution of code is over?

File Handling is the answer.

All the data is being written in text files and during the second time of execution we read text files and input the data into different Hash Tables and Graph Adjacency List is updated. All these functions are called before the user interacts.

Some functions like insert\_linklist, vertexReturn etc. are not specified to a part and needed everywhere in the program, so they are not listed under any part.