# A Graph-based Social Media Networking

Maitrey Katkar
*Information Technology Department*
*Vishwakarma Institute of*
*Technology,Pune*
Pune, India
maitrey.katkar22@vit.edu

Omkar Khanvilkar
*Information Technology Department*
*Vishwakarma Institute of*
*Technology,Pune*
Pune, India
omkar.khanvilkar22@vit.edu

Janvi Kharat
*Information Technology Department*
*Vishwakarma Institute of*
*Technology,Pune*
Pune, India
janvi.kharat22@vit.edu

Gita Kolate
*Information Technology Department*
*Vishwakarma Institute of Technology,Pune*
Pune, India
gita.kolate22@vit.edu

Babusha Kolhe
*Information Technology Department*
*Vishwakarma Institute of Technology,Pune*
Pune, India
babusha.kolhe22@vit.edu

*Abstract—* **The widespread use of social media platforms has recently changed how people connect, share knowledge, and collaborate on a worldwide scale. This study presents a novel method for integrating graph theory ideas with social media networking to increase its efficacy and efficiency. We suggest the use of a framework called Graph-based Social Media Networking (GSMN) to represent the complex network of social interactions and information sharing in online communities. In this article, we give a thorough examination of several graph data structures and how they might be used to represent social relationships. We investigate the development and application of a cutting-edge social media platform that makes use of these data structures to improve user interaction, information sharing, and content suggestion. In addition to strengthening the theoretical underpinnings of graph-based social networks, this research offers useful advice for the development of cutting-edge social media platforms.**

**Keywords—Graph data structures, Graph algorithms, content recommendation**

## I. INTRODUCTION

The explosion of social media platforms in the modern digital landscape has completely changed how individuals interact with one another, share experiences, and create communities on a global level. The complexity of social connections has increased enormously as a result of these platforms' continued encroachment into every part of our life. Users now establish complex webs of relationships that span a variety of connections, including friendships, professional partnerships, and common interests. Users are no longer simply connected. This complexity necessitates a fundamental redesign of the underpinning systems supporting social media networks. This research presents a ground-breaking method for social media networking that takes advantage of the inherent potential of graph data structures in answer to this requirement.

Traditional social platforms, while pervasive, often struggle to capture the nuanced nature of modern social relationships. The multidimensional relationships that determine user interactions are not well represented by conventional designs. A game-changing idea is the idea of using graph data structures, where nodes stand in for users and edges for connections. Social media platforms can overcome their present constraints by using this strategy. A greater knowledge of user interactions is made possible by the intuitive framework that graphs offer to model complex relationships. In order to reimagine the online social experience, this article explores the creative combination of social networking and graph theory. This project aims to usher in a new era of intelligent, adaptive, and socially cohesive online ecosystems, profoundly transforming the social media networking environment through the integration of graph data structures.

## II. METHODOLOGY

The system's core components were meticulously crafted to ensure a seamless user experience. Initially, the system initializes by creating an empty social network graph, setting the stage for subsequent user interactions. To maintain network continuity, existing graph data is seamlessly loaded from "graph_data.txt."

User registration, a fundamental aspect, involves capturing essential user details like name, email, gender, age, userID, and password. A critical step in this process involves verifying the uniqueness of the chosen userID through a dedicated validation mechanism. Upon successful registration, user credentials are securely stored in "user.txt," and individualized user profiles are established within "userID.txt."

User authentication is seamlessly integrated, requiring users to input their userID and password. Authentication is accomplished using our validation function, ensuring secure access to the system. Successful authentication results in the assignment of a unique loggedInUserIndex for the authenticated user.

The "Add Friend" feature allows users to expand their social circles. Users input their friend's username, which undergoes validation against existing records in "user.txt." Upon validation, a connection is established between the user and their friend within the graph, enhancing their social network.

Exploring user profiles is made intuitive. Users enter the target friend's userID, and the system retrieves and displays the relevant profile information from "userID.txt."

Our system also empowers users to create and share posts. Post content is seamlessly appended to the user's data file ("userID.txt"), ensuring data integrity. This functionality is carefully guarded by a verification mechanism that ensures the user is logged in before accepting posts.

The "Display Friend List" feature enhances social connectivity. By checking the user's login status, the system efficiently traverses the adjacency list of the user's index within the graph. Subsequently, it displays the usernames of their friends, fostering a sense of community within the platform.

To ensure persistent data and facilitate future system reinitialization, our methodology includes a robust data saving mechanism. The system writes the graph's current state into "graph_data.txt." This file includes the number of vertices and the graph's adjacency list data, enabling seamless system restoration.

For fundamental operations, the algorithms used have linear time complexity (O(N)), ensuring efficiency even in large networks. Our method ensures scalability and efficient memory utilisation with a space complexity of O(N) for user data and O(N2) for the adjacency matrix-like representation of the graph.
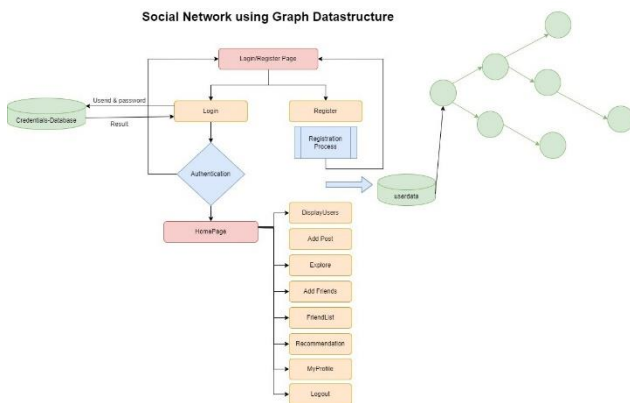


*Fig1: Algorithm for the project*

## III. RESULT AND DISCUSSIONS

The code implements a user-friendly social media networking system, allowing users to register, create unique UserIDs, and log in securely. The system allows users to explore friends' profiles, view posts, and add new ones. Friend connections are established, allowing users to

view detailed profiles. The system operates seamlessly, allowing users to navigate through options like registration, login, adding friends, exploring profiles, adding posts, viewing friend lists, and exiting. This successful implementation demonstrates user-friendly interaction and content sharing. The screenshots of the output of executed code are mentioned below in the figure 2.
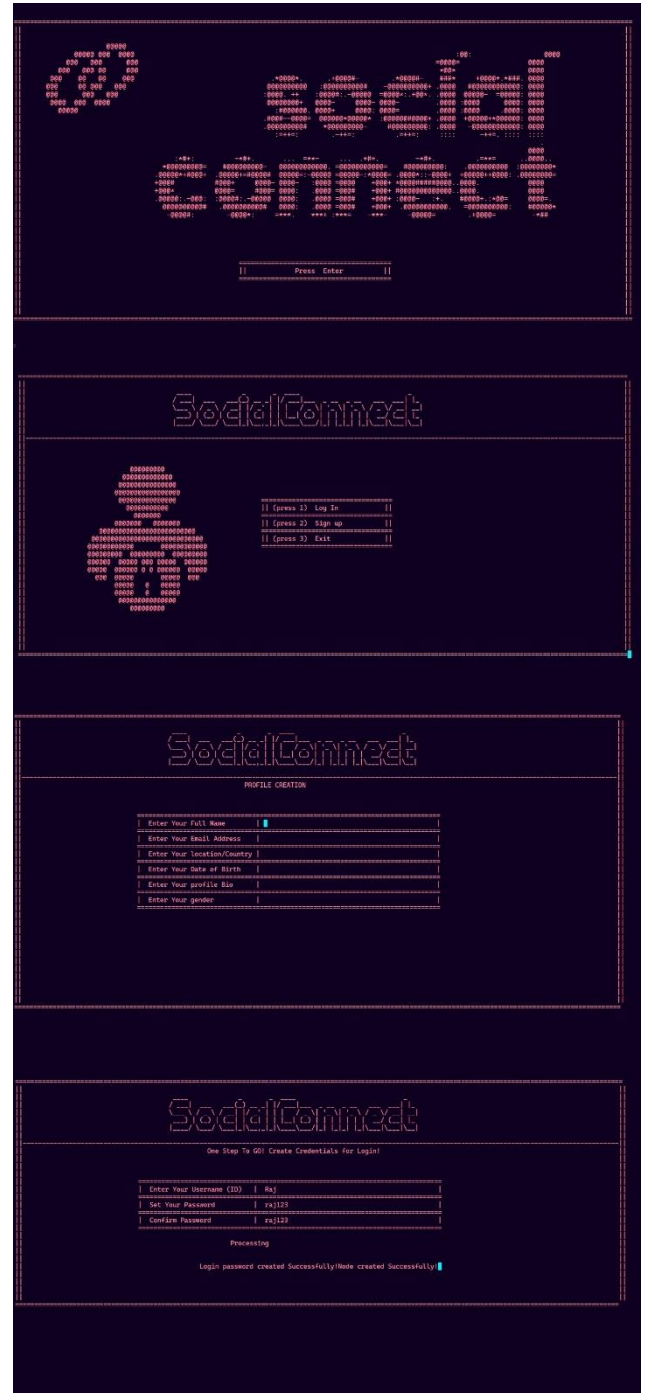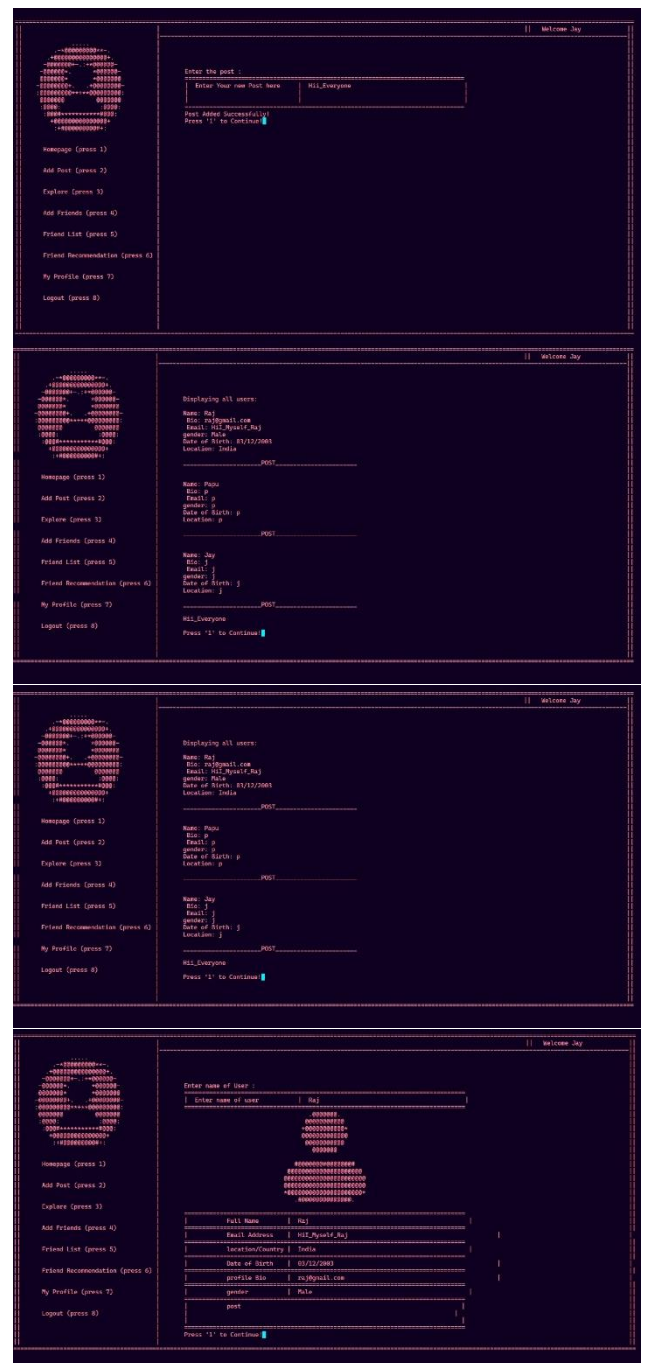
*Fig2: Output of the executed code*

## IV. CONCLUSION

Our initiative revolutionizes social media networking by integrating advanced graph data structures and user-centric design concepts. It ensures accurate user interactions, including registration, authentication, friend connections, profile exploration, and content sharing. Strong validation methods and secure storage protocols preserve user information. The system's scalability ensures peak performance, even as user bases and networks grow. This cutting-edge networking solution promises improved online interactions and sets a new benchmark for technological innovation, user experience, and data management.

## REFERENCES

[1]  https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/

[2]  https://stackoverflow.com/questions/7864194/graph-data-structures-with-millions-of-nodes-social-network

[3]  https://towardsdatascience.com/social-network-analysis-from-theory-to-applications-with-python-d12e9a34c2c7

[4]  https://youtu.be/59fUtYYz7ZU?si=6bt673K26Off1Djn

[5]  https://youtu.be/N2P7w22tN9c?si=jAhRujAT8Mopv3q8

**GitHub Link:**
**https://github.com/omithecoder/ADS_CP.git**