Application Requirement: Social Network Analysis Tool

Objective:

Develop a Social Network Analysis Tool that allows users to manage a social network and perform various analyses on it. The system should support operations such as adding users, creating friendships, finding the shortest path between users, and identifying communities.

Requirements:

User Management:

- Add a new user with details: user ID, name, and email.
- Remove a user-by-user ID.
- List all users.

Friendship Management:

- Create a friendship between two users by their user IDs.
- Remove a friendship between two users by their user IDs.
- List all friends of a user-by-user ID.

Network Analysis:

- Find the shortest path between two users using any suitable algorithm of your choice. (*Refers to finding the minimum number of connections (or "hops") required to travel from one user to another within the social network*)
- Identify communities using any suitable algorithm. (*Refers to detecting groups of users within the social network who are more densely connected to each other than to the rest of the network. These groups are often referred to as "communities" or "clusters."*)
- Calculate the degree centrality of each user. (*Degree centrality is a measure of the number of direct connections a user (node) has in a social network. In other words, it counts the number of friends (edges) each user has. This metric helps identify users who are highly connected within the net*)

Constraints:

- The system should handle many users and friendships efficiently.
- The system should provide clear and concise error messages for invalid operations (e.g., user not found, friendship already exists).

Implementation Details:

- Use appropriate data structures to store users and friendships.
- Implement efficient algorithms with clear documentation on their time and space complexity.
- Write unit tests for the main functionalities.

Example Classes:

- User
- SocialNetwork
- Example Methods:
- addUser(User user)
- removeUser(String userId)
- listUsers()
- createFriendship(String userId1, String userId2)
- removeFriendship(String userId1, String userId2)
- listFriends(String userId)
- findShortestPath(String userId1, String userId2)
- identifyCommunities()
- calculateDegreeCentrality()