Assignment 2: conf-chat, a simple message-sharing system using a P2P architecture.

Task: Build a simple message-sharing/chat system using a P2P architecture.
This project aims to develop and evaluate a Peer-to-Peer (P2P) solution for a two-way and many-to-many chat system.

The work consists of designing and implementing a P2P protocol for a chat system.

A potential/suggested use case: (This is provided as a sample. You are free to simplify or propose your own system. But you must explain your system in README.md, if you change this use case).

A user joins Conf-Chat by providing their username and password. These are defined when the user registers with the system, at which time they must provide his full name. The application allows users to change their passwords.

Once a user logins, they will be able to see which of their friends are logged in. They can also choose to ask for new friends. For this, the user searches for friends by providing names, which are matched (word match, words provided exist in the full name) against all the users of the system. Before a user can be a friend of another user, the other user has to authorize them, at which time they both become friends.

A user can send messages to any of their friends, either they are online or not. If the friend is online, they will receive the message at once, otherwise they will receive the message when they logins again, even if the sender is no longer online.

A user can also create a conference chat, here messages are sent to all the participants. Any of the conference participants can add other participants to the chat, at any time. Users can not join an existing chat unless they are added by participating users. If a user leaves the system, they automatically leaves the conference and will receive no further messages from that chat should they login again, unless they are invited back.

Conf-Chat is a pure P2P application, which does not use any centralized server. Conf-Chat nodes may be added or removed as needed without the need for system reconfiguration.

Submission Details

Step by step process:

1) Fork https://github.com/KathiraveluLab/conf-chat to your own GitHub repository and implement your Conf-Chat there.

2) Install P2P architecture such as OpenChord http://open-chord.sourceforge.net/.

In your README.md, specify the installation steps you followed (especially if it differs from the official guideline). If the installation steps followed the official guidelines, it is sufficient to just add the names of the frameworks you used along with a link to the official installation guide page.

3) Implement the simple message-sharing/chat system using OpenChord or another P2P architecture.

4) The use case is provided as a sample. You can have your own implementation scenario. In that case, clearly describe your scenario/use-case/functionality in the README.md.

5) In class, do a quick demo to show your project at work.

The total points possible is 10:

The rubrics is as below:

1. Functionality of the product, as observed from Demo: Good (3), Fair (2), Poor (1), None (0).
2. Code completeness in the GitHub repository: Good (3), Fair (2), Poor (1), None (0).
3. Documentation (README.md, ...) completeness in the GitHub repository: Good (3), Fair (2), Poor (1), None (0).
4. Clarity of the demo: Excellent (1) or None (0).

Feel free to reach out to the instructor for questions and clarifications at any time.
Submission is, just the GitHub link of your fork.