CS 408 - Computer Networks - Spring 2022

Term Project – Step (Phase) I

SUBook – a Facebook like Social Networking Application

This project is made of two steps (phases).
Project Step (Phase) 1 Deadline: 16.05.2022, 16:30
Project Step (Phase) 1 Demo: to be announced

Introduction

In this project, you are going to develop a social networking application called *SUBook* (an oversimplified version of Facebook) by implementing client and server modules. (i) The ***Server*** module manages the storage of posts, posts feed, and friendships between the users, and (ii) the ***Client*** module acts as a user who shares posts, adds and removes other users from his/her friendship, and views posts of other users.

The server listens on a predefined port and accepts incoming client connections. There might be one or more clients connected to the server at the same time. Each client knows the IP address and the listening port of the server (to be entered through the Graphical User Interface (GUI)). Clients connect to the server on a corresponding port and identify themselves with their usernames. Server needs to keep the usernames of currently connected clients in order to avoid the same name to be connected more than once at a given time to the server.

On the **server side**, there is a predefined database of users which are presumed to be registered to the social network so that you do not need to implement any registration process between a client and the server (actually it was done in HW1). That user database has been provided to you in package with the term project as a textual file named user-db.txt together with this guide document. You should **parse this file** in order to load the usernames in your server program; we can use a different file in the demos, so please **do not hardcode the usernames in your code**. For simplicity, a client, whose username is in the user database, will be able to connect by providing his/her username only (i.e. no password or other type of security). Once connected, he/she will act as mentioned in the rest of this document.

Each step is built on the previous step, and each has specific deadlines and demos, so it is crucial to complete to first step in order to being able to do the second step of this project.

Project Step (Phase) 1 (Deadline: 16.05.2022, 11:30):

After the server starts listening, clients start to connect to the server. Only users from the database text file that we will provide can connect to the server. A connected client can share posts and the server should have the ability to store them and make them ready to show in other users' feeds when requested.

Each user can get all posts of all other users from the server upon request by pressing on a button on client GUI. When such request is issued, the server will send all posts belonging to all users in the platform except those of the requesting user. These posts must be shown in the

requesting client's GUI as they are received with all its attributes which is explained below. The feed request can be issued repeatedly.

Each post has three attributes: **(i)** the username of the post owner, **(ii)** a unique ID assigned by the server to posts, and **(iii)** the timestamp (date and time) that the post is shared. These attributes should also be shown, in addition to the post itself, at the client side.

Note that a post owner can be offline when a particular user requests all of the posts. Still, the requesting client must be able to see all posts even if one or some of the post owners are offline. Therefore, server needs to store the posts with all its attributes in a text file. The format of this text file and the details of parsing are up to you. Users perform all of the operations through a GUI; such as connecting to the server, disconnecting from the server, entering their username, sharing posts, requesting posts feed, etc.

This step is the basis of the project. Although the project has some additional operations (such as add/remove friends, delete posts (bonus), request friend lists, etc.), you do not have to implement those in this step of the project. You will implement them in the second step.

For programming rules and submission specifications, please read the corresponding sections at the end of this document.

Furthermore, in package with the Term Project – Step (Phase) I, you will get the grading criteria upon which your Phase I will be graded. We will determine the points for each criterion after the deadline of Phase I. Also, on SUCourse you will have a demo guide video which briefly explains everything you need to know and do for Phase I.

Term Project – Phase I - Server Specifications:

- There is only one server running on this system.
- The port number on which the server listens is **not to be hardcoded**; it should be taken from the Server GUI.
- The server will start listening on the specified port. It must handle multiple clients simultaneously. To do so, whenever a client is connected to the listening port, the corresponding socket should be added to a list and the server should continuously accept other client sockets while listening.
- Only users who reside in the user database can connect to the server as a client. In other words, no user with a username that does not exist in the user database can connect, and its error message should be given in both GUI.
- All activities of the server should be reported using a rich text box on the Server GUI including the usernames of the connected clients, the usernames of the disconnected clients, error messages (Problem with port number, not a database user etc.) as well as all the post sharing and post feed request details**. We cannot grade your project if we cannot follow what is going on; so the details contained in this text box is very important.**
- Server must handle multiple connections. At the same time, one or more clients can connect, share posts and request their feeds to/from the server.
- Connected clients' usernames must be unique; therefore, the server must keep track of connected clients' names. If a new client comes with an existing username, server must not accept this new client and report an error message on the GUI.
- When the server application is closed (even abruptly), nothing should crash! Also, the process in the operating system regarding the server should be terminated properly.

Term Project – Phase I - Client specifications:

- The server's IP address and the port number **must not be hardcoded** and must be entered via the client GUI.
- There could be any number of clients in the system.
- If the server or the client application closes, the other party should understand disconnection and act accordingly. Your program must not crash!
- All activities of the client should be reported using a rich text box on the client GUI including sharing posts, error messages, and posts feed request details. **We cannot grade your project if we cannot follow what is going on, so the details contained in this text box is very important.**
- Each client must have a unique username. This username must be entered using the client GUI. This username is also sent to the server. The server identifies the clients using their usernames.
- Each client can share posts and request post feeds at any time. If a client shares a post, this post is stored on the server side.
- Each client can disconnect from the system at any time. Disconnection can be done by pressing a disconnect button on client GUI or by just closing the client window.
- If the client application is closed (even abruptly), nothing should crash! Also, the process in the operating system regarding the client should be terminated properly.
- Both connection and post transfer operations will be performed using TCP socket.

----------END OF PHASE I----------

Group Work

- You can work in groups of **one, two, three and maximum four** people for both steps (phases). There is a shared Excel file with the formed groups, so you can decide to join one. No group changes are allowed after the first step (phase). Any group changes must be performed before the submission of the first step. However, we do not recommend changing groups once you start the project since moving codes might lead to plagiarism.
- All group members submit the same solution project for their term project, for each phase.
- Equal distribution of the work among the group members is essential. All members of the group should submit all the codes for both client and server. All members should be present during the demos. In case of any dispute within the group, please do not allow the problematic group members to submit the code, so that he/she will not be graded. However, if a group member submits the same code, then the other members automatically accepts his/her contribution. In such a case, the same group continues with the second step. In other words, submitting step (phase) 1 together and separating for the other steps is **not** possible.
- If a particular student does not submit the first step of the project (due to being without a group or being excluded from a group due to low contribution/effort), he/she can join another group for the second step.
- Similarly, if a particular group member does not work enough in the second step, please do not let him/her submit. You can also inform us about this.
- Your TA Müge Kuşkon (mugekuskon@sabanciuniv.edu) is responsible for keeping track of the groupings. You can contact her if you have any questions related to the term project.
- You have a chance to form your own groups. However, if you cannot find enough people to form a group, then you have to accept to work with people that we assign. If you do not like to work people that you do not know, then form your own groups or work yourself!
- **If someone does all of the phases alone, he/she will get an extra 3% towards the final grade**

Programming Rules

- Preferred languages are C#, Java and Python, but C# is strongly recommended
- Your application should have a graphical user interface (GUI). **It is not a console application!**
- You must use pure TCP sockets as mentioned in the socket lab sessions. No other socket classes is allowed.
- Your program should be portable. It should not require any dependencies specific to your computer. We will download, compile and run it. If it does not run, it means that your program is not running. So do test your program before submission.

Submission

- Submit your work to SUCourse+. Each step will be submitted and graded separately.
- Delete the content of debug folders in your project directory before submission.
- Create a folder named **Server** and put your server related codes here.
- Create a folder named **Client** and put your client related codes here.
- Create a folder named **XXXX-Lastname-OtherNames-StepY**, where XXXX is your SUNet ID and Y is the project step (1 or 2) (e.g. mugekuskon-Kuskon-Muge-Step1). Put your Server and Client folders into this folder.
    - Compress your XXXX_Lastname_OtherNames_StepY folder **using ZIP or RAR**.
- You will be invited for a demonstration of your work. Date and other details about the demo will be announced later.
- **24 hours late submission is possible with 10 points penalty (out of 100).**

For personal questions and support, you should exclusvely send an email to your TA Müge Kuşkon (mugekuskon@sabanciuniv.edu) or attend her online office hours given below (and also updated in the corresponding OH document on SUCourse+):

Through this Zoom link:
https://sabanciuniv.zoom.us/j/2457215757?pwd=ZEl1TjdUd1JUYTJRQmx1KzNYVnIvZz09

CS204 Team (Müge Kuşkon, Artrim Kjamilji)
Good luck!