2019 Computer Network Final Project cnMessage Report

group 23 - 現在放棄寒假就開始囉

Members

- B06902093 王彥仁
- B06902026 吳秉柔
- B06901087 翁瑋襄

Our package (GitHub Repo)

https://github.com/wangyenjen/CN-Project2_ChatRoom/

User and Operator Guide

Our website contains:

- / (index): It contains welcome messages and asks the user to enter the username to chat with. Users can enter a multiperson chatroom by keying in usernames separating by `,`. For example, if one wants to chat with "john" and "mary", he should enter "john, mary".
- `login/`: If users haven't logged in, they will be redirected to this page. It requires the user to enter his username and password. The page also contains a link to signup for those aren't signuped yet.
- `signup/`: Users are required to enter a unique username and password to signup.
- `room/`: The chat room. The chat log shows the date, the time, the username, and the text of past messages. Users can also choose to send files.

Instructions on how to run server and clients

For server

First of all, you need to install `docker` and `docker-compose`. If your environment is Ubuntu, you can execute the `install_docker.sh` to install them, otherwise, you have to install them yourself.
When you have `docker`, you can execute the `run.sh` directly.
The `run.sh` script will help you to start the server.

For client

You can type the **IP address of the server with 8000 port** in any browser, and then you will see the beautiful user interface.

System and Program Design

We use django as our web framework and SQLite as our database.

Other backend operates as following:

- `login/`: Authenticate the user (check the username and the password) and update the current user.
- `signup/`: Enter the username-password pair into the database and login the user. Redirect the user to the index page.
- `logout/`: Set the current user back to Anonymous user.
- `find_room/`: For a pair (or a group if multiperson chatting) of users, we sort the usernames by lexicographical order and concatenate the usernames with `,` . Then, we use sha256 to obtain a hash value, which is the room id for this pair (or group) of users.
- `room/`: We print all historical messages from the database and then sync all messages from each users instantly by using the WebSocket package.

- `messages/`: We save each message into the database, including `room_id`, `sender`, `time`, `text`, and whether it is a file or a text message (`is_file`)
- `upload/`: Save each file into the `SAVED_FILES_DIR` directory.
- `download/<filename>`: Obtain historical files by its filename from `SAVED_FILES_DIR`.

Other things you want to say, if any

Bonus

- multiperson chatting
- storing transferred files as users' historical data.
- beautiful UI
- use database to store all the messages