

Assignment 3: Socket Programming Group Chat

20235140 Boseong Lee

November 2024

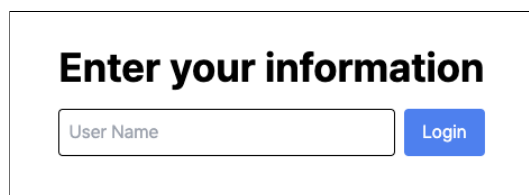
1 Node.js and library environment information

- Node.js version: 20.12.1
- React.js version: 18.3.1
- Nest.js version: 10.4.6
- Socket.io version: 4.8.0

This project uses mono-repo structure.

`packages/client` is the React.js client. `packages/server` is the Nest.js server. `packages/shared` is the shared library between client and server.

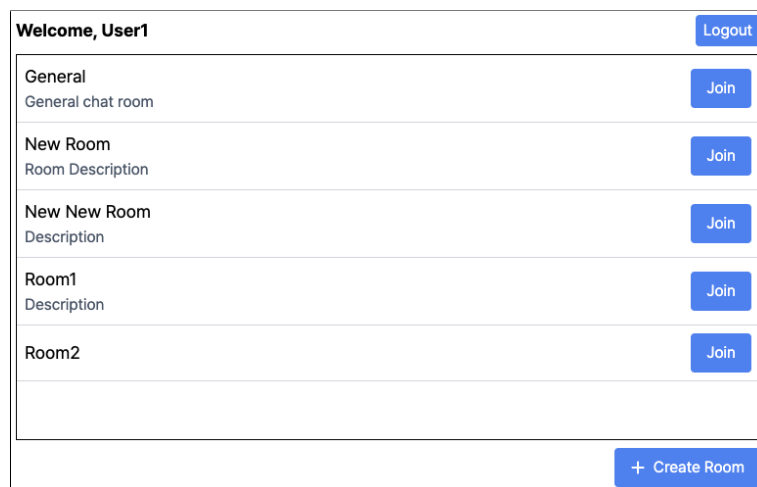
2 Key Features



The login screen features a central white box with a black border. Inside, the text "Enter your information" is displayed in a bold, black font. Below this text is a text input field with the placeholder "User Name" and a blue "Login" button to its right.

Figure 1: Login Screen

In login screen 1 User can set their username appearing in the chat room.



The chat room list page is a rectangular interface. At the top left, it says "Welcome, User1" and at the top right is a blue "Logout" button. Below this is a list of chat rooms, each with a title, a description, and a blue "Join" button. The rooms are: "General" (General chat room), "New Room" (Room Description), "New New Room" (Description), "Room1" (Description), and "Room2". At the bottom right of the page is a blue button with a plus sign and the text "Create Room".

Figure 2: Chat Room List Page

In chat room list page 2 User can see the list of chat rooms. In header of the page, user can see the username set in the login screen and the Logout button to use another username. Each chat room has the title, and description. Pressing the Join button will move to the chat room page. User also can create a new chat room by pressing the Create Room button.

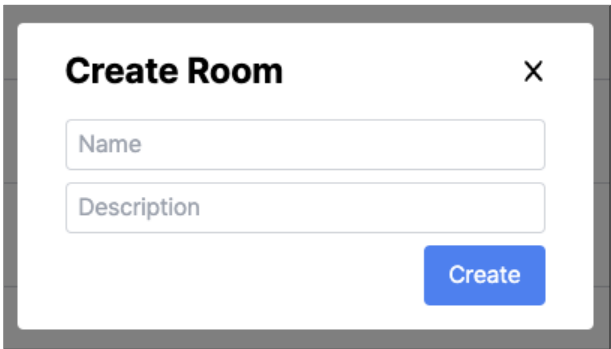
A modal window titled "Create Room" with a close button (X) in the top right corner. It contains two input fields: "Name" and "Description". Below the input fields is a blue button labeled "Create".

Figure 3: Creating Room

Create Room button will show the modal 3 to create a new chat room. User can set the title and description of the chat room. After, pressing the Create button will create a new chat room.

This project uses `socket.io` to implement real-time chat feature. When user create a new chat room, client sends `socket.io` event to the server to create a new chat room. Then, server broadcasts the new chat room to all the clients except the user who created the chat room. It makes all the clients to see the new chat room in real-time.

A chat room interface. At the top, it says "Welcome, User1" with a "Logout" button. Below that, it says "Room: Room3" with a "Leave" button. The main area is a large white box containing the text "- enter message below to send -", "- 'User2' joined -", and "- 'User3' joined -". There are two message bubbles: "User2: hello~" and "User3: hihi~~~". At the bottom, there is a text input field and a blue "Send" button.

Figure 4: Chat Room

When user joins the chat room, the user can see the chat messages 4 in real-time. User can send the message by typing in the input box and pressing the Send button. The chat messages are shown in the chat room in real-time using `socket.io`. Once user leaves the chat room, the user will not receive the chat messages from the chat room. Also, there are system messages to notify the user joining and leaving the chat room.

3 How to run the project

This project uses `yarn berry` as a package manager. Especially, this project uses mono-repo structure, so you need to run `yarn` in the root directory. `yarn berry` can be installed by using `corepack` which is installed by `node.js` (<https://nodejs.org>). After, you can run the project by following the steps below.

```
$ npm install -g corepack # if you don't have corepack installed
$ corepack enable

# Install dependencies
$ yarn install

# Run the server and client (run in different terminal)
$ yarn workspace server start
$ yarn workspace client dev
```

You can access the client at `http://localhost:5173`. The port number can be different.