#Chat.io (Simnet, Gnome**) [79] [+ntl 85]

@Gnome**
@bjoggib
@mr.miyaki
@chatiob0t
+larry
+karatekid
lion-o
LeBrona

* Now talking in Chat.io
* Topic is 'Everyone is welcome!'
| Fubar for el presidente!
* Set by Gnome**gnome@chatio.simnet.is
on Wed March 6 13:37:00

Hello, everybody! Welcome to Chat.io!

Chat.io

A terrible thing just happened! The mIRC has been wiped from the planet and nobody has a copy of its installation file! The **CWS** (*Chat World Service*) wants you to create a new mIRC so the world can chat once again! Will you lend us your helping hand?

Assignment description

In this assignment, you should write a chat program using **React** and socket.io. Below is an enlisting of all the functionality for this assignment:

- **(10%)** The user should upon arrival specify his/her nickname. If the nickname is free, i.e. no other user is currently active with the same nickname, he/she can proceed, otherwise a new nickname must be provided. Note: *no password is required, only nickname*
- **(10%)** After the user has been identified, he/she should see a list of chat rooms already active
- **(10%)** The user should be able to join a chat room, and leave a room as well. It should of course also be possible to create a new room
- **(10%)** Inside a given room, the user should be able to send messages to the room, see previous messages, and see new messages appear in real time (*without having to refresh the page manually*)
- **(10%)** It should be possible to send a private message to another user
- **(10%)** The creator/ops of a room should be able to kick a user from a room. A user which has been kicked from a room can re-enter. The creator/ops can also ban a user, which means he/she won't be able to join the room again
- **(10%)** The creator/ops can give other group members an "op". An op is similar to granting someone admin privilege

In addition, the following technical requirements are given:

- **(10%)** All state which is truly global should reside in the **Redux** store state - therefore **Redux** must be setup in this project and used accordingly
- **(10%)** Each component should reside in a single folder, where the implementation of the component is and a test for each component
  - Each component should be tested using Jest
  - Each component should make use of **PropTypes** (*if it accepts props*)
- **(5%)** All external dependencies should be installed with npm/yarn
- **(5%)** Webpack should be setup and do the following:
  - Bundle all components in a single JS file
  - Run tests
  - Minify the code
  - Offer an option to enable watch mode
  - Integrate Babel in order to be able to write ES6 code

Server
The server can be downloaded in the assignment section in **Canvas**. It is a **NodeJS** server and information on how to use the server is in the **README.md** file in the .zip file for the server.

Dependencies
All dependencies are allowed in this project

Penalties
You must remember to exclude **node_modules/** from your submission. If you fail to do this you will get a penalty of -1. Please don't forget this!

Submission
A single compressed file (***.zip, *.rar**) should be submitted to **Canvas**.
Note: unless you make changes to the supplied server, then DON'T hand in that code! The server code should not be a part of your application folder.