

Full stack programming javascript Task v1

Implement chat application using NodeJS webserver with websockets. The application work-flow should be the following:

- introduction page with user\_name textbox and login button (no password is required). Login is not browser and session based but socket based – meaning every browser tab should be treated as different socket and user.

- after login a small chat window with textbox and “Send” button is provided.

- send button will read the text from textbox and send it to the server (this should be done using websockets).

- messages and user data should be kept in the server variables – no external data storage is needed (once the server is shut down all data will be erased)

- upon logging in all messages from the beginning of the server should be displayed

- Message should be in following format: {{user\_name}} on {{datetime}} wrote: {{message}}

- certain backend validation should be considered:

1. if user\_name is already logged in write message to choose different user\_name
2. messages should only contain alphanumeric values along with following characters . , ? ! : “ ' Also message must not be empty string

- Logout button should be present also which will return user to initial page.

- If user is idle more than 5 minutes he/she should be automatically logged out

- There should be option for user delete message only if it is not older than 15 minutes.

- Chat should be global meaning everybody should be able to see everyones messages

- There should be jQuery plugin for choosing colors. When users selects color only his/her messages should be colored with that color. Color is black on first login (but color should be saved for next login).

Please make usage of promises instead of making “callback hell”. Also please use as much of ES6 features as possible (string literals, arrow functions, destructed objects, const and let instead of var).

On front-end only jQuery framework, its plugins, socket.io and promise libraries are allowed. Please add libraries through package managers (npm or bower).

Also please commit your task to any git cloud hosting daily so we can monitor your progress.

This task should take no more than 2-3 days.