W1 - Stack JavaScript

W-JSC-502

My_IRC

Développer votre Internet Relay Chat

2.0

My_IRC

repository name: my_irc

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

Sommaire

Introduction

Il s'agit dans ce projet de réaliser un serveur IRC grâce à NodeJS.

Votre serveur devra accepter plusieurs connexions simultanées.

Votre serveur devra implémenter la notion de "channels".

Il doit être possible de rejoindre plusieurs "channels" simultanément (Par exemple via un système d'onglet).

Restrictions

Pour ce projet, vous devez utiliser, et n'utiliser que :

- node.js (node)
- socket.io (web sockets et rooms)
- express.js (node)
- react.js (moteur de template js)

Cahier des charges

Gestion des utilisateurs

- Un système de connexion : l'utilisateur de votre site doit pouvoir se connecter en fournissant un nom d'utilisateur.
- Tous les membres peuvent modifier leurs informations et ajouter des channels.
- Le membre qui aura créé son channel pourra le supprimer et le modifier.

Gestion des channels

- Chaque action (création et suppression) sur les channels et changement de pseudo enverra un message global visible sur tous les channels.
- Un nouvel utilisateur se connectant à un channel devra envoyer un message visible sur ce channel.
- Un channel devra s'auto supprimer si personne n'y a écrit depuis 2 jours (pour la soutenance mettre 5 minutes).
- Les membres connectés à un channel devront pouvoir envoyer un message à tous les utilisateurs de ce channel, et seulement à celui-ci.
- Le serveur devra maintenir à jour la liste des utilisateurs connectés (les sockets) ainsi que des channels (avec la listes des personnes connectées).

Commandes

Il doit être possible d'entrer des commandes dans le chat afin de réaliser différentes actions:

- **/nick** `nickname` : définit le surnom de l'utilisateur au sein du serveur.
- **/list** `[string]` : liste les channels disponibles sur le serveur. N'affiche que les channels contenant la chaîne "string" si celle-ci est spécifiée.
- **/create** `channel` : créer un channel sur le serveur.
- **/delete** `channel` : suppression du channel sur le serveur.
- **/join** `channel` : rejoint un channel sur le serveur.
- **/leave** `channel` : quitte le channel.
- **/users** : liste les utilisateurs connectés au channel.
- **/msg** `nickname message` : envoie un message à un utilisateur spécifique.
- `message` : envoie un message à tous les utilisateurs connectés au channel.

  L'envoi des messages se fait obligatoirement en tapant sur entrée, donc pas de saut à la ligne.

Front

Ajouter une interface intuitive et responsive en REACT afin que l'utilisateur puisse utiliser le serveur IRC sans avoir à connaître les commandes du mode terminal.

Bonus

Vous êtes libre de faire les bonus que vous souhaitez.

Par Exemple :

- BBcode pour les messages
- Emojis dans les messages
- Autocompletion des commandes, channels, users
- Autolink des #channels et des @usernames
- Ctrl + entrée pour retour à la ligne dans un message
- Respecter la RFC1459
- Créer une BDD (avec Mongo par exemple, mais pas obligatoire) afin de sauvegarder toutes les informations nécessaires pour conserver les channels existants, les informations sur les utilisateurs qui utilisent le service et toutes autres informations qui vous sembleraient nécessaires.