AJAX

**DEADLINE:**

## FOLDER STRUCTURE

|  |  |
| --- | --- |
| FL16\_HW5/ \*  ├─ task  ├─ homework/ \*  ├─ task1/ \*  ├─ index.html \*  ├─ .eslintrc.js \*  ├─ js/ \*  ├─ index.js \*  ├─ task2/ \*  ├─ client/ \*  ├─ index.html \*  ├─ style.css \*  ├─ chat.js \*  ├─ server/ \*  ├─ index.js \*  ├─ .eslintrc.js \*  ├─ package.json \*  ├─ package-lock.json \* | \*  - required |

## TASK - 1

Use [this API](https://jsonplaceholder.typicode.com/) (<https://jsonplaceholder.typicode.com/>) for your calls.

1) get userlist GET /users

2) Display them as list with possibility to edit.

3) When editining is finished update user on the server(call PUT method)

PUT/user/${id}

4) Add possibility to delete user DELETE /user/${id}

5) Show spinner on every request call

You can modify index.html if you need it.

## TASK – 2

1. Run the WebSocket server from **task2** folder via command: **npm run start**

After that server will be available on *ws://localhost:8080*

1. The task is to implement the chat that can broadcast messages between 2 or more different windows as it is shown on the screenshot bellow.
2. Message should contain username, time and text content.

Graphical user interface, text, application

Description automatically generated

* Pay attention that you will not receive your own messages from server, you should connect at least 2 different browser windows to it, so that way one will send message and the other will receive it (and vice versa).
* You can use prompt() to ask user for his name when opening the chat.
* It is not required to strictly follow the layout and styles shown on the screenshot.
* You can edit any files in client folder.

**RESTRICTIONS**

* Using any external libraries are forbidden.
* Adding **task/** folder is forbidden. Do not push it to repository. (Only **homework/** folder should be pushed)
* Editing server folder is forbidden.

## BEFORE SUBMIT

* Code should be clean, without comments, readable, and tested
* Make sure your GitLab folder structure meets folder structure from this document (without **task** folder)
* Fix all issues which was found by w3org Markup Validation Service (<https://validator.w3.org/>)
* Use linter :

- In order to use npm package manager you should install nodejs (https://nodejs.org/ )

- Install eslint to check your code (npm install -g eslint)

- open a terminal (or cmd)

- run eslint (i.e. eslint ./js/task1.js)

Code should be without ‘errors’

## SUBMIT

* The **FL16\_HW5** folder without **task** folder should be uploaded to GitLab repository “**FL-16**” into **master** branch.