

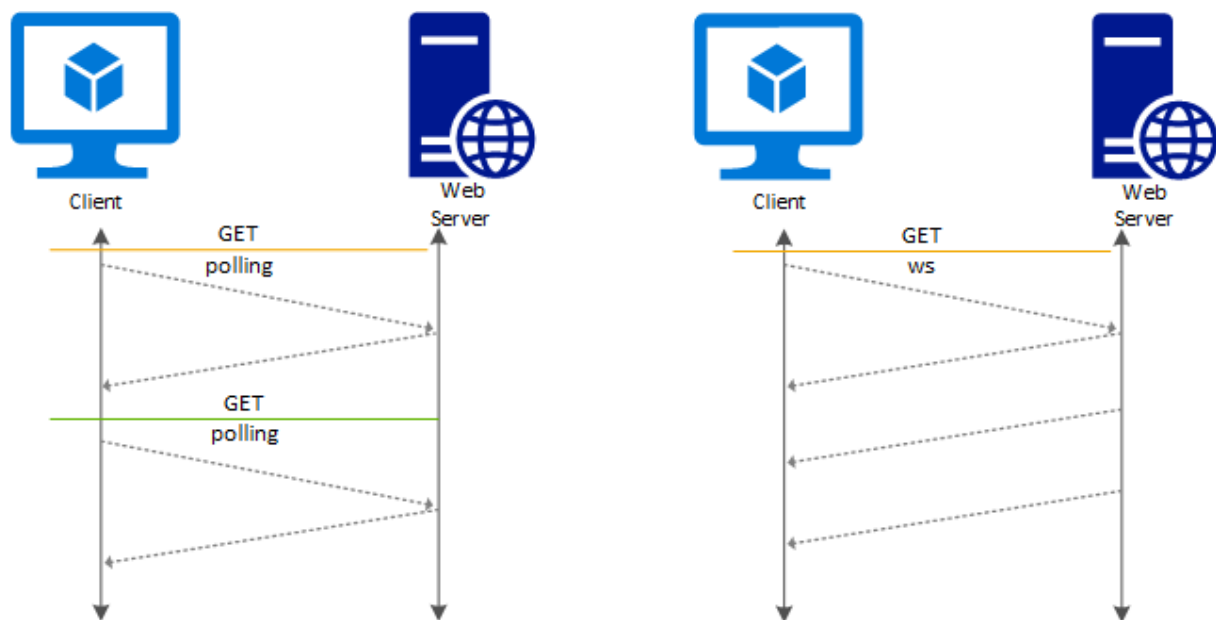
Projet 9 – B : Découverte de Socket.io et jQuery pour réaliser un Chat asynchrone :

Ressources :

- <https://socket.io/>
- <https://socket.io/get-started/chat>
- <https://github.com/michelonlineformapro/Projet-9-B-Socket-IO-jQuery>
- Sans socket : <https://github.com/michelonlineformapro/PHP-Ajax-Mysql-Chat>

Présentation des web-socket :

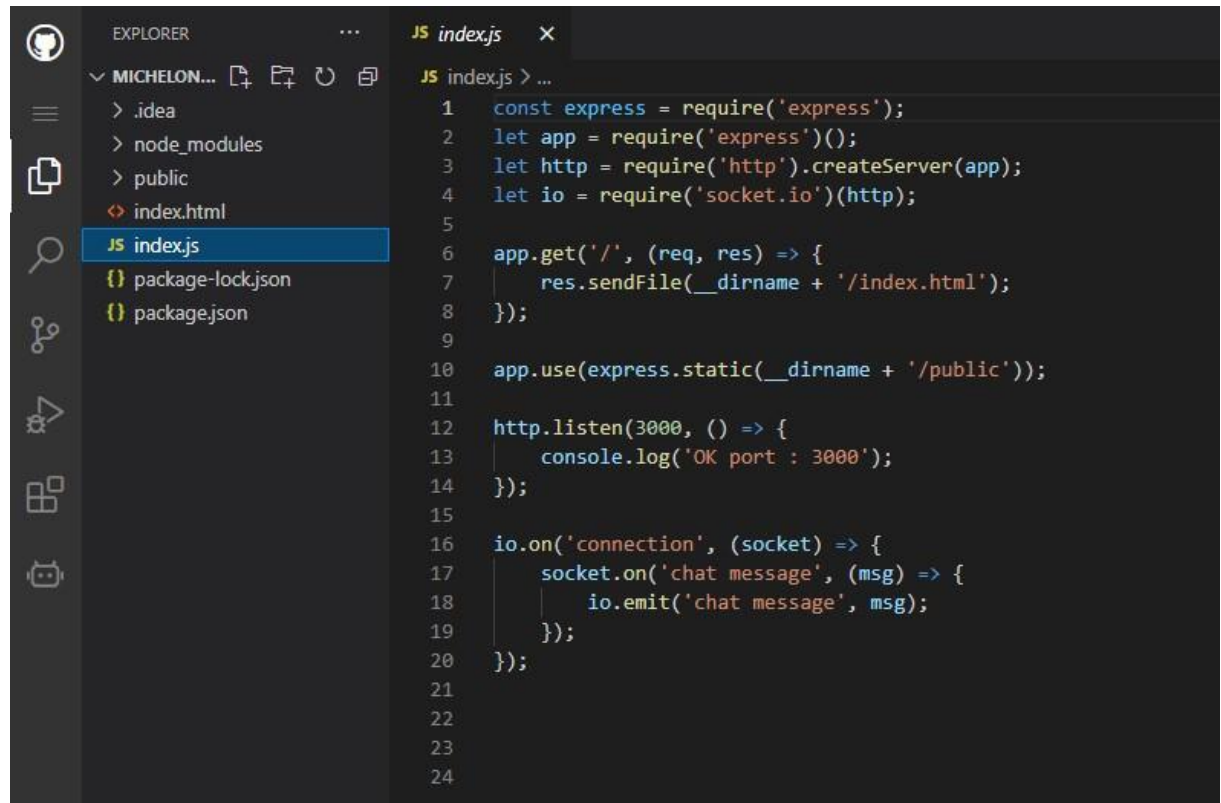
Web Socket est un standard du Web désignant un protocole réseau de la couche application et une interface de programmation du World Wide Web visant à créer des canaux de communication full-duplex par-dessus une connexion TCP pour les navigateurs web.



CREER LE CHAT :

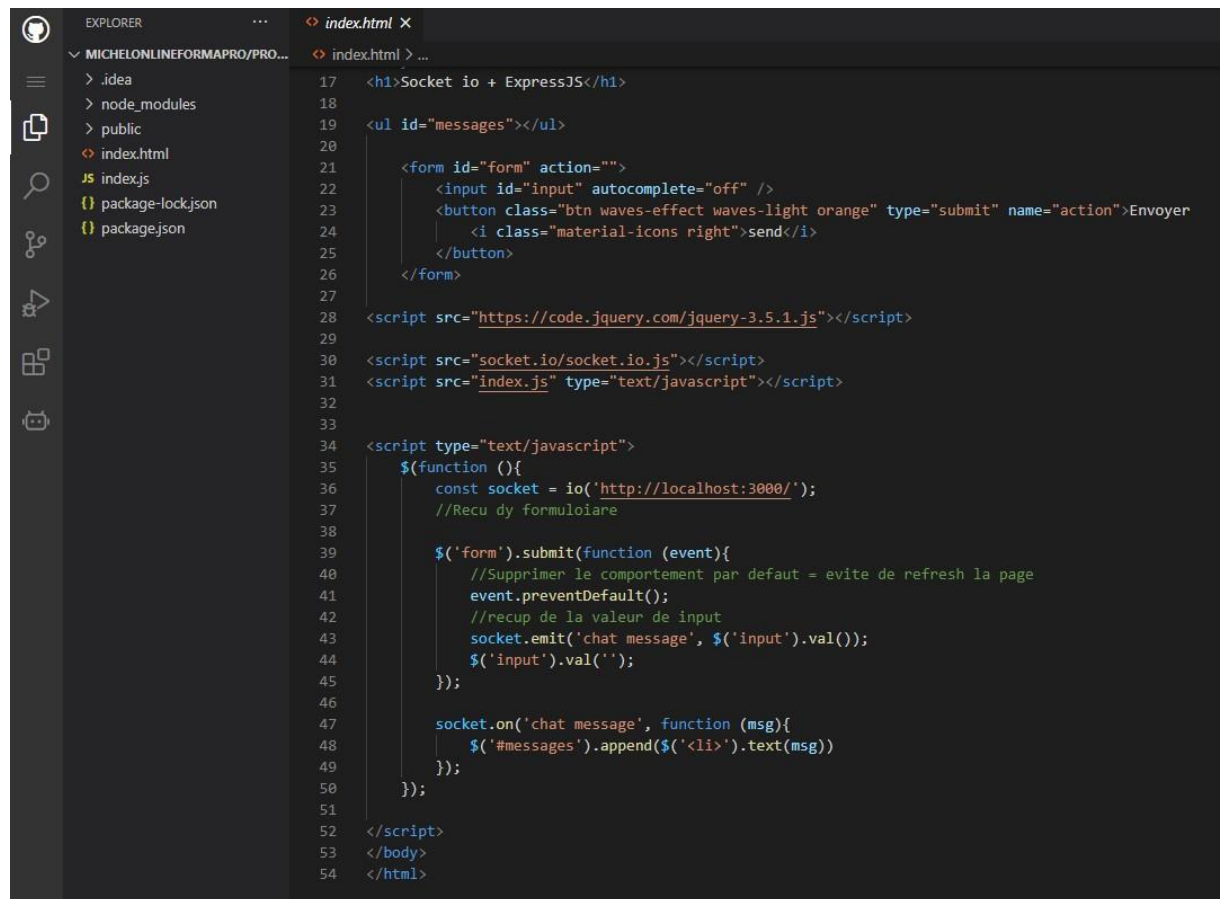
1. Introduction au Framework ExpressJs pour créer
2. <https://expressjs.com/fr/>
3. Retour sur NodeJs et les gestionnaires de paquet (npm, yarn, etc...)
4. Retour sur la gestion des dépendances package.json
5. Installation d' ExpressJs via npm
6. Création du fichier index.js, point d'entrée de l'application chat
7. Import du Framework et stock dans une constante
8. Instance du Framework
9. Import de http et création d'un server sur le port 3000
10. Utilisation de la méthode GET, création d'une route de base et création d'une promesse qui retourne de HTML
11. Appel du serveur sur le port 3000

12. Création d'un fichier index.html qui contient un formulaire avec un input texte + bouton de soumission
13. Installation de socket.io et test de connexion dans index.js



```
1  const express = require('express');
2  let app = require('express')();
3  let http = require('http').createServer(app);
4  let io = require('socket.io')(http);
5
6  app.get('/', (req, res) => {
7    res.sendFile(__dirname + '/index.html');
8  });
9
10 app.use(express.static(__dirname + '/public'));
11
12 http.listen(3000, () => {
13   console.log('OK port : 3000');
14 });
15
16 io.on('connection', (socket) => {
17   socket.on('chat message', (msg) => {
18     io.emit('chat message', msg);
19   });
20 });
21
22
23
24
```

14. Import de socket.js dans le fichier HTML et test de connexion – déconnexion
15. Emmettre des événements à l'aide du formulaire :



```
17 <h1>Socket io + ExpressJS</h1>
18
19 <ul id="messages"></ul>
20
21 <form id="form" action="">
22   <input id="input" autocomplete="off" />
23   <button class="btn waves-effect waves-light orange" type="submit" name="action">Envoyer
24     <i class="material-icons right">send</i>
25 </button>
26 </form>
27
28 <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
29
30 <script src="socket.io/socket.io.js"></script>
31 <script src="index.js" type="text/javascript"></script>
32
33
34 <script type="text/javascript">
35   $(function (){
36     const socket = io('http://localhost:3000/');
37     //Recu dy formulaire
38
39     $('form').submit(function (event){
40       //Supprimer le comportement par default = evite de refresh la page
41       event.preventDefault();
42       //recup de la valeur de input
43       socket.emit('chat message', $('input').val());
44       $('input').val('');
45     });
46
47     socket.on('chat message', function (msg){
48       $('#messages').append($('li').text(msg))
49     });
50   });
51
52 </script>
53 </body>
54 </html>
```

16. A chaque entrée utilisateur on génère une balise HTML qui affiche le texte entré.

Voici quelques idées pour améliorer l'application :

- Diffusez un message aux utilisateurs connectés lorsque quelqu'un se connecte où se déconnecte.
- Ajout de la prise en charge des surnoms.
- N'envoyez pas le même message à l'utilisateur qui l'a envoyé. Au lieu de cela, ajoutez le message directement dès qu'il appuie sur Entrée.
- Ajoutez la fonctionnalité « {user} est en train de taper ».
- Montrez qui est en ligne.
- Ajouter une messagerie privée.
- Partagez vos améliorations !