

---

## TP4 : WebSocket

Ayoub KARINE (ayoub.karine@isen-ouest.yncrea.fr)

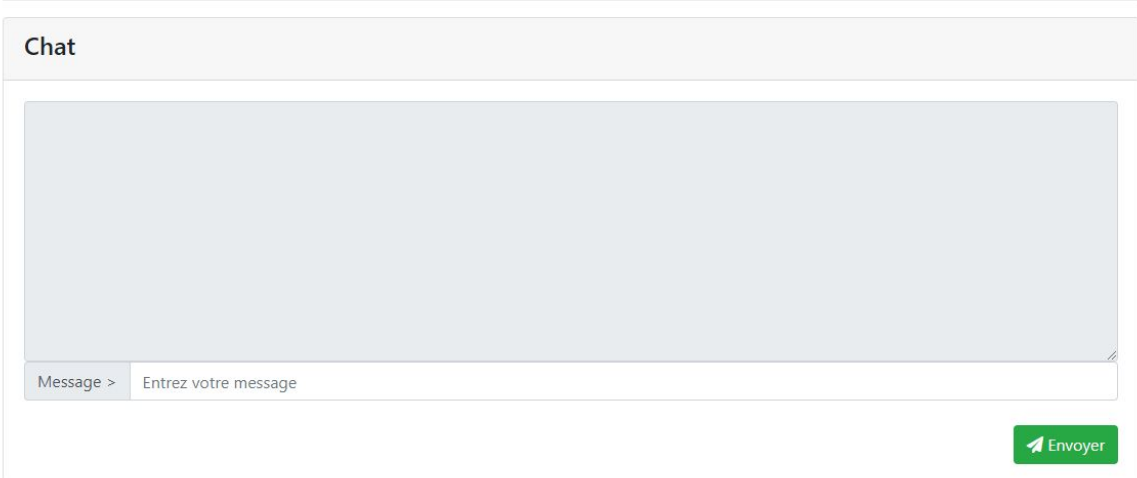
---

L'objectif de ce TP est de créer un chat en ligne en se basant sur les websockets

### Structure HTML du chat :

1. Créer un fichier index.html qui, en injectant des éléments Bootstrap (container, card, card-header, card-body, textarea, input, button), génère l'affichage ci-dessous :

#### TP n°4 WebSockets



The screenshot shows a web-based chat interface. At the top, there is a header with the word "Chat". Below this is a large, empty rectangular area for messages. At the bottom, there is a text input field with the placeholder text "Entrez votre message" and a green button labeled "Envoyer" with a paper plane icon.

### Implémentation côté client (JavaScript) :

Dans cette partie, on souhaite écrire un script "chat.js" permettant de dialoguer avec le serveur de chat au travers d'une WebSocket

2. Créer un script "chatClient.js" dans un dossier "js"

## Création du canal de communications :

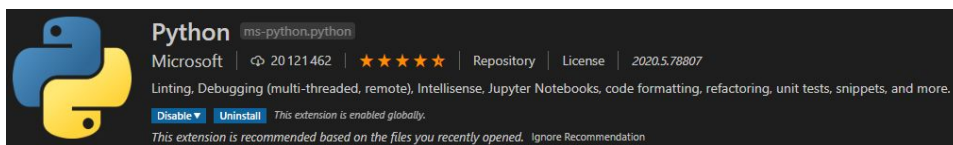
3. créer une variable globale websocket (elle va représenter le canal de communications entre le client et le serveur)
4. créer une variable globale login qui demande à l'utilisateur de saisir un login. Pour ce faire, utiliser la fonction prompt.
5. créer puis appeler une fonction createWebSocket sans arguments. Dans celle-ci :
  - a. initialiser la variable globale websocket avec le constructeur WebSocket en précisant comme adresse et port "localhost:12345"
  - b. appeler une fonction sendMessage une fois qu'on clique sur le bouton envoyer
  - c. ajouter un callback à l'évènement onmessage qui permettra d'ajouter, dans la fenêtre de chat (dans le textarea), les messages reçus du serveur

## Gestion de l'envoi d'un message :

6. Créer une fonction sendMessage(event) permettant :
  - a. de stopper la propagation de l'évènement submit du formulaire avec la fonction preventDefault()
  - b. de récupérer le message à envoyer
  - c. d'envoyer le message sous la forme "login : message". Exemple de message envoyé :  
**Dupont : Bonjour**
  - d. supprime le texte envoyé

## Implémentation côté serveur (Python) :

7. Installer l'extension Python sur VsCode ou utiliser Pycharm



8. Créer un fichier test.py dans le dossier python qui affiche un texte et tester l'interpréteur de VsCode (dans le cas d'utilisation de pycharm)

9. Installer le module websockets en utilisant le terminal inclut dans VScode ou dans Pycharm :

`pip install websockets`

10. Dans un fichier chatServer.py, créer une fonction clientHandler qui
- ajoute les websockets dans un tableau (ce tableau va être initialisé dans la partie test)
  - reçoit le message du chat et l'envoi à tous les clients. Pensez à utiliser une boucle while True
  - supprime le websocket associé au client dans le cas de la déconnexion de celui-ci
11. Ajouter dans le fichier chatServer.py le test ci-dessous :

```
clients = []
server = websockets.serve(clientHandler, "localhost", 12345)
asyncio.get_event_loop().run_until_complete(server)
asyncio.get_event_loop().run_forever()
```

12. Lancer le fichier python/CIRChatServer.py pour démarrer le serveur puis ouvrir la page index.html pour tester le chat
- dans un onglet du navigateur
  - dans 2 onglets du navigateur
  - dans plusieurs onglets du navigateur