
TP #3 – CONNEXION À UN CHAT COMMUN

8INF128 – Conception & programmation de sites web

Afin d'appliquer les notions vues en cours, le TP3 demande la création d'une page de chat. Celle-ci devra s'authentifier auprès d'un serveur afin de récupérer un token de connexion, récupérer les messages sur une API et dialoguer via un websocket.

Informations

Le serveur est sur le nom de domaine « kevin-chapron.fr » avec le port 8080. En fonction du endpoint et du protocole, certains comportements changent (API vs WebSocket).

Endpoint	Méthodes autorisées	Utilité
/login	POST, OPTIONS	<p>Permet à l'utilisateur de s'authentifier.</p> <p>Il doit recevoir un objet JSON ayant la clé « Code » et pour valeur votre code permanent.</p> <p>Il vous renverra soit une erreur, soit un objet contenant :</p> <ul style="list-style-type: none">- Code permanent (clé : Code)- Nom/Prénom (clé : Name)- Token (clé : Token)
/messages	GET, OPTIONS	<p>Permet d'avoir la liste des 50 derniers messages envoyés sur le chat.</p> <p>Nécessite d'être authentifié avec le header « Authorization » et le type « Basic » suivi du token renvoyé lors de l'authentification.</p> <p>En cas d'erreur, il renverra une erreur adaptée à votre requête.</p> <p>En cas de succès, vous aurez un tableau contenant plusieurs objets JSON formaté comme ceci :</p> <ul style="list-style-type: none">- « From » : Nom de l'expéditeur- « Date » : Date d'envoi du message (attention au format :)- « Text » : Texte envoyé par l'utilisateur

/ws	Connexion WebSocket	<p>Ouvre une connexion websocket avec le serveur.</p> <p>Cependant, celui-ci n'envoie absolument rien tant que vous ne vous êtes pas authentifié auprès de lui.</p> <p>Vous pouvez lui envoyer un message formaté en JSON. Il doit y avoir SOIT la clé « auth » contenant le token d'authentification, SOIT la clé « message » avec le message à envoyer.</p> <p>Si vous êtes authentifié avec succès, vous aurez un message vous le prouvant et vous recevrez les messages en temps réel (sous le même format que le /messages).</p>
-----	---------------------	---

Exigences

10% de la note

- Le code HTML doit être valide auprès du W3C (<https://validator.w3.org/>) et lisible (indentation correcte !)
- Le code CSS doit être facilement lisible (indentation correcte !)
- Aucune librairie externe (sauf pour des icônes) n'est tolérée

35% de la note

1. Le chat doit être centré horizontalement & verticalement. Pour ce faire, il faudra lui donner une largeur et une hauteur (*800x600 est celui avec lequel je l'ai fait*).
2. Le chat doit être formaté en deux parties : La première contiendra l'ensemble des messages envoyés, la deuxième contiendra le nom de l'utilisateur courant, une zone de texte (input, pas textarea) et un bouton d'envoi.
3. Le nom de l'utilisateur courant doit être en gras, à gauche de l'input. Il doit aussi être en petites majuscules.
4. L'input doit prendre la plus grande partie de la partie basse du chat.
5. Le bouton doit envoyer la donnée sur le chat (appuyer sur « Enter » sur l'input doit aussi envoyer la donnée)
6. Il faut que l'input soit stylisé.
7. Chaque message doit être formaté comme suit :

[2019-10-24 16:53:21] (*Kévin Chapron*) À vous de jouer ! :)

Il est possible de changer légèrement le CSS, mais il faut un caractère avant/après la date et le nom, ainsi qu'un style particulier pour la date & le nom (qu'ils soient différents l'un de l'autre).

55% de la note

1. Au chargement de la page, il faut automatiquement faire une requête avec votre code permanent pour récupérer le token d'authentification.
2. Une fois la requête placée et réussie, il faudra placer le nom de l'utilisateur dans la case prévue à cet effet.
3. Une fois le token récupéré, il faudra faire une requête sur le serveur afin de récupérer les messages et de les afficher dans le chat (le plus bas doit être le dernier, et doit TOUJOURS être visible ! Il faut ajuster la scrollbar en conséquence)
4. Une fois les messages affichés, il faut se connecter au websocket.
5. Il faudra ensuite s'authentifier auprès du websocket.
6. Une fois authentifié, la page doit être capable de recevoir des données (et les placer dans l'espace prévu à cet effet) et d'en émettre (les envoyer sur le websocket vous renverra votre message automatiquement).
7. Évidemment, tous les appels API & websocket doivent être complètement transparent pour l'utilisateur final.

Conseils

- Se faire une fonction pour faire des appels AJAX (prenant en paramètre la méthode, l'URL, les données & une éventuelle fonction de retour). Elle pourrait aussi gérer des headers au besoin !
- Se faire une fonction pour gérer un message entrant du chat (gestion de la date, adaptation du JSON vers le HTML, ajout du HTML généré dans le chat)
- Se faire des fichiers séparés pour les fonctionnalités (exemple : ajax.js, websocket.js, messageHandler.js) et un fichier principal (exemple : app.js) qui va ensuite appeler les fonctions des autres fichiers. Ne pas oublier de les inclure dans le HTML via la balise script, dans l'ordre !

Format de rendu


Le projet DOIT respecter cette structure de dossiers. Le fichier « index.html » est à la racine du dossier. À côté de celui-ci se trouvent les dossiers « css », « js » et « img ». Ils contiennent respectivement l'ensemble des fichiers CSS, l'ensemble des fichiers JavaScript et l'ensemble des images nécessaires au projet.

Pour être valide, il faut aussi rendre un unique fichier .zip comportant le dossier racine. Ce fichier devra aussi être nommé : « TP3_NOM_PRENOM.zip » en remplaçant bien entendu les « NOM » et « PRENOM » par vos noms et prénoms respectifs.

Date de rendu

Le projet devra être remis avec **un seul fichier .zip au plus tard le 11 Novembre à 23h59**. Le rendu se fera sur Moodle via l'option créée à cet effet.

Mon exemple !



The screenshot shows a chat window with a light gray background. At the top, there are three messages. The first message is from the 'Server' and says 'Server Started @ 2019-10-24T16:52:14-04:00'. The second message is from 'Kévin Chapron' and says 'À vous de jouer ! :)'. The third message is from 'Local' and says 'Vous êtes bien connecté !'. Below the messages is a large white text input area. At the bottom of the window, there is a footer bar. On the left, it says 'KÉVIN CHAPRON' in bold, followed by 'Votre message ...'. On the right, there is a gray button labeled 'Envoyer !'. A vertical scrollbar is visible on the right side of the message area.

[2019-10-24 16:52:14] (Server) Server Started @ 2019-10-24T16:52:14-04:00
[2019-10-24 16:53:21] (Kévin Chapron) À vous de jouer ! :)
[2019-10-29 00:34:39] (Local) Vous êtes bien connecté !

KÉVIN CHAPRON Votre message ...

Envoyer !