TME 7 – Implémentation des services

TME 7 – Implémentation des services

7.1 Serveur basique et axios

7.1.1 Création du serveur node

Avec Express, créer un serveur écoutant le port 8000 qui :
- renvoie au client la chaîne de caractères "Message reçu" à une requête à l'URL /
- affiche dans sa console le contenu de la propriété texte du `body` de la requête en JSON envoyée en POST à l'URL /

Tester les deux URL avec Postman.

7.1.2 Création du client React

Créer un composant React :
- possédant un champ de formulaire et un bouton d'envoi
- un état `champ` synchronisé avec la valeur du champ de formulaire
- au click sur le bouton, une requête POST est envoyée au serveur contenant le body `texte: champ`

À l'envoi, vous devriez obtenir une erreur CORS. Pour éviter cela, installez sur le serveur le module cors puis ajoutez dans le code les lignes

```
const cors = require('cors');
app.use(cors({origin:"*"}));
```

7.2 Premier service : Création d'utilisateurs

On va utiliser pour tous les services les codes de statut HTTP :
https://www.restapitutorial.com/httpstatuscodes.html.
Un message d'erreur plus détaillé peut être donné dans la réponse.
Pour rappel, voici la description du service de création d'utilisateur :

Nom du web service	CreateUser
URL du web service	/user/ avec PUT
Description du service	Permet la création d'un nouvel utilisateur
Paramètres en entrée	login ;password ;password2 ;nom ;prenom ;...
Format de sortie	ok, ou erreur
Exemple de sortie	`{status : XXX, message : "xxxxx", details: "yyyyy" }`, par exemple `{status : 201, message : "Utilisateur créé" }`
Erreurs possibles	requête mal formulée ; paramètres manquants ; password et password2 différents ; serveur base de données inaccessible ; utilisateur déjà existant
Avancement du Service	À faire
Classes/Fichiers JavaScript	Users.exists, Users.create
Informations additionnelles	xxxx

7.2.1 Donnez le JSON retourné en cas de succès et d'erreur, ainsi que le statut HTTP de la réponse

7.2.2 Écrire l'algorithme pour réaliser le service `createUser`, en considérant les connexions à la base de données.

7.3 Second service : Login

Vous allez utiliser le squelette fourni (fichier `server.zip`).
Nous allons considérer le système d'authentification suivant :

- Lors du *login*, une clef sera générée automatiquement par le serveur, et transmise sous forme de cookie. Si vous utilisez le *middleware* `express-session`, cela est fait automatiquement
- La clef aura une durée de vie déterminée

7.3.1 Spécifier le service *login*.

7.3.2 Écrire l'algorithme correspondant

7.3.3 Quelles sont les fonctions d'interrogation de la base de données dont vous allez avoir besoin ?

7.3.4 Écrire le service correspondant en JavaScript

7.4 Autres services

7.4.1 Faire la liste des services de base qui devront être implémentées dans votre projet. Les services seront regroupés par familles :

- authentification
- utilisateurs
- messages
- demandes

Pour chaque service, vous spécifierez les entrées et sorties. La spécification complète et documentée des services sera à compléter chez vous.