

---

## TP4 : API REST

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

---

1. Dans la WSL, démarrez le serveur apache
2. Dans la WSL, déplacez-vous dans `/var/www/html`, et puis tapez :  
`code` .
3. Créer un dossier TP2\_ComWeb dans le serveur apache

### Exercice1

Le but du TP est d'écrire le code PHP permettant de créer/accéder/mettre à jour/supprimer des Tweets stockés dans une base de données. Afin de gérer les Tweets, cinq requêtes, suivant le formalisme REST, sont effectuées par le code JavaScript "tweets.js":

**GET** `php/request.php/tweets/` Récupération des tweets

**GET** `php/request.php/tweets/?login=...` Récupération des tweets d'un user

**POST** `php/request.php/tweets/ login=...&text=...` Ajout d'un tweet

**PUT** `php/request.php/tweets/i login=...&text=...` Modification du tweet

**DELETE** `php/request.php/tweets/i?login=...` Suppression du tweet

Afin d'afficher correctement les Tweets, le code JavaScript attend, du serveur, une chaîne JSON de type :

```
[
  - {
    id: 1,
    text: "Un tweet des CIR1 !!",
    login: "cir1"
  },
  - {
    id: 2,
    text: "Un tweet des CIR2 !!",
    login: "cir2"
  },
  - {
    id: 3,
    text: "Un tweet des CIR3 !!",
    login: "cir3"
  },
  - {
    id: 4,
    text: "Un tweet des M1 !!",
    login: "m1"
  },
  - {
    id: 5,
    text: "Un tweet des M2 !!",
    login: "m2"
  }
]
```

### Initialisation de la base de données :

1. Téléchargez le dossier [tp2Input](#)
2. Lancez postgresql
3. Créez une base de données appelée tweetsdb
4. Connectez vous à cette base de données via SQLTools
5. Exécutez le fichier "sql.sql" afin de créer et remplir la table tweets. Vérifiez si la table a été bien créée en inspectant son contenu
6. Modifiez le fichier constants.php avec vos propres informations d'accès à la base de donnée "tweetsdb"

### Encodage de l'information à envoyer au client en utilisant PHP :

7. Découvrez le contenu du fichier database.php qui contient trois fonctions :
  - a. **dbConnect()** : pour se connecter à la base de données
  - b. **dbRequestTweets(...)** : qui permet de récupérer soit tous les tweets soit les tweets correspondants à un utilisateur
  - c. **dbAddTweet(...)** : pour ajouter un tweet
  - d. **dbModifyTweet(...)** : pour modifier le tweet
  - e. **dbDeleteTweet(...)** : pour supprimer le tweet
8. Ecrire un fichier "**request.php**" dans le dossier php qui permet de :

- a. établir une connexion avec la base de données
- b. récupérer le type de la requête (GET, POST, PUT ou DELETE) avec `$_SERVER['REQUEST_METHOD']`
- c. récupérer les informations sur le nom du chemin fourni par le client concernant le nom du fichier exécutant le script courant, sans la chaîne relative à la requête si elle existe. Par exemple, pour la requête :  
<http://localhost/CIR2/ComWeb/tp2/Sources/php/request.php/tweets>  
le contenu de la variable "\$Ressourcerequest" est bien "tweets"  
Ceci peut être obtenu via le code suivant :

```

$request = substr($_SERVER['PATH_INFO'], 1);
$request = explode('/', $request);
$requestRessource = array_shift($request);

```
- d. stocker dans une variable le retour de la fonction correspondante à chaque type de requête. Par exemple, cette variable va contenir le retour de la fonction `dbRequestTweets(..)` si la requête GET est demandée. Les variables globales à utiliser sont : `$_GET`, `$_POST` et `$_PUT`. Cette variable qui contient la réponse de la requête doit être encodé en JSON et envoyé au serveur par un simple affichage avec `echo`

### Requêtes AJAX du client en JS :

Les requêtes correspondantes aux 4 verbes CRUD ainsi que l'insertion de la réponse dans la page HTML sont déjà fournies dans les fichiers "ajax.js" et "tweets.js". Il est à noter, que dans le fichier tweets.js, la bibliothèque [jQuery](#) est utilisée pour gérer les événements et le DOM

9. Découvrez le contenu des fichiers "ajax.js" et "tweets.js"

10. En lançant index.html, testez si :

- a. Tous les tweets sont affichés en démarrant la page index.html et

aussi en cliquant sur  (utilisez `$_GET`) :

TP n°2 REST

Liste des tweets

cir1 : Un tweet des CIR1 !!

✎

✖

cir2 : Un tweet des CIR2 !!

✎

✖

cir3 : Un tweet des CIR3 !!

✎

✖

m1 : Un tweet des M1 !!

✎

✖

m2 : Un tweet des M2 !!


✎

✖



Tweet >



Entrez votre tweet

Envoyer


- b. En cliquant sur , les tweets du CIR2 sont affichés (utilisez \$\_GET):

TP n°2 REST

Liste de mes tweets  

cir2 : Un tweet des CIR2 !!  

Tweet > Entrez votre tweet


 Envoyer

- c. En saisissant un texte dans :


Tweet > Entrez votre tweet

Un tweet est ajouté pour le login cir2

Utilisez \$\_POST

- d. En cliquant sur , on peut modifier un tweet. Par défaut, PHP génère deux tableaux : \$\_GET pour les données passées dans l'URL (après le ?) et \$\_POST pour les données envoyées dans le xhr.send(data) avec POST. Dans le cas d'un envoi avec PUT, ce tableau n'est pas créé. Une solution pour récupérer les données envoyées par une requête PUT est d'utiliser la ligne de code suivante :

```
parse_str(file_get_contents('php://input'), $_PUT);
```

- e. En cliquant sur , on peut supprimer un tweet. \$\_DELETE n'existe pas dans PHP, mais vous pouvez utiliser \$\_GET pour récupérer les informations nécessaires.