

TP API REST

OBJECTIF : Réaliser une API REST complète permettant d'interfacer une base de données avec un site internet.

La base de données servant de support à ce TP est la base **lycee** et sa table **etudiant**.

Les fonctions à réaliser sont résumées dans le tableau suivant :

Route	Méthode	Type	Description
http://{url}/api/etudiants	GET	JSON	Récupérer tous les étudiants
http://{url}/api/etudiants/{id}	GET	JSON	Récupérer les données d'un seul étudiant
http://{url}/api/etudiants	POST	JSON	Insérer un nouvel étudiant
http://{url}/api/etudiants/{id}	PUT	JSON	Mettre à jour un étudiant
http://{url}/api/etudiants/{id}	DELETE	JSON	Supprimer un étudiant

1 - Point de départ

Nous avons créé un fichier **PHP** pour le back-end :

```
1  <?php
2      // Connexion
3      $servername = "127.0.0.1";
4      $username = "demo";
5      $password = "demo";
6      $dbname = "lycee";
7      $conn = mysqli_connect($servername, $username, $password, $dbname);
8
9      if (!$conn) {
10         die("Echec de connexion : ".mysqli_connect_error());
11     }
12     //echo("Connexion réussie<br/>");
13
14     // Accès aux données
15     $requete = "SELECT * FROM etudiant";
16     $result = mysqli_query($conn, $requete);
17     $etudiants = array();
18     if (mysqli_num_rows($result) > 0) {
19         while($row = mysqli_fetch_assoc($result)) {
20             $etudiant = array();
21             $etudiant['id'] = $row['idEtudiant'];
22             $etudiant['nom'] = $row['nom'];
23             $etudiant['prenom'] = $row['prenom'];
24             $etudiant['tel'] = $row['tel'];
25             $etudiant['pseudo'] = $row['pseudo'];
26             array_push($etudiants, $etudiant);
27         }
28         echo json_encode($etudiants);
29     }
30  ?>
```

- Ce programme renvoie un flux JSON contenant toute la table **etudiant** :

```
[
  - {
    id: "1",
    nom: "MARTON COULIS",
    prenom: "Wendy",
    tel: "0696192279",
    pseudo: "icetoice"
  },
  - {
    id: "2",
    nom: "HILLION",
    prenom: "Yannik",
    tel: "0696393833",
    pseudo: "Leygoman_972"
  },
  - {
    id: "3",
    nom: "LAGRAND",
    prenom: "Mathis",
    tel: "0767522009",
    pseudo: "xl_GhxstBlack"
  },
  - {
    id: "4",
    nom: "MAUGEE",
    prenom: "Killian",
    tel: "0696016389",
    pseudo: "KIKI_972"
  },
],
```

(capture incomplète)

2 – Création d'un nouveau fichier PHP qui ne renvoie qu'un seul étudiant choisi avec son **id** (clé primaire)

- La requête SQL permettant de récupérer un étudiant par sa clé primaire est :

```
SELECT * FROM `etudiant` WHERE `idEtudiant` = 3
```

- Important : On enregistre le fichier PHP sous un nouveau nom pour faire les modifications sur une copie.
On obtient cette version simplifiée dans laquelle il n'y a plus besoin du tableau **etudiants** car on ne veut plus qu'un seul étudiant:

```
13
14 // Accès aux données
15 $requete = "SELECT * FROM etudiant WHERE idEtudiant = 3";
16 $result = mysqli_query($conn, $requete);
17 if (mysqli_num_rows($result) > 0) {
18     while($row = mysqli_fetch_assoc($result)) {
19         $etudiant = array();
20         $etudiant['id'] = $row['idEtudiant'];
21         $etudiant['nom'] = $row['nom'];
22         $etudiant['prenom'] = $row['prenom'];
23         $etudiant['tel'] = $row['tel'];
24         $etudiant['pseudo'] = $row['pseudo'];
25     }
26     echo json_encode($etudiant);
27 }
```

- L'inconvénient de ce programme est que pour voir un autre étudiant, il faudra modifier la valeur de la clé primaire dans le code PHP.
- Solution : On va préciser la valeur de cette clé primaire en la passant en paramètre par l'url (Méthode **GET** : revoir l'exo6 en PHP pour le calcul du volume d'un cône)

- Modifications à effectuer :

```
14 // Accès aux données
15 $id = $_GET['id'];
16 $requete = "SELECT * FROM etudiant WHERE idEtudiant = ".$id;
```

- Pour tester cette modification, ne pas oublier de préciser la valeur de **id** dans l'URL :

← → ↻ 🏠 ⚠ Non sécurisé | 10.10.13.115/exos/bdd/demo4.php?id=4

```
{
    id: "4",
    nom: "MAUGEE",
    prenom: "Killian",
    tel: "0696016389",
    pseudo: "KIKI_972"
}
```

5 – Ajout d'un nouvel étudiant dans la table :

- Exemple de requête SQL permettant d'insérer un nouvel **enregistrement** à la table **etudiant** obtenue à partir de l'onglet **Insérer** de **phpMyAdmin** :

Serveur: MariaDB 10 » Base de données: lycee » Table: etudiant

Parcourir Structure SQL Rechercher Insérer Exporter Importer

Colonne	Type	Fonction	Null	Valeur
idEtudiant	int(11)	<input type="text"/>	<input type="text"/>	
nom	varchar(45)	<input type="text"/>	<input type="text"/>	BOBÉ
prenom	varchar(45)	<input type="text"/>	<input type="text"/>	Wilson
tel	varchar(11)	<input type="text"/>	<input type="text"/>	
pseudo	varchar(40)	<input type="text"/>	<input type="text"/>	

Exécuter

```
INSERT INTO `etudiant` (`idEtudiant`, `nom`, `prenom`, `tel`, `pseudo`) VALUES (NULL, 'BOBÉ', 'Wilson', '', '');
```

- Pour cette nouvelle fonction, 2 fichiers seront nécessaires :
 - Un formulaire codé en HTML pour saisir les différents champs du nouvel étudiant :

← → ↻ 🏠 ⚠ Non sécurisé | 10.10.13.115/exos/bdd/formEtudiant.html

Nom :

Prénom :

Téléphone :

Pseudo :

- Un fichier PHP appelé par le clic sur **Enregistrer**.
Ces données sont passées au programme PHP (Méthode **POST** : revoir l'exo7 en HTML et en PHP pour le calcul du volume d'un cône) qui va ajouter le nouvel **enregistrement** dans la table **etudiant** :

```

14 // Récupération des données aux données
15 $nom = $_POST['nom'];
16 $prenom = $_POST['prenom'];
17 $tel = $_POST['telephone'];
18 $pseudo = $_POST['pseudo'];
19 $requete = "INSERT INTO `etudiant` (`idEtudiant`, `nom`, `prenom`, `tel`, `pseudo`)
20 | VALUES (NULL, '$nom', '$prenom', '$tel', '$pseudo')";
21 $result = mysqli_query($conn, $requete);

```

4 – Regroupement des fonctions en un seul fichier PHP :

Nous sommes arrivés au stade où les 3 premières fonctions de l'API sont réalisées :

Route	Méthode	Type	Description
http://{url}/api/etudiants	GET	JSON	Récupérer tous les étudiants
http://{url}/api/etudiants/{id}	GET	JSON	Récupérer les données d'un seul étudiant
http://{url}/api/etudiants	POST	JSON	Insérer un nouvel étudiant
http://{url}/api/etudiants/{id}	PUT	JSON	Mettre à jour un étudiant
http://{url}/api/etudiants/{id}	DELETE	JSON	Supprimer un étudiant

- Nous allons regrouper les deux premières fonctions dans un seul fichier **etudiants.php** qui sera l'API ou le **back-end** :

```

6  function getEtudiants() {
7      global $conn;
8      $requete = "SELECT * FROM etudiant";
9      $result = mysqli_query($conn, $requete);
10     $etudiants = array();
11     if (mysqli_num_rows($result) > 0) {
12         while($row = mysqli_fetch_assoc($result)) {
13             $etudiant = array();
14             $etudiant['id'] = $row['idEtudiant'];
15             $etudiant['nom'] = $row['nom'];
16             $etudiant['prenom'] = $row['prenom'];
17             $etudiant['tel'] = $row['tel'];
18             $etudiant['pseudo'] = $row['pseudo'];
19             array_push($etudiants, $etudiant);
20         }
21         echo json_encode($etudiants);
22     }
23 }

25 function getEtudiant($id=0) {
26     global $conn;
27     $requete = "SELECT * FROM etudiant WHERE idEtudiant = ".$id;
28     $result = mysqli_query($conn, $requete);
29     $etudiant = array();
30     if (mysqli_num_rows($result) > 0) {
31         while($row = mysqli_fetch_assoc($result)) {
32             $etudiant = array();
33             $etudiant['id'] = $row['idEtudiant'];
34             $etudiant['nom'] = $row['nom'];
35             $etudiant['prenom'] = $row['prenom'];
36             $etudiant['tel'] = $row['tel'];
37             $etudiant['pseudo'] = $row['pseudo'];
38         }
39         echo json_encode($etudiant);
40     }
41 }

```

- Ensuite on se base sur la présence ou l'absence du paramètre **id** dans l'url pour choisir une des 2 fonctions :

```

56     if (empty($_GET["id"])) {
57         getEtudiants();
58     }
59     else {
60         $id = intval($_GET["id"]);
61         getEtudiant($id);
62     }

```

- Pour intégrer la 3^e fonction (insertion d'un nouvel étudiant), il faudra savoir si la méthode utilisée est **GET** ou **POST** en utilisant cette instruction :

```

4     $method = $_SERVER["REQUEST_METHOD"];

```

- Puis choisir la fonction correspondante :

```

55     switch($method) {
56         case 'GET' :
57             if (empty($_GET["id"])) {
58                 getEtudiants();
59             }
60             else {
61                 $id = intval($_GET["id"]);
62                 getEtudiant($id);
63             }
64             break;
65         case 'POST' :
66             addEtudiant();
67             break;
68     }

```

Travail supplémentaire à effectuer :

- Réaliser les deux dernières fonctions dans le back-end :
 - Modification d'un étudiant
 - Suppression d'un étudiant
- Solution possible pour le front-end :
 - Mettre un lien sur le nom de chaque étudiant dans le tableau affichant la liste des étudiants, qui renvoie vers un formulaire pré-rempli avec les données de l'étudiant choisi
 - À partir de ce formulaire, on pourra effectuer des modifications, puis valider avec un bouton.
 - Toujours sur la page de ce formulaire, un autre bouton permettra de supprimer l'étudiant choisi.
- Empêcher la visualisation du contenu du dossier de l'API
- Ajouter le fichier **.htaccess** permettant d'avoir des liens simplifiés de type :

<http://127.0.0.1/bdd/api/etudiant/2>

au lieu de :

<http://127.0.0.1/bdd/api/etudiants.php?id=2>