

PHP

MySQL

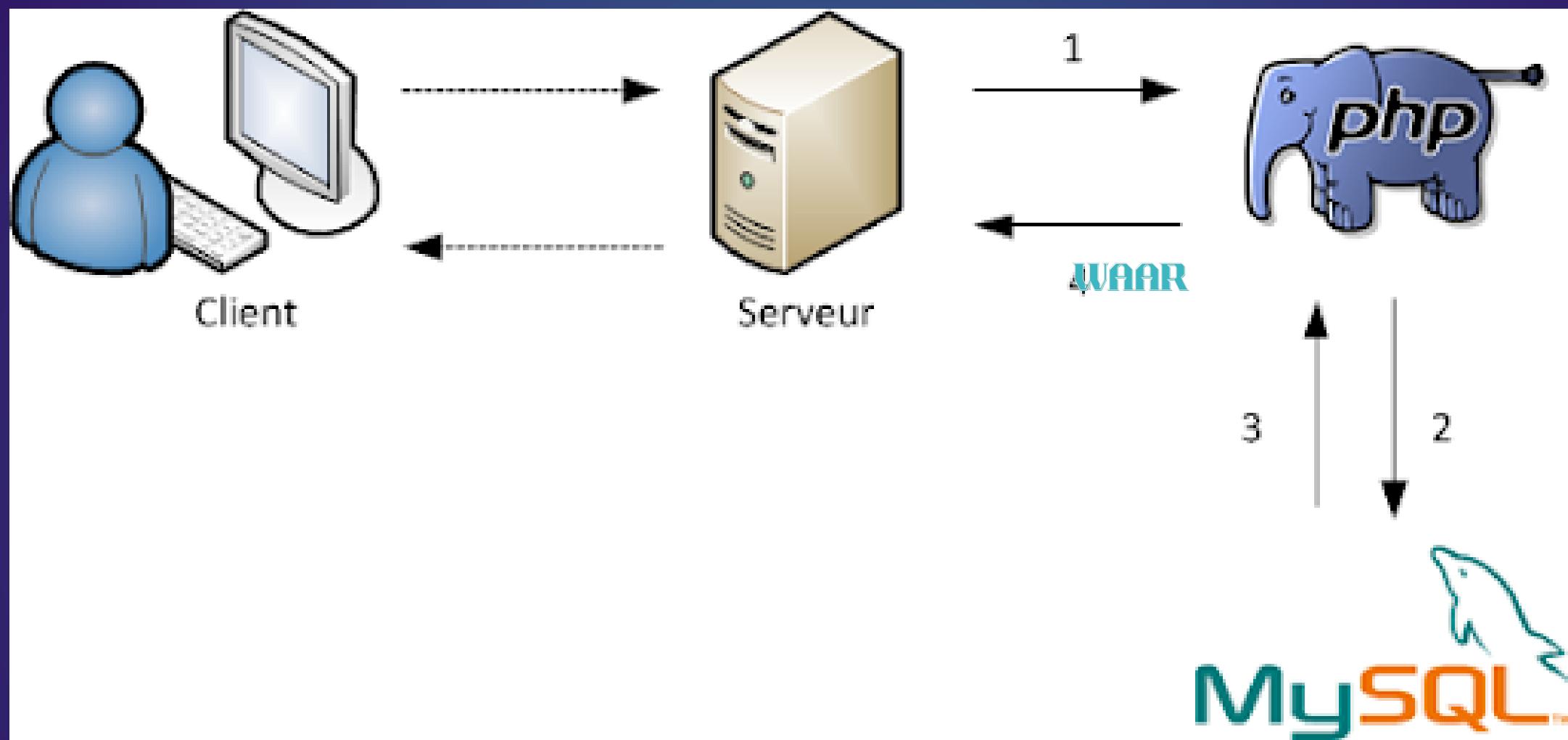
SQL



Comment ça marche?

[Back to Navigation Page](#)

Coté server side



Création de la base de données via MySQL avec SQL

1

VIA MY SQL

2

CREATION DE LA
DATABASE

3

CREATION DES TABLES

4

CREATION DES TABLES
INTERMEDIAIRES

5

UTILISATION DE
REQUETES POUR
EFFECTUER DES
JOINTURES OU UTILISER
DES CONTRAINTES

VIEW TABLE MYSQL

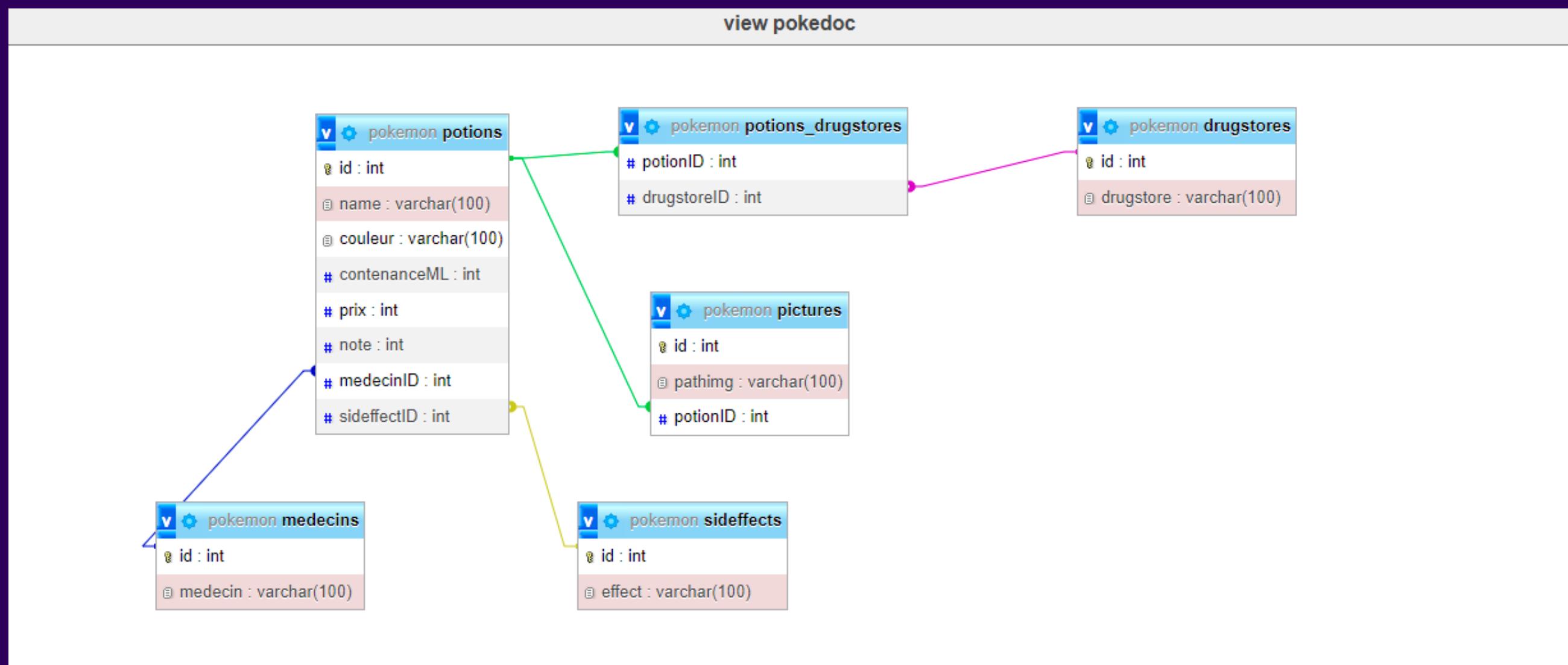
```
|1> SELECT*FROM potions;
+-----+-----+-----+-----+-----+-----+
| name | couleur | contenanceML | prix | note | medecinID | sideffectID |
+-----+-----+-----+-----+-----+-----+
| guerison | violet | 50 | 700 | 9 | 1 | 1 |
| force | vert | 30 | 300 | 6 | 2 | 2 |
| vitesse | bleu | 40 | 500 | 8 | 3 | 3 |
| defense | rose | 60 | 800 | 10 | 4 | 4 |
| capture | orange | 80 | 800 | 8 | 5 | 1 |
| transformation | rouge | 90 | 600 | 7 | 6 | 2 |
+-----+-----+-----+-----+-----+-----+
1 rows in set (0.00 sec)
```

VIEW TABLE

PHPmyADMIN

	← T →		▼	id	name	couleur	contenanceML	prix	note	medecinID	sideffectID	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	guerison	violet	50	700	9	1	1	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	force	vert	30	300	6	2	2	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	vitesse	bleu	40	500	8	3	3	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	4	defense	rose	60	800	10	4	4	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	5	capture	orange	80	800	8	5	1	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	6	transformation	rouge	90	600	7	6	2	

VIEW de notre base de données



Connexion à la base de données

```
● ● ●  
1 <!-- On definit nos variables d'environnement pour se connecter à la base de données MySQL -->  
2 <?php  
3   $config = [  
4     'dbhost' => 'localhost',  
5     'dbname' => 'pokemon',  
6     'dbport' => '3306',  
7     'dbuser' => 'root',  
8     'dbpass' => ''  
9   ];  
10  
11
```

Connexion à la base de données

```
1 <!-- connexion à la base de données avec l objet PDO -->
2 <?php
3
4 // $config prend les valeurs du tableau pour la connexion à la base de données
5 function getPDOlink ($config){
6
7 // Etape 2 Data Source Name de connexion :
8 $dsn = 'mysql:dbname=' . $config['dbname'] . ';host=' . $config['dbhost'] . ';port=' . $config['dbport'];
9 // echo $dsn;
10
11 // Etape 3 try and catch On tente de se connecter à la base de données
12 // try = essai de se connecter à la base de données
13 try {
14
15 // Etape 4 On instancie l'objet PDO avec 3 éléments et on crée notre variable $db, si la connexion échoue un message d'erreur apparaîtra:
16 $db = new PDO($dsn, $config['dbuser'], $config['dbpass']);
17
18 // Etape 5 si la connexion réussie execute = On envoie notre requête en utf8 pour afficher les caractères spéciaux :
19 $db->exec("SET NAMES utf8");
20
21 // Etape 6 setAttribute (méthode de PDO) récupère donne lui cette attribut
22 // FETCH_ASSOC évite la répétition // à chaque fetch par défaut fais le en fetch assoc
23 $db->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
24
25 return $db;
26 // traque/attrape moi l'erreur et montre le(s) moi / $e = variable et représentation de PDOException
27 } catch (PDOException $e) {
28 // -> sert à accéder au détail de l'erreur
29 exit('BDD Erreur de connexion : ' . $e->getMessage());
30 }
31 }
```

Utilisation require et dirname **_DIR_**

dirname_DIR_ est utilisé lorsque le fichier actuel n'est pas dans le dossier source
DIR utilisé lorsque le fichier est dans le dossier source

```
● ● ●  
1 <?php  
2 // REQUIRE_ONCE dirname(__DIR__) . nous permet d'intégrer un fichier et ses propriétés et évite  
3 // les doublons de déclaration.  
4 // dirname(__DIR__) .= le fichier actuel est dans un dossier du dossier racine (on lui indique le chemin)  
5 require_once dirname(__DIR__) . ('/function/database.fn.php');  
6 require_once dirname(__DIR__) . ('/config/config.php');  
7 $db = getPDOlink($config);  
8 ?>
```

```
● ● ●  
1 <?php  
2 // REQUIRE_ONCE __DIR__ nous permet d'intégrer un fichier et ses propriétés et évite  
3 // les doublons de déclaration.  
4 // __DIR__ = le fichier actuel est dans le fichier racine  
5 require_once __DIR__ . "/utilities/header.php";  
6 require_once __DIR__ . "/function/index.fn.php";  
7  
8 // var_dump (getPDOlink ($config));  
9  
10 ?>
```

REQUIRE_ONCE

un seul fichier ne peut être lu qu'une seule fois

**REQUIRE : il affiche l' erreur fatale dans
L'index PHP et n'affichera pas le code
entierement**

Utilisation des fonctions et requêtes SQL

Nous permet d'éviter la répétition et d'utiliser les requêtes SQL pour lier les tables entre elles et utiliser les contraintes

INNER JOIN/ JOIN

WHERE

```
1  <!-- mapage -->
2  <!-- création d'une fonction pour afficher les potions en fonction de l'ID
3  utilisation des inner join afin de lier les tables entre elles afin de récupérer les champs et les appeler dans notre html
4  -->
5  <?php
6  function findPotionsById($db,$currentId){
7      $sql = "SELECT p.id,p.name, p.couleur,p.contenanceML,p.prix,p.note,
8          m.medecin AS medecins,
9          s.effect AS sideeffects,
10         GROUP_CONCAT(d.drugstore SEPARATOR ', ')
11        AS drugstores FROM potions AS p
12        INNER JOIN medecins m ON p.medecinID = m.id
13        INNER JOIN sideeffects s ON p.sideeffectID = s.id
14        INNER JOIN potions_drugstores pd ON p.id = pd.potionID
15        JOIN drugstores d ON pd.drugstoreId = d.id
16        WHERE p.id = $currentId;";
17        $requete = $db->query($sql);
18        $potions = $requete->fetch();
19        return $potions;
20    }
21
22 // <!-- création d'une fonction pour afficher les photos en fonction de la potion
23 // utilisation de la requête SELECT*FROM pour récupérer les photos en fonction de l'id de la potion
24 // -->
25 function findPictureByPotions($db){
26     $sql = 'SELECT * FROM pictures WHERE potionID';
27     $requete = $db->query($sql);
28     $pictures = $requete->fetch();
29     return $pictures;
30 }
31
32
```

Utilisation des fonctions et requêtes SQL

ASC



```
1 <!-- PAGE index -->
2 <?php
3 // fonction pour récupérer la bdd
4 // Cette fonction prend deux paramètres : $db qui est une instance de la classe PDO définie dans le try/catch
5 // et $limit qui est le (un chiffre entier) nombre de films à afficher par page.
6 function findBestpotions($db, $limit) {
7     // On select la table potions SQL
8     // ORDER BY permet de mettre les potions de la moins cher à la plus cher
9     // On utilise la clause LIMIT pour ne sélectionner un nombre défini de potion de +cher à la - cher
10    $sql = "SELECT * FROM potions ORDER BY prix ASC LIMIT $limit;";
11
12    // On exécute la requête SQL en utilisant la méthode query() de l'objet PDO
13    // = fait cette requête sur cette base de données
14    $requete = $db->query($sql);
15
16    // On récupère les résultats de la requête en utilisant la méthode fetchAll() qui parcourt la table et stocke
17    // les éléments dans un tableau de l'objet PDOStatement
18    $potions = $requete->fetchAll();
19
20    return $potions;
21 }
```

Nous permet d'éviter la répétition et d'utiliser les requêtes SQL pour lier les tables entre elles et utiliser les contraintes

Utilisation des fonctions et requêtes SQL

Appel de nos champs liés à notre base de données grâce à la requête effectuer dans notre fonction

```
•••
1  <!-- page MAPAGE -->
2  <!-- cards des potions -->
3  <div class="container col-6 mb-5">
4      <div class="card text-center">
5          <h1>Détail</h1>
6          <!-- image-->
7          <img class="card-img-top" src="= $path ?&gt;" alt="..." /&gt;
8          &lt;div class="card-body p-4"&gt;
9              &lt;div class="text-center"&gt;
10                 &lt;h4&gt;
11                     &lt;?= $potions['name'] ?&gt;
12                 &lt;/h4&gt;
13                 &lt;div class="d-flex justify-content-center small text-warning mb-2"&gt;
14                     &lt;?= $potions['note'] . ' / 10 &nbsp;'; ?&gt;
15                 &lt;/div&gt;
16                 &lt;p class="fw-bold"&gt;Par :
17                     &lt;?= $potions['medecins'] ?&gt;
18                 &lt;/p&gt;
19                 &lt;p&gt;Drugstore :
20                     &lt;?= $potions['drugstores'] ?&gt;
21                 &lt;/p&gt;
22                 &lt;div class="d-flex justify-content-around my-2"&gt;
23                     &lt;span class="badge bg-primary"&gt;
24                         &lt;p&gt;Prix :
25                             &lt;?= $potions['prix'] ?&gt;
26                         &lt;/p&gt;
27                     &lt;/span&gt;
28                     &lt;span class="badge bg-primary"&gt;
29                         &lt;p&gt;Effet Secondaire :
30                             &lt;?= $potions['sideeffects'] ?&gt;
31                         &lt;/p&gt;
32                     &lt;/span&gt;
33                 &lt;/div&gt;
34             &lt;/div&gt;
35         &lt;/div&gt;
36     &lt;/div&gt;
37 &lt;/div&gt;</pre
```

Difficultés :

Adapter le code à mon projet

Appris:

Beaucoup mieux comprendre PHP

Aimé:

Entraide

Sources :

Utopia

W3C

Doc SQL

README