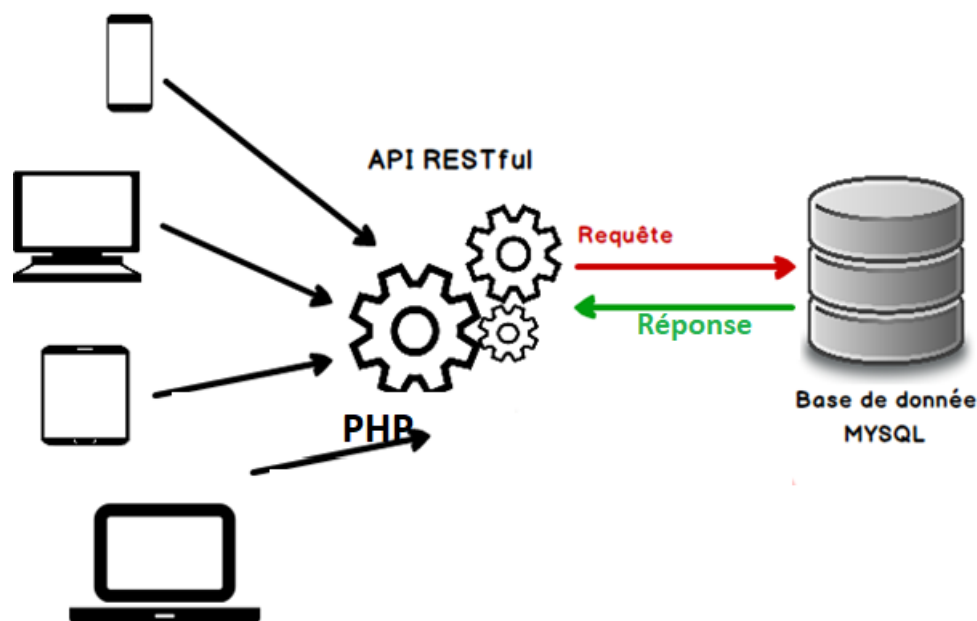


Créer et utiliser une API REST en PHP

REST (Representational State Transfer) est l'un des moyens pour accéder aux Web services. L'API REST est utilisée pour effectuer une requête HTTP GET, POST, PUT ou DELETE côté client vers le serveur pour récupérer ou pour modifier certaines informations sur le serveur.



Les services Web basés sur REST peuvent produire une sortie dans n'importe quel format, tel que CSV, JSON, RSS, etc. Cela dépend donc du format que vous souhaitez analyser facilement avec votre langage. Dans ce tutoriel, nous allons créer une API REST simple à l'aide du langage PHP.

Commençons par créer un exemple d'API rest, nous allons créer un répertoire 'api/'. Ce répertoire contient tous les fichiers nécessaires pour ce tutoriel. Nous créerons un api rest pour le module produit qui aura la requête HTTP Get, Post, Put et Delete pour récupérer, ajouter, mettre à jour et supprimer les enregistrements de mysql. Les détails de l'API rest sont les suivants:

Route	Méthode	Type	Description
http://127.0.0.1/api/produits	GET	JSON	Récupérer toutes les produits.
http://127.0.0.1/api/produits/{id}	GET	JSON	Récupérer les données d'un seul produit
http://127.0.0.1/api/produits	POST	JSON	Insérer un nouveau produit dans la base de données.
http://127.0.0.1/api/produits/{id}	PUT	JSON	Mettre à jour un produit dans la base de données.

Route	Méthode	Type	Description
http://127.0.0.1/api/produits/{id}	DELETE	JSON	Supprimer un produit de la base de données.

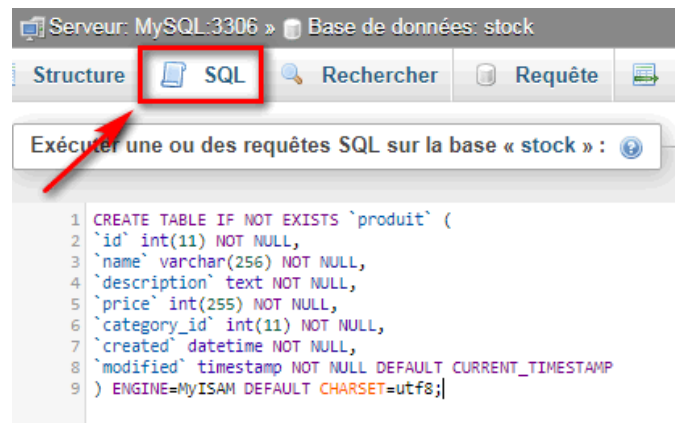
Nous allons utiliser les fichiers suivants pour ce chapitre:

- **index.php**: Ce fichier est un fichier d'entrée. Ce fichier empêche la navigation dans les fichiers de répertoire.
- **db_connect.php**: Ce fichier utilisera la chaîne de connexion mysql.
- **produits.php**: Ce fichier contient tous les méthode d'API REST.
- **post.php**: Ce fichier permet d'envoyer une requête POST à notre API REST pour tester l'ajout d'un nouveau produit.
- **put.php**: Ce fichier permet d'envoyer une requête PUT à notre API REST pour tester la mise à jour d'un nouveau produit.
- **delete.php**: Ce fichier permet d'envoyer une requête DELETE à notre API REST pour tester la suppression d'un nouveau produit.

Schéma de la table des produits:

Créez une base de données MySQL nommée 'stock' par exemple. Exécutez le code ci-dessous dans la fenêtre de requête SQL de la base de données. Comme illustré dans l'image ci-dessous :

```
CREATE TABLE IF NOT EXISTS `produit` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(256) NOT NULL,
  `description` text NOT NULL,
  `price` int(255) NOT NULL,
  `category_id` int(11) NOT NULL,
  `created` datetime,
  `modified` timestamp DEFAULT CURRENT_TIMESTAMP,
  primary key (id)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```



Pour le moment, ajouter ces enregistrements manuellement, parce que nous allons voir comment les ajoutés en utilisant l'api Rest.

```
INSERT INTO produit VALUES (1, 'PEN', 'Pen Blue', 1.23, 1, '2021-7-04', '2021-9-03');
INSERT INTO produit VALUES (2, 'PC', 'Apple products', 300.99, 2, '2021-01', '2021-5-14');
INSERT INTO produit VALUES (3, 'Car', 'Mercedes benz', 985470.47, 3, '2021-2-06', '2021-6-23');
```

Connexion à la base de données MySQL avec PHP

Après avoir créé la table « produit », nous allons créer le fichier db_connect.php pour établir une connexion avec la base de données MySQL.

```
<?php
$server = "localhost";
$username = "root";
$password = "";
$db = "stock";
$conn = mysqli_connect($server, $username, $password, $db);
?>
```

API REST pour récupérer tous les enregistrements de MySQL

Nous allons créer une requête HTTP de type GET pour accéder à tous les enregistrements de produits à partir de MySQL. Nous allons utiliser une requête

MySQL pour récupérer les données de la table « produit » et envoyer les données JSON au client en tant qu'objet.

Nous allons maintenant créer le fichier produits.php et inclure le fichier de connexion MySQL pour accéder aux données de la base de données.

```
<?php
// Se connecter à la base de données
include("db_connect.php");
$request_method = $_SERVER["REQUEST_METHOD"];
...
?>
```

Nous avons également utilisé la méthode **\$_SERVER** pour accéder aux informations sur l'action, comme l'ajout, la modification ou la suppression.

Dans le même fichier, Nous allons « switcher » entre les méthodes de requête(comme GET, POST, DELETE et PUT) à l'aide de l'instruction « switch » en PHP :

```
<?php
switch($request_method)
{
case 'GET':
if(!empty($_GET["id"]))
{
// Récupérer un seul produit
$id = intval($_GET["id"]);
getProducts($id);
}
else
{
// Récupérer tous les produits
getProducts();
}
break;
default:
// Requête invalide
header("HTTP/1.0 405 Method Not Allowed");
break;
}
```

?>

Nous avons créé une requête GET pour extraire toutes les données relatives aux produits de la base de données mysql. Pour les données relatives à un seul produit, nous transmettons l'id du produit. Nous avons défini la méthode `getProducts()` afin que nous puissions créer une méthode comme ci-dessous:

```
<?php
function getProducts()
{
    global $conn;
    $query = "SELECT * FROM produit";
    $response = array();
    $result = mysqli_query($conn, $query);
    while($row = mysqli_fetch_array($result))
    {
        $response[] = $row;
    }
    header('Content-Type: application/json');
    echo json_encode($response, JSON_PRETTY_PRINT);
}
?>
```

La méthode **`mysqli_query()`** récupère les données de la table « produit » de MySQL et les stocke comme objet dans la variable 'result'. La méthode **`json_encode()`** convertit un tableau en json.

Maintenant, accédez à l'url « <http://127.0.0.1/api/produits> » et vous obtiendrez un enregistrement de tous les produits de la table « produit ». Comme illustré dans l'image ci-dessous :

```
← → ↻ 🏠 ⓘ 127.0.0.1/api/produits

[
  {
    "0": "1",
    "id": "1",
    "1": "PEN",
    "name": "PEN",
    "2": "Pen Blue",
    "description": "Pen Blue",
    "3": "1",
    "price": "1",
    "4": "1",
    "category_id": "1",
    "5": "2020-07-04 00:00:00",
    "created": "2020-07-04 00:00:00",
    "6": "2020-09-03 00:00:00",
    "modified": "2020-09-03 00:00:00"
  },
  {
    "0": "2",
    "id": "2",
    "1": "PC",
    "name": "PC",
    "2": "Apple products",
    "description": "Apple products",
    "3": "301",
    "price": "301",
    "4": "2",
    "category_id": "2",
    "5": "2019-01-01 00:00:00",
    "created": "2019-01-01 00:00:00",
    "6": "2019-05-14 00:00:00",
    "modified": "2019-05-14 00:00:00"
  },
  {
    "0": "3",
    "id": "3",
    "1": "Car",
    "name": "Car"
  }
]
```

Api REST pour récupérer un seul produit

Nous allons créer une requête de type HTTP GET pour accéder à un seul enregistrement de produit à partir de la base de données MySQL via php.

```
function getProduct($id=0)
{
    global $conn;
    $query = "SELECT * FROM produit";
    if($id != 0)
    {
        $query .= " WHERE id=".$id." LIMIT 1";
    }
    $response = array();
    $result = mysqli_query($conn, $query);
    while($row = mysqli_fetch_array($result))
```

```
{
$response[] = $row;
}
header('Content-Type: application/json');
echo json_encode($response, JSON_PRETTY_PRINT);
}
```

Maintenant, accédez à l'url « <http://127.0.0.1/api/produits/1> », et vous obtiendrez un seul enregistrement de produit à partir de la table des produits.

Api Reste pour créer un nouvel enregistrement dans la base de donnée MySQL

Nous allons créer une nouvelle Api Reste pour insérer un nouveau produit dans MySQL en utilisant php. Nous allons créer une requête de type POST car nous publierons des données JSON sur le serveur.

Nous allons ajouter un nouveau cas dans le bloc « Switch » comme indiqué ci-dessous :

```
<?php
...
case 'POST':
// Ajouter un produit
AddProduct();
break;
...
?>
```

Nous allons maintenant créer la méthode **AddProduct()** dans le fichier « produits.php ».

```
<?php
function AddProduct()
{
global $conn;
$name = $_POST["name"];
$description = $_POST["description"];
$price = $_POST["price"];
$category = $_POST["category"];
```

```

$created = date('Y-m-d H:i:s');
$modified = date('Y-m-d H:i:s');
echo $query="INSERT INTO produit(name, description, price, category_id, created,
modified) VALUES('".$name."', '".$description."', '".$price."', '".$category."',
'".$created."', '".$modified."')";
if(mysqli_query($conn, $query))
{
$response=array(
'status' => 1,
'status_message' => 'Produit ajoute avec succes.'
);
}
else
{
$response=array(
'status' => 0,
'status_message' => 'ERREUR!.'. mysqli_error($conn)
);
}
header('Content-Type: application/json');
echo json_encode($response);
}
?>

```

Nous allons maintenant tester notre API Rest pour ajouter un nouveau produit en envoyant une requête POST, pour cela nous allons créer le fichier « post.php » et en ajoutant le code suivant.

```

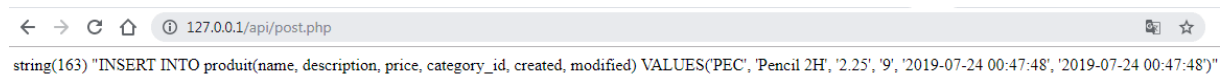
<?php
$url = 'http://127.0.0.1/api/produits';
$data = array('name' => 'PEC', 'description' => 'Pencil 2H', 'price' => '2.25',
'category' => '9');
// utilisez 'http' même si vous envoyez la requête sur https:// ...
$options = array(
'http' => array(
'header' => "Content-type: application/x-www-form-urlencoded\r\n",
'method' => 'POST',

```



```
'content' => http_build_query($data)
)
);
$context = stream_context_create($options);
$result = file_get_contents($url, false, $context);
if ($result === FALSE) { /* Handle error */ }
var_dump($result);
?>
```

Maintenant, accédez à l'url « `http://127.0.0.1/api/post.php` », et vous voyez un nouvel enregistrement de produit est inséré dans la table des produits, et sur le navigateur vous aurez le résultat suivant :



Api Reste pour modifier un enregistrement dans la base de donnée MySQL

Nous allons créer une nouvelle requête HTTP PUT pour mettre à jour les données dans la base de données MySQL.

Nous allons ajouter un nouveau cas dans le bloc « Switch » comme indiqué ci-dessous :

```
<?php
...
case 'PUT':
// Modifier un produit
$id = intval($_GET["id"]);
updateProduct($id);
break;
...
?>
```

Maintenant, nous allons créer la méthode **updateProduct()** dans le fichier « `produits.php` ». Nous allons utiliser l'id du produit que nous voulons mettre à jour et deuxièmement, les données mises à jour au format JSON.

Puisque PHP n'a pas de variable `$_PUT` similaire à `$_GET` et `$_POST` pour récupérer les valeurs transmises, nous utilisons [les flux d'entrée \(input stream](#)

```
'php://input'
```

) pour obtenir ces valeurs pour mettre à jour un produit.

```
<?php
function updateProduct($id)
{
    global $conn;
    $_PUT = array(); //tableau qui va contenir les données reçues
    parse_str(file_get_contents('php://input'), $_PUT);
    $name = $_PUT["name"];
    $description = $_PUT["description"];
    $price = $_PUT["price"];
    $category = $_PUT["category"];
    $modified = date('Y-m-d H:i:s');
    //construire la requête SQL
    $query="UPDATE produit SET name="."$name.", description="."$description.",
    price="."$price.", category_id="."$category.", modified="."$modified." WHERE
    id="."$id";
    if(mysqli_query($conn, $query))
    {
        $response=array(
            'status' => 1,
            'status_message' =>'Produit mis a jour avec succes.'
        );
    }
    else
    {
        $response=array(
            'status' => 0,
            'status_message' =>'Echec de la mise a jour de produit. '. mysqli_error($conn)
        );
    }
    header('Content-Type: application/json');
    echo json_encode($response);
}
?>
```

TUTORIEL CURL EN PHP CURL est un moyen d'accéder à une URL à partir de notre code pour en obtenir une réponse HTML.

```
<?php
$url = "http://127.0.0.1/api/produits/1"; // modifier le produit 1
$data = array('name' => 'MAC', 'description' => 'Ordinateur portable', 'price' =>
'9658', 'category' => '2');
$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "PUT");
curl_setopt($ch, CURLOPT_POSTFIELDS, http_build_query($data));
$response = curl_exec($ch);
var_dump($response);
if (!$response)
{
return false;
}
?>
```

Maintenant, accédez à l'url « <http://127.0.0.1/api/put.php> », et vous voyez que le produit en question a été mis à jour dans la table des produits, et sur le navigateur vous aurez le résultat suivant :



```
← → ↻ 🏠 ⓘ 127.0.0.1/api/put.php
string(63) '{"status":1,"status_message":"Produit mis a jour avec succes."'
```

Api Reste pour supprimer un enregistrement dans la base de donnée MySQL

Nous allons créer une nouvelle requête d'API REST de type DELETE utilisant PHP pour supprimer un produit de la base de données MySQL. Nous transmettons l'id du produit que vous souhaitez supprimer en tant que paramètres.

Nous allons ajouter un nouveau cas dans le bloc « Switch » comme indiqué ci-dessous :

```
<?php
...
case 'DELETE':
// Supprimer un produit
```

```
$id = $_GET["id"];
deleteProduct($id);
break;
...
?>
```

Nous allons maintenant créer la méthode **deleteProduct()** dans le fichier « produits.php ». Nous utiliserons l'id pour supprimer l'enregistrement de la table « produit ».


```
<?php
function deleteProduct($id)
{
    global $conn;
    $query = "DELETE FROM produit WHERE id=".$id;
    if(mysqli_query($conn, $query))
    {
        $response=array(
            'status' => 1,
            'status_message' =>'Produit supprime avec succes.'
        );
    }
    else
    {
        $response=array(
            'status' => 0,
            'status_message' =>'La suppression du produit a echoue. '. mysqli_error($conn)
        );
    }
    header('Content-Type: application/json');
    echo json_encode($response);
}
?>
```

Nous allons maintenant tester notre API Rest pour supprimer le produit en envoyant une requête DELETE via cURL, pour cela nous allons créer le fichier « delete.php » et en ajoutant le code suivant.

```
<?php
```

```
$url = "http://127.0.0.1/api/produits/1"; // supprimer le produit 1
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "DELETE");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$response = curl_exec($ch);
var_dump($response);
curl_close($ch);
?>
```

Maintenant, accédez à l'url « `http://127.0.0.1/api/delete.php` », et vous voyez que le produit en question a été supprimé dans la table des produits, et sur le navigateur vous aurez le résultat suivant :



```
string(61) "{\"status\":1,\"status_message\":\"Produit supprime avec succes.\"}"
```

Conclusion

Il y'a une autre solution, pour tester les méthodes GET/POST/PUT/DELETE que nous avons vu dans ce chapitre en utilisant Postman, qui est un outil open-source pour tester les API RESTful que vous avez créées vous-même. Il offre une interface utilisateur avec laquelle faire des requêtes HTML, sans avoir à écrire un tas de code juste pour tester les fonctionnalités d'une API.

Curl : <https://waytolearnx.com/2020/01/tutoriel-curl-en-php.html>

Liens d api accuweather : <https://developer.accuweather.com/>

Exemple AccuweatherExemple.php