EVALUATION

TP1

LabREST_03_V2

Construire une API REST

- Production
- Compressez votre projet au formation zip (sans le répertoire vendor).
- Déposez le fichier dans Teams.
- Nommage du fichier : Nom_Prenom_apiRestPHP_TP1.zip.

Application : à vous de jouer…

- Objectif : créer des services WEB REST qui permettent de :
  - Créer un produit
  - Afficher les caractéristiques d'un produit à partir de son id
  - Lister l'ensemble des produits
  - Modifier les caractéristiques d'un produit à partir de son id
  - Supprimer un produit à partir de son id

Application : à vous de jouer…

- Objectif 1 : Développer l'application
  - Créer la base de données MySQL nommée db_labrest
  - Créer la table nommée T_PRODUIT
  - Insérer des enregistrements dans la table T_PRODUIT
  - Créer l'entité Produit (Produit.php)
  - Créer la classe de connexion à la base de données (Connexion.php)
  - Créer la classe dao pour l'entité Produit (ProduitDao.php)
  - Créer l'API Rest
- Objectif 2 : Tester l'accès à l'API REST via l'outil Postman
- Objectif 3 : Tester l'accès à l'API REST en PHP via Curl
- Objectif 4 : Documenter l'API REST via la librairie Swagger-PHP
- Objectif 5 : Consommer l'API REST via AJAX (JavaScript) avec l'API Fetch
  - Créer une page d'accueil nommée index.html
  - Créer un fichier Apifetch.js qui contient les différentes fonctions JavaScript

A VOUS DE JOUER...

OBJECTIF 1

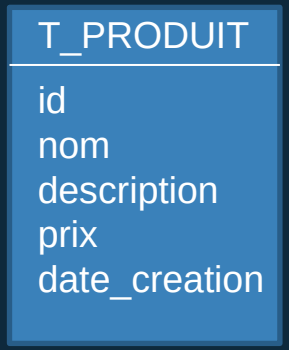
Développer l'application

Application : à vous de jouer…

- 1 table : T_PRODUIT
  id = clé primaire

- 1 entité : modele\entite\Produit.php

- 1 classe dao : modele\dao\ProduitDao.php
  - create ($produit) : crée le produit $produit en BD
  - findAll () : retourne l'ensemble des produits de la BD
  - findById ($id) : retourne le produit correspondant à $id
  - delete ($id) : supprime le produit en BD correspondant à $id
  - update ($produit) : modifie le produit en BD à partir de $produit
- 1 classe pour la connexion à la base de données : modele\dao\Connexion.php




```
class Produit {

    // Attributs
    private $id;
    private $nom;
    private $description;
    private $prix;
    private $date_creation;

    public function __construct(){
    }
}
```

Produit.php

Application : à vous de jouer…

- Api REST : méthodes HTTP utilisées
  - GET : utilisée pour récupérer tous les produits ou un produit à partir de son id
  - POST : utilisée pour créer un produit
  - PUT : utilisée pour mettre à jour un produit
  - DELETE : utilisée pour supprimer un produit

- Api REST : fichiers
  - lire() : retourne tous les produits (méthode GET) : api\lire.php
  - lireUn() : retourne un produit en fonction de son ID (méthode GET) : api\lire_un.php
  - creer() : crée un produit (méthode POST) : api\creer.php
  - modifier() : modifie les attributs d'un produit (méthode PUT) : api\modifier.php
  - supprimer() : supprime un produit à partir de son ID (méthode DELETE) : api\supprimer.php

Application : à vous de jouer…

- Créer un fichier .htaccess à la racine de votre projet : permet la réécriture des URLs
- Fichier de configuration Apache (apache\conf\httpd.conf) : Décommenter la ligne LoadModule rewrite_module modules/mod_rewrite.so
- Permet de personnaliser/simplifier les URLs

```
RewriteEngine on
# Mapping des URLs
RewriteRule ^api/produit/list/?$ api/lire.php
RewriteRule ^api/produit/new/?$ api/creer.php
RewriteRule ^api/produit/update/?$ api/modifier.php
RewriteRule ^api/produit/delete/?$ api/supprimer.php
RewriteRule ^api/produit/delete/([0-9]+)/?$ api/supprimer.php?id=$1
RewriteRule ^api/produit/listone/?$ api/lire_un.php
RewriteRule ^api/produit/listone/([0-9]+)/?$ api/lire_un.php?id=$1

Header set Access-Control-Allow-Origin *
```

.htaccess

http://localhost/LabREST_03/api/produit/lire.php

devient

http://localhost/LabREST_03/api/produit/list

Application : à vous de jouer…

- Avec la version de l'API

.htaccess

```
RewriteEngine on
# Mapping des URLs
RewriteRule ^api/v1.0/produit/list/?$ api/lire.php
RewriteRule ^api/v1.0/produit/new/?$ api/creer.php
RewriteRule ^api/v1.0/produit/update/?$ api/modifier.php
RewriteRule ^api/v1.0/produit/delete/?$ api/supprimer.php
RewriteRule ^api/v1.0/produit/delete/([0-9]+)/?$ api/supprimer.php?id=$1
RewriteRule ^api/v1.0/produit/listone/?$ api/lire_un.php
RewriteRule ^api/v1.0/produit/listone/([0-9]+)/?$ api/lire_un.php?id=$1

Header set Access-Control-Allow-Origin *
```

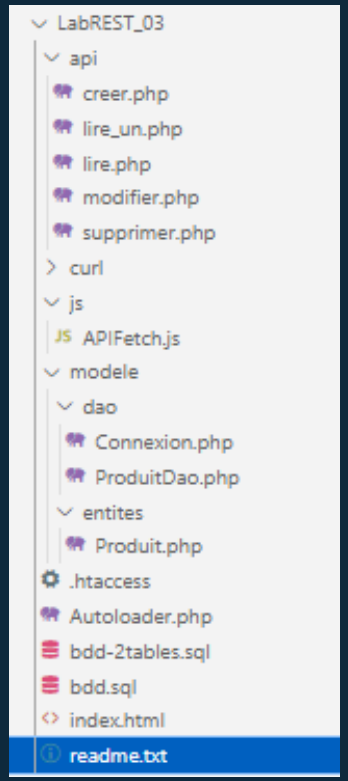
http://localhost/LabREST_03/api/produit/lire.php

devient

http://localhost/LabREST_03/api/v1.0/produit/list

Application : à vous de jouer…

- Organisation de l'application



Back-end : Services WEB REST

Commandes curl pour consommer les services WEB REST

Front-end : Requêtes Ajax pour consommer les services WEB REST

Back-end : Connexion à la base de données

Back-end : Accès à la table T_PRODUIT (CRUD)

Back-end : Entité

Back-end : Réécriture des URLs

Autoloader : Chargement des classes

Front-end : Page d'accueil de l'application

Application : à vous de jouer…

- Méthode **GET** : Obtenir tous les produits

  http://localhost/LabREST_03/api/produit/lire.php

  http://localhost/LabREST_03/api/produit/list/ ← Fichier .htaccess RewriteRule

- Codes de retour

  Status code = 200 : Ok

  Status code = 404 : Not found — Message = Impossible d'obtenir les produits

  Status code = 405 : Method Not Allowed — Message = Méthode non autorisée

  Status code = 503 : Service Unavailable — Message = Impossible de traiter la requête

Application : à vous de jouer…

- Méthode **GET** : Obtenir un produit à partir de son id
- Envoi de l'ID par le corps de la requête

http://localhost/LabREST_03/api/produit/lire_un.php

http://localhost/LabREST_03/api/produit/listone/ ← Fichier .htaccess RewriteRule

```
{
    "id": "67"
}
```

← Donnée au format JSON

- Codes de retour

Status code = 200 : Ok — Message = Produit existe

Status code = 404 : Not found — Message = Produit n'existe pas

Status code = 405 : Method Not Allowed — Message = Méthode non autorisée

Status code = 503 : Service Unavailable — Message = Impossible de traiter la requête

Application : à vous de jouer…

- Méthode **GET** : Obtenir un produit à partir de son id
- Envoi de l'ID par l'URL

  http://localhost/LabREST_03/api/produit/lire_un.php?id=67

  http://localhost/LabREST_03/api/produit/listone/67

Fichier .htaccess
RewriteRule

- Codes de retour

Status code = 200 : Ok	Message = Produit existe
Status code = 404 : Not found	Message = Produit n'existe pas
Status code = 405 : Method Not Allowed	Message = Méthode non autorisée
Status code = 503 : Service Unavailable	Message = Impossible de traiter la requête

Application : à vous de jouer...

- Méthode **POST** : Créer un nouveau produit
- Envoi des données par le corps de la requête

http://localhost/LabREST_03/api/produit/creer.php

http://localhost/LabREST_03/api/produit/new/

Fichier .htaccess
RewriteRule

```
{
   "nom": "Produit1",
   "description": "Ma nouvelle description",
   "prix": "50",
   "date_creation": "2023:04:27 12:12:00"
}
```

Données au format JSON

Note : Pas de champ id

- Codes de retour

Status code = 201 : Created	Message = Produit a été créé
Status code = 503 : Service Unavailable	Message = Impossible de traiter la requête
Status code = 405 : Method Not Allowed	Message = Méthode non autorisée

Application : à vous de jouer...

- Méthode **PUT** : Modifier un produit
- Envoi des données par le corps de la requête

http://localhost/LabREST_03/api/produit/modifier.php

http://localhost/LabREST_03/api/produit/update/

Fichier .htaccess
RewriteRule

```
{
   "id": "67",
   "nom": "Produit1",
   "description": "Ma nouvelle description",
   "prix": "500",
   "date_creation": "2023:04:27 12:12:00"
}
```

Données au format JSON

- Codes de retour

Status code = 200 : Ok ou 204 : Content	Message = Modification effectuée
Status code = 202 : Accepted	Message = Modification non effectuée
Status code = 405 : Method Not Allowed	Message = Méthode non autorisée

Application : à vous de jouer…

- Méthode **DELETE** : Supprimer un produit à partir de son id
- Envoi de l'ID par le corps de la requête

http://localhost/LabREST_03/api/produit/supprimer.php

http://localhost/LabREST_03/api/produit/delete/ ← Fichier .htaccess RewriteRule

```
{
    "id": "67"
}
```

← Données au format JSON

- Codes de retour

Status code = 200 : Ok	Message = Suppression effectuée
Status code = 204 : No content	Message = Produit n'existe pas
Status code = 503 : Service unavailable	Message = Impossible de traiter la requête
Status code = 405 : Method Not Allowed	Message = Méthode non autorisée

Application : à vous de jouer…

- Méthode **DELETE** : Supprimer un produit à partir de son id
- Envoi de l'ID par l'URL

  http://localhost/LabREST_03/api/produit/supprimer.php?id=67

  http://localhost/LabREST_03/api/produit/delete/67 ← Fichier .htaccess RewriteRule

- Codes de retour

Status code = 200 : Ok	Message = Suppression effectuée
Status code = 204 : No content	Message = Produit n'existe pas
Status code = 503 : Service unavailable	Message = Impossible de traiter la requête
Status code = 405 : Method Not Allowed	Message = Méthode non autorisée

Application : Headers

- Accès depuis n'importe quel site ou appareil (*)

  header("Access-Control-Allow-Origin: *");
- Format des données envoyées = JSON

  header("Content-Type: application/json; charset=UTF-8");
- Méthode autorisée = GET (GET, POST, PUT ou DELETE)

  header("Access-Control-Allow-Methods: GET");
- Durée de vie de la requête

  header("Access-Control-Max-Age: 3600");
- Entêtes autorisés

  header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

GET – POST - PUT - DELETE

Application : exemple de service WEB REST

```
// Envoyer les headers requis
// Accès depuis n'importe quel site ou appareil (*)
header("Access-Control-Allow-Origin: *");
…

// Vérifier que la méthode utilisée est correcte
if ($_SERVER['REQUEST_METHOD'] == 'GET') {
        // Instancier un produit
        // Instancier la classe dao
        // Récupérer les produits : appel de la méthode findAll() de ProduitDao
        // Créer un tableau de produits ($tableauProduits)
        // Retourner le status code 200 : Ok
        http_response_code(200);

      // Encoder le tableau en json et envoyer le tableau
      echo json_encode($tableauProduits);
   }
// SINON méthode incorrecte
} else {
   // Retourner le status code 405 : Method Not Allowed
   http_response_code(405);
   // Encoder le message en json et envoyer le message
   echo json_encode(["message" => "Méthode non autorisée"]);
}
```

Fichier lire.php
Méthode HTTP : GET
Retourner tous les produits

$_SERVER est un tableau contenant des informations telles que les en-têtes, les chemins et les emplacements de script.

'REQUEST_METHOD'
Méthode de requête utilisée pour accéder à la page ; par exemple GET, HEAD, POST, PUT.

Application : exemple de service WEB REST

```
// Envoyer les headers requis
// Accès depuis n'importe quel site ou appareil (*)
header("Access-Control-Allow-Origin: *");
…
// Vérifier que la méthode utilisée est correcte
if ($_SERVER['REQUEST_METHOD'] == 'DELETE') {
        // Instancier un produit
        // Instancier la classe dao
        // Récupérer l'id du produit envoyé au format JSON
        $donnees = json_decode(file_get_contents("php://input"));
        // SI l'ID existe
                // Supprimer le produit : appel de la méthode delete() de ProduitDao
                // SI La suppression a fonctionné
                        // Retourner le Status code = 200 : Ok
                        // Encoder le message en json et envoyer le message
                // SINON suppression pas ok
                        // Retourner le Status code = 503 : Service Unavailable
                        // Encoder le message en json et envoyer le message
        // SINON ID n'existe pas
                // Retourner le Status code = 400  : Not found
                // Encoder le message en json et envoyer le message
// SINON méthode incorrecte
        // Retourner le status code 405 : Method Not Allowed
        // Encoder le message en json et envoyer le message
```

Fichier supprimer.php
Méthode HTTP : DELETE
Supprimer un produit à partir de son id

php://input : flux en lecture seule qui vous permet de lire les données brutes du corps de la requête. Il renvoie toutes les données brutes après les en-têtes HTTP de la requête, quel que soit le type de contenu.

Cette fonction retourne le contenu d'un fichier dans une chaîne de caractères.

Cette fonction prend une chaîne JSON et la convertit en une variable PHP qui peut être un tableau ou un objet.

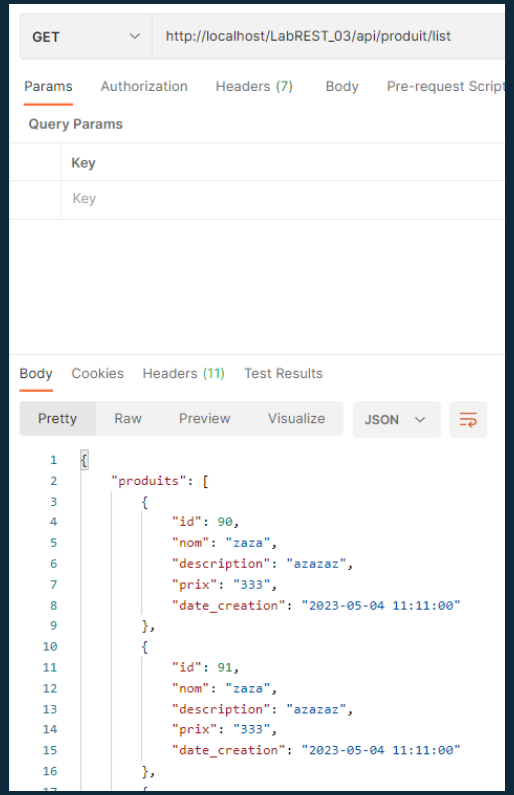
A VOUS DE JOUER…

OBJECTIF 2

Tester l'accès à l'API REST via l'outil Postman

Application : à vous de jouer…

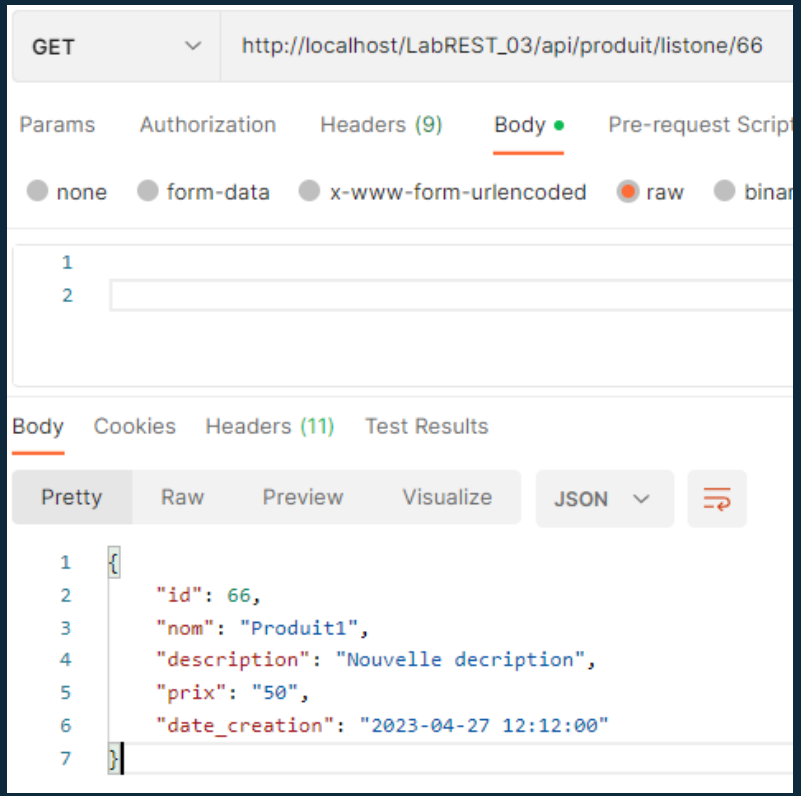
- Tester l'accès à l'API REST via l'outil Postman : méthode GET



Retourne tous les produits au format JSON
Méthode HTTP : GET
URL : http://localhost/LabREST_03/api/produit/list/

Application : à vous de jouer…

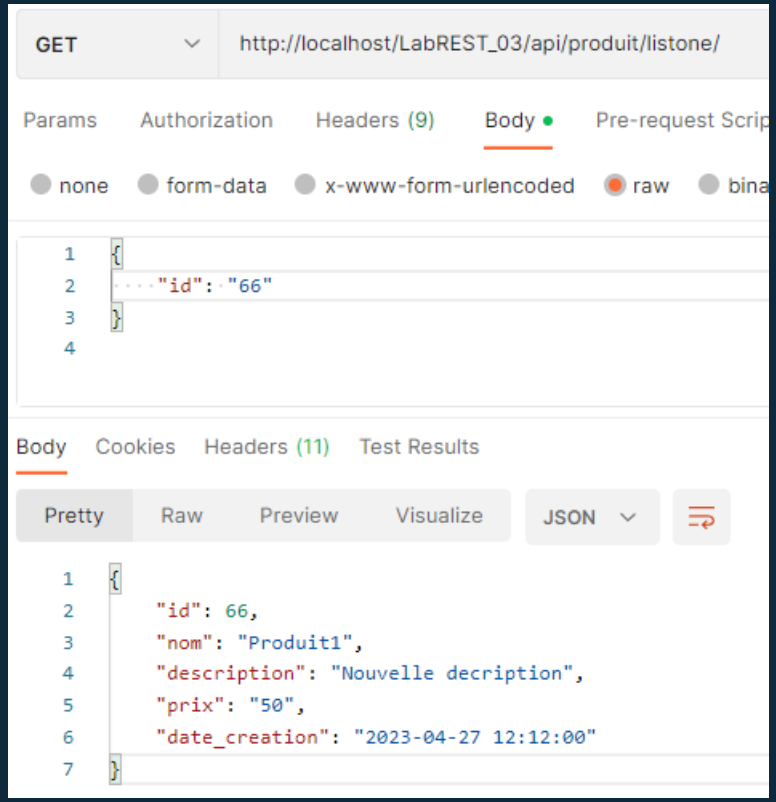
- Tester l'accès à l'API REST via l'outil Postman : méthode GET



Retourne un produit à partir de son id
Id transmis par l'URL
Méthode HTTP : GET
URL : http://localhost/LabREST_03/api/produit/listone/66

Application : à vous de jouer…

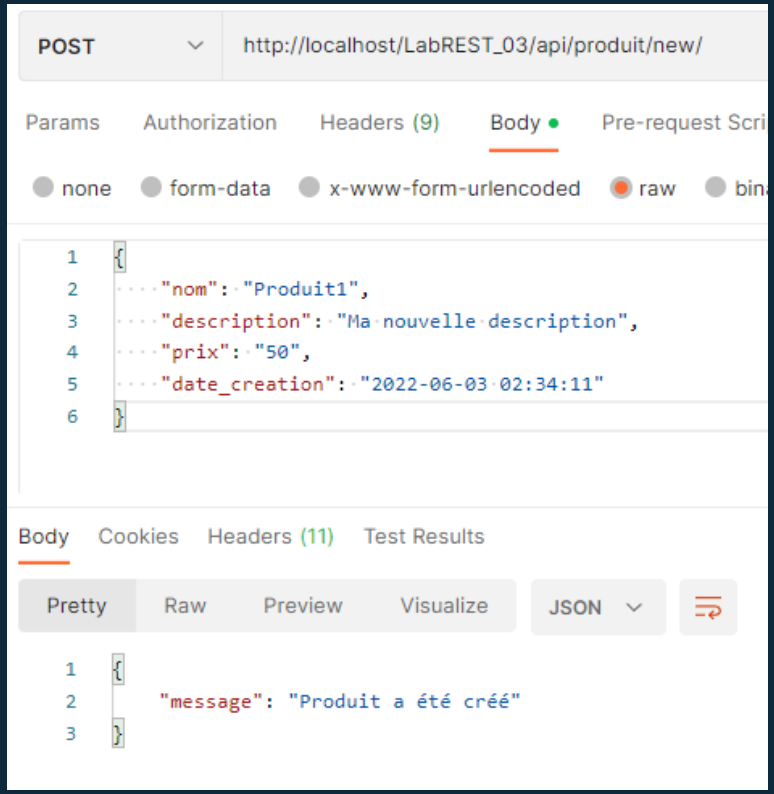
- Tester l'accès à l'API REST via l'outil Postman : méthode GET



Retourne un produit à partir de son id
id transmis dans le corps de la requête
Méthode HTTP : GET
URL : http://localhost/LabREST_03/api/produit/listone/

Application : à vous de jouer…

- Tester l'accès à l'API REST via l'outil Postman : méthode POST



Crée un produit à partir de données au format JSON
Données transmises dans le corps de la requête
Méthode HTTP : POST
URL : http://localhost/LabREST_03/api/produit/new/

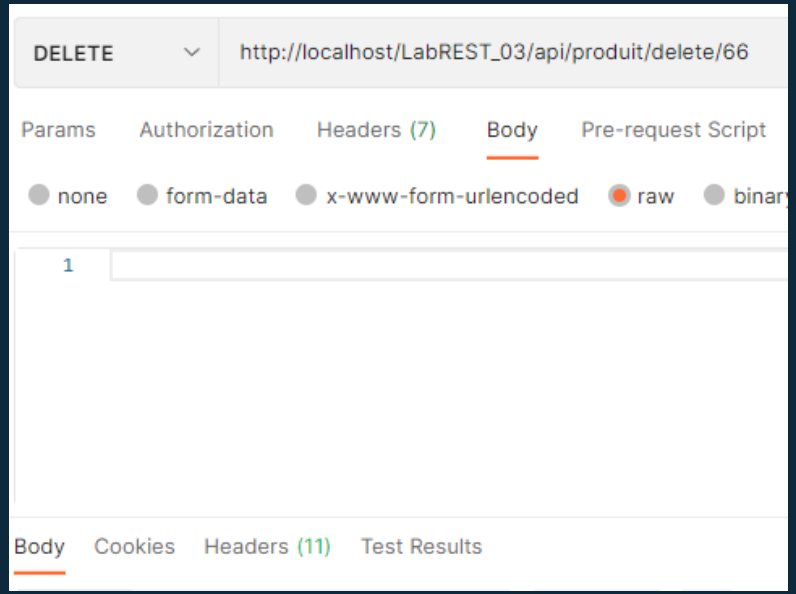
Application : à vous de jouer…

- Tester l'accès à l'API REST via l'outil Postman : méthode DELETE

Supprime un produit à partir de son id
id transmis au format JSON dans le corps de la requête
Méthode HTTP : DELETE
URL : http://localhost/LabREST_03/api/produit/delete/

Application : à vous de jouer…

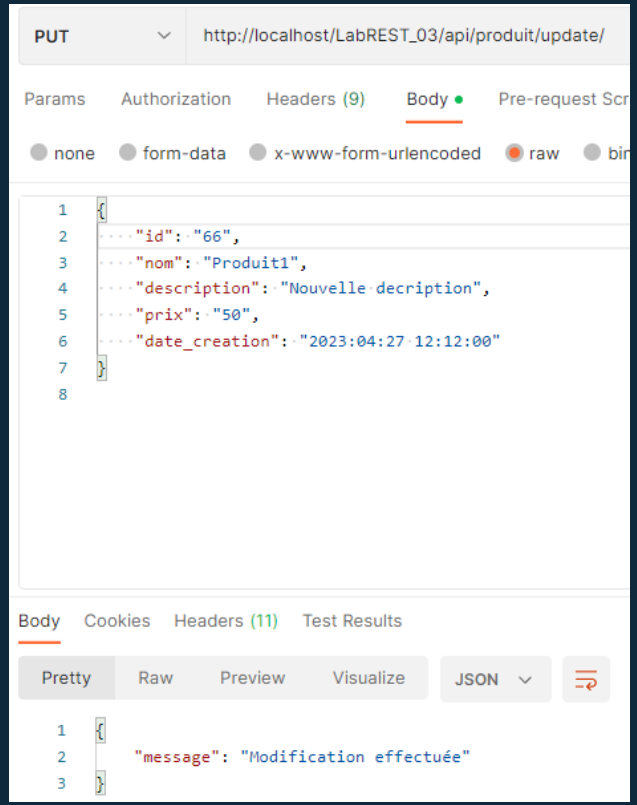
- Tester l'accès à l'API REST via l'outil Postman : méthode DELETE



Supprime un produit à partir de son id
id transmis dans l'URL
Méthode HTTP : DELETE
URL : http://localhost/LabREST_03/api/produit/delete/66

Application : à vous de jouer…

- Tester l'accès à l'API REST via l'outil Postman : méthode PUT



Modifie un produit à partir des données au format JSON
Données transmises au format JSON dans le corps de la requête
Méthode HTTP : PUT
URL : http://localhost/LabREST_03/api/produit/update/

A VOUS DE JOUER...

OBJECTIF 3

Tester l'accès à l'API REST via CURL

Application : à vous de jouer…

- Tester l'accès à l'API REST via PHP
  - curl_init() : Initialiser une session CURL
  - curl_setopt() : Définir les options de transmission CURL
    - CURLOPT_URL
    - CURLOPT_CUSTOMREQUEST
    - CURLOPT_RETURNTRANSFER
    - CURLOPT_POSTFIELDS
    - CURLOPT_HTTPHEADER
  - curl_exec() : Exécuter une session CURL
  - curl_close() : Fermer une session CURL

https://www.php.net/manual/fr/ref.curl.php

Application : à vous de jouer…

Fichier curl\delete.php

```
<?php
// Définir l'URL
$url = "http://127.0.0.1/LabREST_03/api/produit/delete/";

// Donnée à transmettre au format JSON : id du produit à supprimer
$data = json_encode(array('id' => '78'));

// Initialiser une session CURL
$ch = curl_init();

// Définir les options de transmission CURL : url, méthode, données, …
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "DELETE");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_POSTFIELDS,$data);

// Exécuter la session CURL et récupérer la réponse
$response  = curl_exec($ch);

// Afficher la réponse du service WEB REST
var_dump($response);

// Fermer la session CURL
curl_close($ch);
?>
```

Méthode HTTP : DELETE
Supprimer un produit à partir de son id

Exécution via le terminal
php delete.php

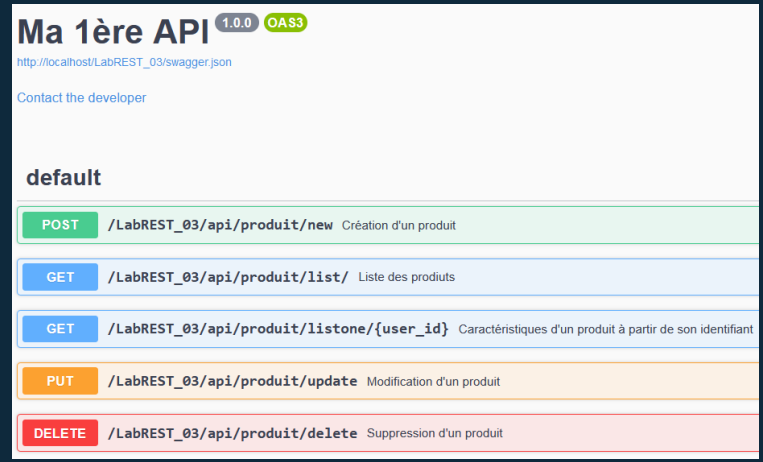
A VOUS DE JOUER...

OBJECTIF 4

Documenter l'API REST via la librairie Swagger-PHP

Documentation de l'API

- Installer la librairie Swagger-PHP avec Composer
- Ajouter des annotations à votre API
  - api\lire.php
  - api\lire_un.php
  - api\creer.php
  - api\modifier.php
  - api\supprimer.php
- Télécharger Swagger-UI via git
- Copier le répertoire dist dans votre projet
- Configurer Swagger-UI
- Accéder à la documentation de l'API



A VOUS DE JOUER...

OBJECTIF 5

Consommer les services WEB REST via AJAX

Application : consommer l'Api...

- Créer une page HTML qui permet de consommer les services WEB via des requêtes AJAX.

Application : consommer l'Api...

- Créer un fichier Apifetch.js
- Ajouter les listeners sur les boutons
- Créer les différentes fonctions JavaScript :
  - getProduits() : récupère tous les produits
  - getProduit() : récupère un produit en fonction de son id
  - deleteProduit() : supprime un produit en fonction de son id
  - createProduit() : crée un produit en fonction des données saisies
  - updateProduit() : modifie un produit en fonction de son id et des données saisies
  - displayProduits(json) : alimente la combo box avec les produits reçus
  - displayProduit(json) : affiche les caractéristiques du produit reçu
  - displayMessage(json) : affiche le message retourné