#### Exercici 1: Creació i Consum d'una API REST amb PHP

#### Context:

En aquest exercici, treballareu en la creació d'una API REST que gestionarà una llista de productes d'una botiga online. Aquesta API permetrà afegir, consultar i esborrar productes utilitzant els mètodes HTTP POST, GET i DELETE. A més, desenvolupareu un petit script PHP que consumirà aquesta API per mostrar la llista de productes i permetre l'eliminació o afegit de nous productes.

# Objectius:

- Crear una API REST bàsica en PHP que gestioni productes.
- Implementar els mètodes GET, POST i DELETE.
- Consumir aquesta API mitjançant cURL des d'un altre script PHP.

### Requisits del Cas Pràctic:

- 1. Creació de l'API (api.php):
  - o **GET:** Retorna la llista de productes en format JSON.
  - POST: Afegeix un nou producte (amb "name" i "price") a la "base de dades" (un fitxer JSON).
  - o **DELETE:** Elimina un producte pel seu "id".
- 2. **Base de dades simulada:** Utilitzeu un fitxer JSON (**products.json**) per emmagatzemar els productes. Exemple inicial:

### 3. Consumir l'API (client.php):

- Llistar els productes utilitzant una petició GET.
- Afegir un nou producte mitjançant una petició POST.
- Permetre eliminar un producte pel seu id utilitzant DELETE.

### Resultat esperat:

- Heu de mostrar com es fa la petició i com es processa la resposta.
- La pàgina hauria de mostrar la llista de productes i proporcionar opcions per afegir o eliminar productes.

### Exercici 2: Generació Dinàmica de Contingut amb Twig

#### Context:

En aquest exercici, creareu una petita aplicació web per gestionar una llista de llibres utilitzant Twig com a motor de plantilles. Aquesta aplicació permetrà visualitzar la llista de llibres i afegir nous llibres mitjançant formularis.

### Objectius:

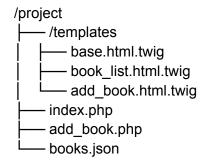
- Comprendre i aplicar el motor de plantilles Twig.
- Separar la lògica del servidor de la presentació amb plantilles netes i organitzades.
- Generar contingut dinàmic basat en dades introduïdes per l'usuari.

## Requisits del Cas Pràctic:

## 1. Configuració de Twig:

 Instal·leu Twig utilitzant Composer: composer require "twig/twig:^3.0"

#### 2. Estructura del projecte:



#### 3. Funcionalitats:

- index.php: Llegeix els llibres de books.json i els mostra a book\_list.html.twig.
- add\_book.php: Formulari per afegir nous llibres (títol, autor, any) utilitzant la plantilla add\_book.html.twig. Quan l'usuari envia el formulari, el llibre es desa a books.json i es redirigeix a la p\u00e0gina principal.

### Exemple de plantilla (book\_list.html.twig):

```
{% extends 'base.html.twig' %}

{% block title %}Llista de Llibres{% endblock %}

{% block content %}
<h1>Llibres Disponibles</h1>

{% for book in books %}
<strong>{{ book.title }}</strong> de {{ book.author }} ({{ book.year }})
{% else %}
```

```
No hi ha llibres disponibles.
{% endfor %}

<a href="add_book.php">Afegir un nou llibre</a>
{% endblock %}
```

# Resultat esperat:

- Heu d'entendre com funciona la separació de la lògica de la presentació.
- L'aplicació ha de ser clara, amb plantilles ben estructurades i contingut dinàmic adaptat a les dades introduïdes.