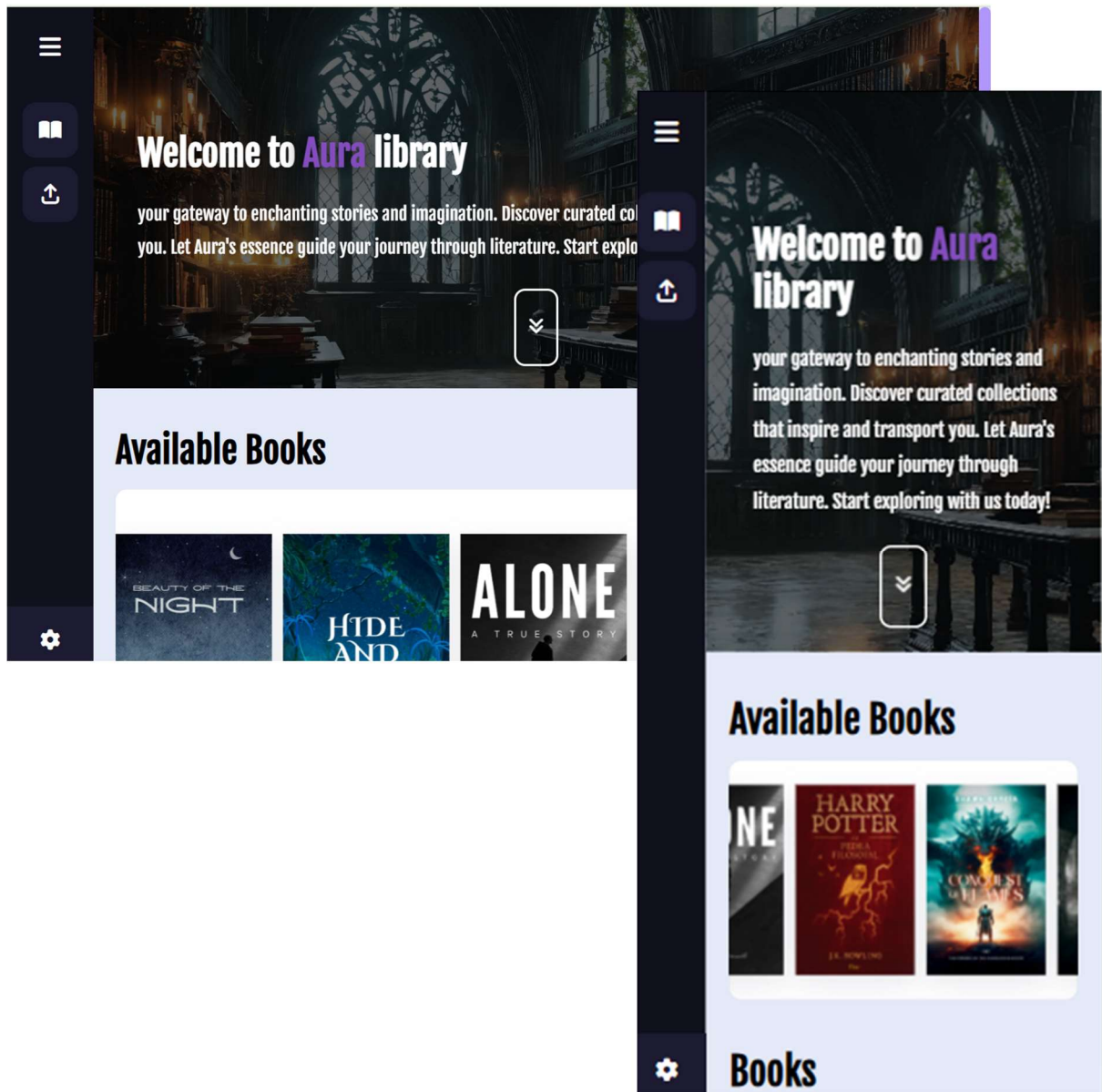


Projet : Gestion de Bibliothèque



Réalisé par : *imad rouah G2*

Date : 21/05/2024

Encadré par : *Pr Z. Lamghari.*

Introduction

Ce projet consiste en la création d'un système de gestion de bibliothèque utilisant HTML, CSS, JavaScript (jQuery), PHP et MySQL. Il permet aux utilisateurs d'ajouter, mettre à jour, supprimer et rechercher des livres, tout en offrant une interface utilisateur intuitive et des fonctionnalités de recherche en temps réel.

Structure des Pages HTML

1. Page Principale (index.html)

La page principale de l'application contient les sections suivantes :

- **Sidebar de Navigation** : Permet de naviguer entre les sections de la page (Livres, Téléchargement, Paramètres).
- **Section d'Introduction** : Accueil avec une image et une description de la bibliothèque.
- **Section des Livres** : Affiche la liste des livres disponibles avec des options de recherche et de tri.
- **Section de Téléchargement** : Formulaire permettant l'ajout de nouveaux livres.
- **Footer** : Contient des informations de copyright.

2. Page de Paramètres (setting.html)

La page de paramètres permet d'afficher des statistiques sur la bibliothèque et de gérer les livres enregistrés :

- **Navigation** : Lien de retour à la page principale et liens vers les statistiques et la liste des livres.
- **Section des Statistiques** : Affiche le nombre total d'auteurs et de livres, ainsi qu'une liste des auteurs avec leurs statistiques.
- **Section de Gestion des Livres** : Affiche la liste des livres avec des options de mise à jour et de suppression.

Fonctionnalités Principales

1. Ajouter un Livre

Formulaire d'ajout :

- **Champs** : Titre, Auteur, Année de publication, Image de couverture (facultative).
- **Validation** : Vérifie que tous les champs obligatoires sont remplis.
- **Ajout via AJAX** : Utilise une requête AJAX pour envoyer les données au serveur sans recharger la page.

2. Afficher les Livres

Affichage dynamique :

- **Recherche en temps réel** : Filtre les résultats de la recherche en temps réel à mesure que l'utilisateur tape.
- **Tri des livres** : Permet de trier les livres par titre, auteur ou année de publication.
- **Affichage des résultats** : Les livres sont affichés sous forme de cartes avec leur image de couverture, titre, auteur et année.

3. Gestion des Livres

Mise à jour et suppression :

- **Mise à jour** : Permet de modifier les informations d'un livre existant.
- **Suppression** : Permet de supprimer un livre de la bibliothèque.
- **Suppression de tous les livres** : Option pour supprimer tous les livres enregistrés.

4. Statistiques

Affichage des statistiques :

- **Nombre total d'auteurs et de livres.**
- **Détails par auteur** : Nombre de livres, année de publication la plus ancienne et la plus récente.

Tests avec Postman1. Obtenir la Liste des Livres

1. Test de la récupération des livres (requête GET)

Configuration de la requête GET

- Sélectionnez la méthode GET.
- Saisissez l'URL
`http://localhost/projet_dev/php/getBooks.php.`

HTTP

http://localhost/projet_dev/php/getBooks.php

Save

GET

http://localhost/projet_dev/php/getBooks.php

Send

ParamsAuthHeaders (6)BodyScriptsTestsSettingsCookies

PrettyRawPreviewVisualizeJSON

```
1 [
2   {
3     "id": "1",
4     "title": "the night",
5     "author": "Auth1",
6     "year": "1970-01-01",
7     "pictype": "jpg"
8   },
9   {
10    "id": "2",
11    "title": "Hide and seek",
12    "author": "Auth5",
13    "year": "2001-09-12",
14    "pictype": "jpg"
15  },
16  {
17    "id": "3",
18    "title": "Alone",
19    "author": "Auth3",
20    "year": "2005-03-13",
21    "pictype": "jpg"
22  },
23  {
24    "id": "4",
25    "title": "Harry",
```

- Pas besoin de formulaire données
- Cliquez sur **Send**.
- La réponse devrait être une liste **JSON** des livres présents dans la base de données.
- **Récupération de livres** : Le script **getBooks.php** retourne tous les livres sous forme de tableau JSON. Si une colonne de tri est spécifiée via le paramètre **sort**, les livres sont triés en conséquence.

2. Test de l'ajout d'un livre (requête POST)

Ouverture de Postman et configuration de la requête POST

- Sélectionnez la méthode **POST**.
- Saisissez l'URL
http://localhost/projet_dev/php/addBook.php.

Ajout des données du formulaire

- Allez dans l'onglet **Body**.
- Sélectionnez **form-data**.
- Ajoutez les champs suivants:
 - **title** : Le titre du livre (par exemple, **New Book**).
 - **author** : L'auteur du livre (par exemple, **Author Name**).
 - **year** : L'année de publication (par exemple, **2024-05-21**).
- Cliquez sur **Send**.
- La réponse devrait être **2** si l'ajout est réussi



Params Authorization Headers (8) **Body** ● Scripts Tests Settings

☐ none
☒ form-data
☐ x-www-form-urlencoded
☐ raw
☐ binary
☐ GraphQL

| | Key | | Value |
|-------------------------------------|--------|--------|------------|
| <input checked="" type="checkbox"/> | title | Text ▾ | new book |
| <input checked="" type="checkbox"/> | author | Text ▾ | author |
| <input checked="" type="checkbox"/> | year | Text ▾ | 2020-12-12 |
| | Key | Text ▾ | Value |

Body Cookies Headers (7)

Pretty Raw Preview

```

1  2

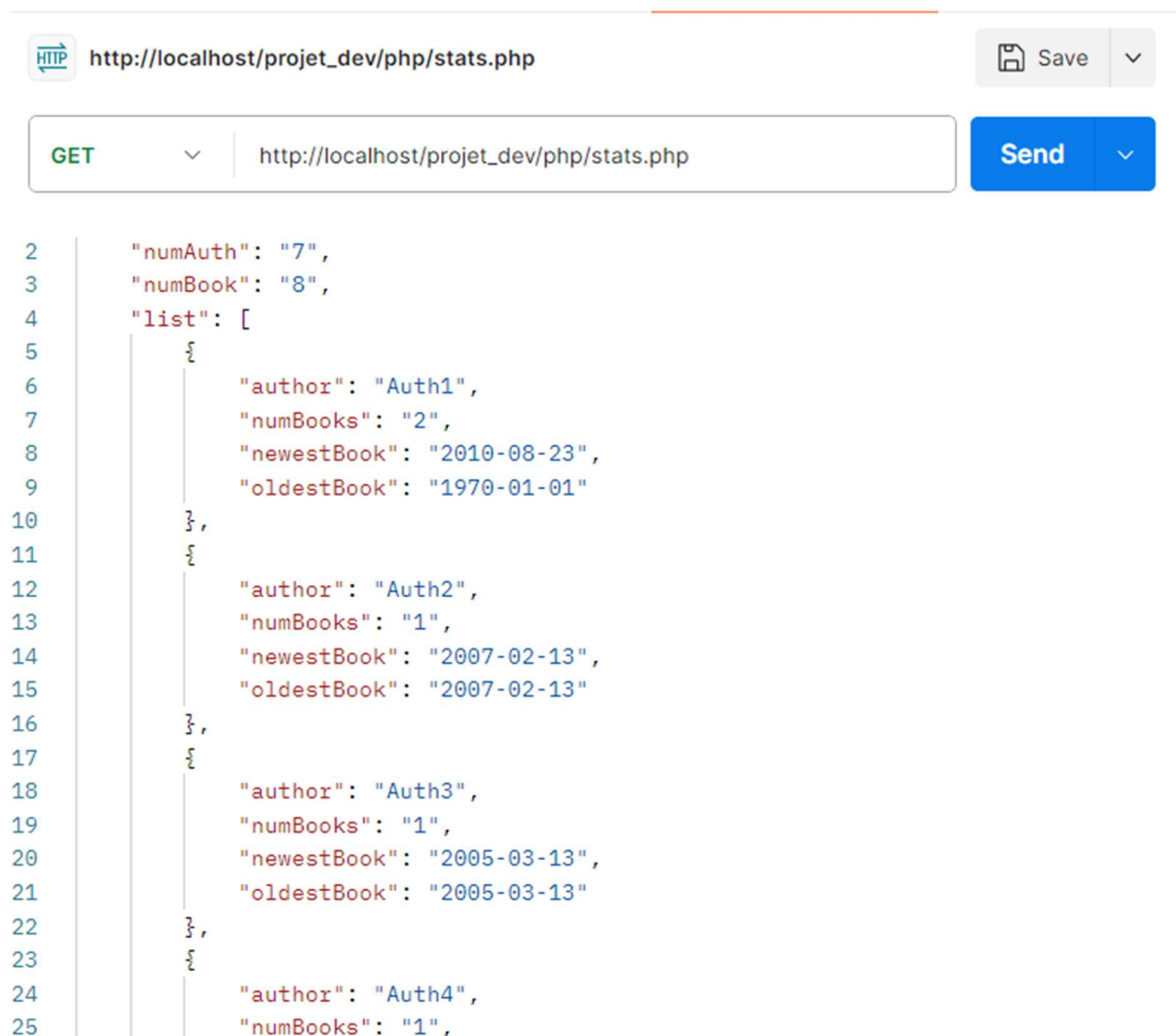
```

- **Ajout de livres** : Lors de l'ajout d'un livre avec une image, le script **addBook.php** vérifie si le livre existe déjà. Si ce n'est pas le cas, il insère le nouveau livre dans la base de données et gère le téléchargement de l'image. En cas de succès, la réponse est **2**. Il y a 3 résultats possible (1, 2 et 3)
 - 1- Ça veut dire que le livre existe déjà
 - 2- Signifie que le livre a été enregistré avec succès
 - 3- Signifie qu'il y a un problème avec l'image téléchargée

3. Test de la récupération des statistiques (requête GET)

Configuration de la requête GET pour stats.php

- Sélectionnez la méthode **GET**.
- Saisissez l'URL
http://localhost/projet_dev/php/stats.php.
- Cliquez sur **Send**.
- La réponse devrait être un objet JSON contenant des statistiques sur les livres et les auteurs.



The screenshot shows a web browser interface for testing HTTP requests. The address bar displays the URL `http://localhost/projet_dev/php/stats.php`. Below the address bar, the method is set to `GET` and the same URL is entered in the request field. A blue `Send` button is visible. The response area shows a JSON object with the following structure:

```
2  "numAuth": "7",
3  "numBook": "8",
4  "list": [
5    {
6      "author": "Auth1",
7      "numBooks": "2",
8      "newestBook": "2010-08-23",
9      "oldestBook": "1970-01-01"
10   },
11   {
12     "author": "Auth2",
13     "numBooks": "1",
14     "newestBook": "2007-02-13",
15     "oldestBook": "2007-02-13"
16   },
17   {
18     "author": "Auth3",
19     "numBooks": "1",
20     "newestBook": "2005-03-13",
21     "oldestBook": "2005-03-13"
22   },
23   {
24     "author": "Auth4",
25     "numBooks": "1",
```


Analyse de la réponse

- La réponse JSON devrait contenir :
 - **numAuth** : Le nombre total d'auteurs distincts.
 - **numBook** : Le nombre total de livres.
 - **list** : Une liste des auteurs avec le nombre de livres, la date de publication la plus récente et la plus ancienne.

4. Test de la recherche en direct (requête POST)

Configuration de la requête POST pour livesearch.php

- Sélectionnez la méthode POST.
- Saisissez l'URL
`http://localhost/projet_dev/php/livesearch.php`.

Ajout des données du formulaire

- Allez dans l'onglet Body.
- Sélectionnez form-data.
- Ajoutez les champs suivants:
 - **live** : Mettez n'importe quoi, pour indiquer que c'est une recherche en direct.
 - **searchData** : Les données de recherche (par exemple, Harry).

HTTP http://localhost/projet_dev/php/livesearch.php Save

POST http://localhost/projet_dev/php/livesearch.php Send

Params Auth Headers (8) **Body** Scripts Tests Settings Cookies

form-data

| | Key | Value | Description | ... | Bulk Edit |
|-------------------------------------|------------|-------|-------------|-----|-----------|
| <input checked="" type="checkbox"/> | live | Text | test | | |
| <input checked="" type="checkbox"/> | searchData | Text | harry | | |

pretty Raw Preview Visualize JSON

```

1  [
2    {
3      "title": "Harry",
4      "author": "Auth4"
5    }
6  ]

```

5. Test de la recherche avec tri (requête POST)

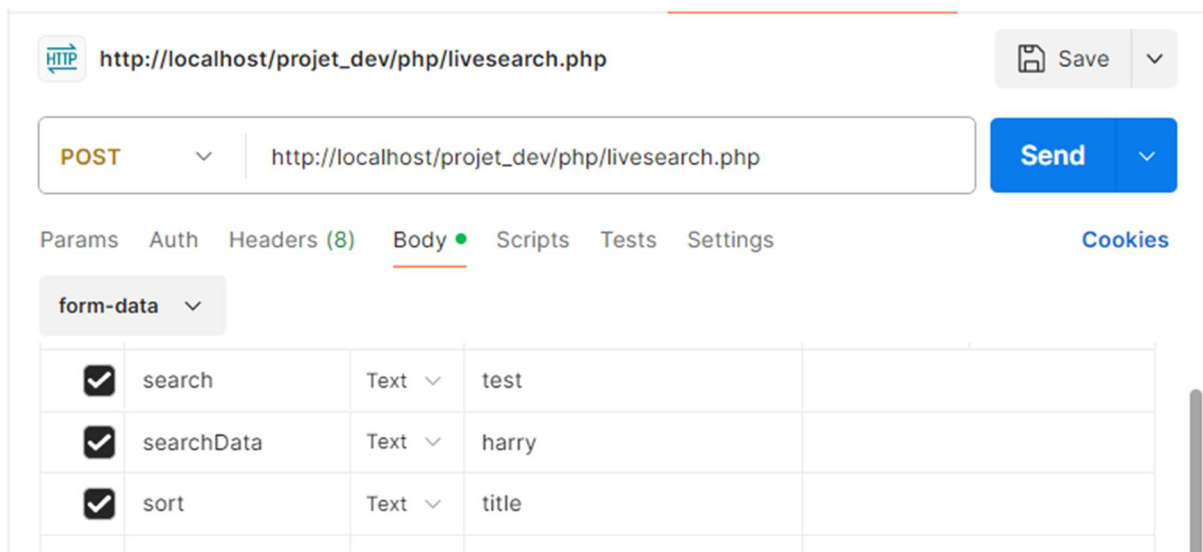
Configuration de la requête POST pour la recherche avec tri

- Sélectionnez la méthode **POST**.
- Saisissez l'URL
http://localhost/projet_dev/php/livesearch.php.

Ajout des données du formulaire

- Allez dans l'onglet **Body**.
- Sélectionnez **form-data**.
- Ajoutez les champs suivants:

- **search** : Mettez n'importe quoi, pour indiquer que c'est une recherche avec tri.
- **searchData** : Les données de recherche (par exemple, **Harry**).
- **sort** : Le champ par lequel trier les résultats (par exemple, **title**).



The screenshot shows a web client interface with the following details:

- URL:** `http://localhost/projet_dev/php/livesearch.php`
- Method:** `POST`
- Body Type:** `form-data`
- Body Fields:**

| Field Name | Type | Value |
|--|------|-------|
| <input checked="" type="checkbox"/> search | Text | test |
| <input checked="" type="checkbox"/> searchData | Text | harry |
| <input checked="" type="checkbox"/> sort | Text | title |

Envoi de la requête et analyse de la réponse

- Cliquez sur **Send**.
- La réponse devrait être une liste JSON des livres correspondants à la recherche, triés par le champ spécifié.

| retty | Raw | Preview | Visualize | JSON |
|-------|-----|-----------------------|-----------|------|
| 1 | [| | | |
| 2 | { | | | |
| 3 | | "id": "4", | | |
| 4 | | "title": "Harry", | | |
| 5 | | "author": "Auth4", | | |
| 6 | | "year": "2019-03-02", | | |
| 7 | | "pictype": "jpg" | | |
| 8 | } | | | |
| 9 |] | | | |