

Ludothèque Application Web

Sommaire

<hr/>	
Introduction	
- Description du projet:	2
→ Consignes:	2
→ Objectif personnel :	2
- Présentation:	2
<hr/>	
Les bases	
- les étapes de création:	3
- HTML :	4
- Formulaire:	5
<hr/>	
Méthodes utilisées	
- GET:	6
- POST :	7
<hr/>	
Explications	
- Searchbar :	8

Introduction

Description du projet

Consignes :

Créer une application web Ludothèque qui rassemble les compétences acquises en HTML, CSS, et PHP. On doit pouvoir au minimum ajouter et supprimer des jeux et avoir un aperçu sur la page d'accueil des jeux dont on dispose dans sa ludothèque.

Objectif personnel :

Réussir le projet et ajouter les fonctionnalités suivantes :

- Ajouter une fonctionnalité de modification d'un ou plusieurs champs d'un jeu déjà ajouté
- ajouter une fonction de recherche qui trouve les jeux associés contenant les lettres tapées dans les champs
- ajouter une limite de 10 jeux max affichés sur la page d'accueil avec un bouton qui permet la possibilité d'étendre le tableau
- ajouter une page de connexion et la possibilité de récupérer ses jeux en se connectant à son compte.

Présentation



page insertion

nom du jeu* :

choix plateforme :

age minimum :

prix d'achat* :

date d'achat :

editeur :

description* :

importer une photo :

[retour au menu](#)

localhost/ludotheque/accueil.php

Bienvenue florent
[deconnection](#)

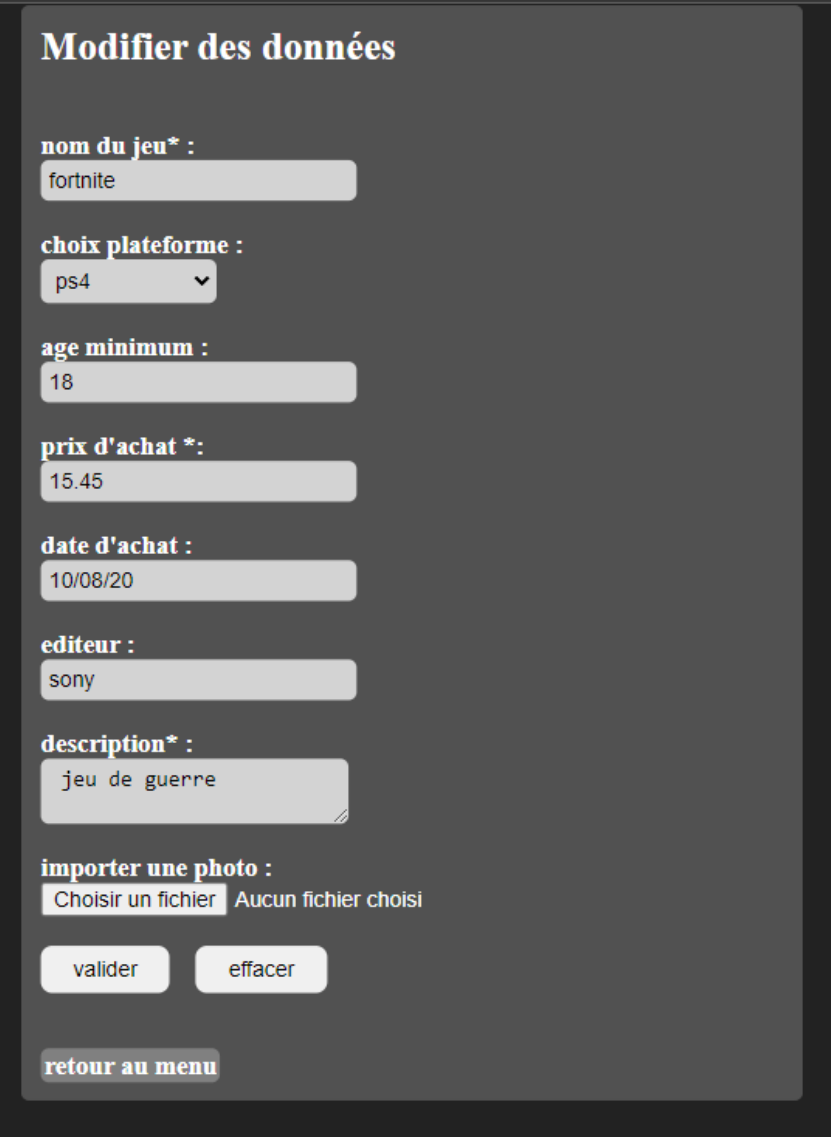
ludothèque de florent

Search...

[inserer des données](#)

Mes jeux

nom du jeu	age requis	prix d'achat	date d'achat	description	photo	plateforme	editeur	modifier	corbeille
fortnite	18	15.45	10/08/20	jeu de guerre	photo	ps4	sony	modifier	



Modifier des données

nom du jeu* :
fortnite

choix plateforme :
ps4 ▼

age minimum :
18

prix d'achat* :
15.45

date d'achat :
10/08/20

editeur :
sony

description* :
jeu de guerre

importer une photo :
Choisir un fichier Aucun fichier choisi

valider effacer

retour au menu

Les bases

projet

Pour créer un site dynamique il faut d'abord commencer par le html qui est le squelette/ l'ossature du site. ensuite on vient y ajouter le php et si on veut récupérer et stocker ces données on crée en parallèle une base de donnée

les étapes:

- penser et créer sa base de donnée
- se faire une idée de l'organisation de son site et des pages (la mise en page propre on le fait à la fin pour faire joli avec le CSS)
- faire son ossature en html
- ajouter du css au code html

Penser sa base de donnée :

faire un listing des informations que l'on souhaite récupérer auprès de l'utilisateur, ex nom, prénom etc..

une fois ce listing fait on peut dégager les informations redondantes, c'est à dire qui vont revenir plusieurs fois, on crée alors des tables en fonction de ces informations

exemple concret pour une ludothèque:

je veux nom du jeu, prix, dates d'achat, description, plateforme, éditeur ... etc

On remarque que l'éditeur, la plateforme et le nom du jeu par exemple sont des info redondantes, en effet on peut trouver plusieurs jeux sur PS4.

//note: on aurait pu faire une table age pour l'âge minimum

de ce fait on créera donc 3 tables en suivant cet exemple
jeu, plateforme et éditeur

maintenant qu'on a les idées un peu plus claires on peut organiser nos tables avec les différents champs et ensuite créer le script SQL

cette étape permet aussi de rajouter les derniers champs auxquels on aurait pas pensé comme un champ ID pour nommer de manière unique les lignes

un champs ID par table minimum pour être pro et pouvoir faire nos jointures tranquillou qui dit jointures dit clés primaires et étrangères, c'est après avoir organisé nos tables qu'on peut décider clairement quel champs on va mettre en clé primaire même si en général on met le champ ID de chaque table.

exemple :

SCHEMA TABLE

une fois ce schéma fait au propre, on peut commencer à s'amuser un peu.

il va falloir maintenant faire le script SQL sur un éditeur de texte comme notepad ou alors sur phpMyAdmin si t'as la flemme de te casser la tête avec le SQL

// note il faut quand même réfléchir au type des champs (INTEGER , VARCHAR , etc)et à leur taille pour avoir une base propre

pour le script SQL on peut s'aider des scripts du premier semestre script1.sql le 2 et le 3 en respectant la méthodologie

1- création table avec champs, type, taille

2- entrer des valeurs dans les tables au moins une ligne

3- on ajoute les clés étrangères.

II) ossature du site en HTML

Comme pour la base de donnée on réfléchit à l'idée et à ce qu'on veut faire avant de se jeter sur internet pour avoir un beau site.

questions à se poser:

qu'est ce que je mets sur ma page d'accueil ? Est ce que je fais plusieurs pages, si oui j'y mets quoi?

Étant donné qu'on veut ajouter une interaction avec l'utilisateur par la suite il faut penser aux formulaires, sur quelle page je le ou les mets?

Une fois cette réflexion posée, pas besoin d'avoir une idée hyper sophistiquée, on passe au code.

le html s'écrit dans des balises <HTML></HTML> qui comporte une entête, un corp et un pied de page.

on va principalement travailler dans le corps donc le body <body></body>

donc pour le moment on a ça :

```
<html>
  <body>
    notre corps page
  </body>
</html>
```

quelques balises pratiques

</br> = retour à la ligne

<p></p> = balise pour écrire du texte

 = le texte qui se trouve entre ces balise sera écrit en gras

<h1></h1> = les titre (en vrai c'est pour le référencement mais c'est bien pratique et vu qu'on met pas en ligne on s'en fou) ça va de h1 à h6 et h1 c'est le plus grand

<button></button> = ça crée un bouton

créer un lien vers une autre page ou un site internet:

nom qui apparaît à l'écran

et enfin le formulaire

on place dans les balise form

<form></form>

action="" ⇒ on met le nom de la page ou on veut envoyer les données du formulaire, la ou se trouvera le php

method="" ⇒ on utilisera la méthode POST comme ça pas d'embrouille toute les données sont en arrière plan

il existe plusieurs type de <input type="" />

voir ce site : https://www.w3schools.com/html/html_form_input_types.asp

on utilisera type="text" et type="submit" qui servent respectivement à entrer du texte et à valider le formulaire donc envoyer les données sur la page spécifiée par action=""

exemple de formulaire :

ce qui donne ça

<form action="traitement.php" method="POST">

nom du jeu : <input type="text" name="nomJeu" /> nom du jeu :

<input type="submit" value="valider" />

</form>



maintenant à toi de faire ta page et ton ou tes formulaires

III) Le PHP

une fois l'ossature créée, encore une fois pas besoin que ce soit joli il faut avoir les information que l'on souhaite à l'écran rien de plus même si c'est pas très beau pour le moment.

on va maintenant récupérer les informations de notre formulaire donc la le nom de notre jeu

le php s'écrit lui dans des balises `<?phpnotre code... ?>`

on récupère les valeurs du formulaire avec le "name" dans le formulaire qu'on vient de faire le name est "nomjeu"

on va venir récupérer avec la variable superglobale `$_POST` et affecté à une variable qu'on appelle comme on veut

//note on ne fait pas d'espace dans les noms de variables on fait de tirets à la place et on met un nom qui permet de s'y retrouver

on écrit donc

```
<?php
$nomJeu = $_POST['nomJeu'];
?>
```

donc maintenant si dans mon formulaire j'ai écrit "clash royal" ma variable `$nomJeu` vaut "cash royal"

Méthodes utilisées



Les variables POST et GET sont des variables superglobales tout comme \$_SESSION, \$_COOKIE ou encore \$_FILES et servent au passage de valeurs à travers les pages.

GET

La méthode GET est utilisée dans le cas de passage de valeur par l'URL, des valeurs propres au site qui ne compromettent pas sa sécurité. En effet les valeurs passées dans l'URL sont facilement identifiables car affichées en clair.

Avantages	Inconvénient
<ul style="list-style-type: none">- Plus efficace que la méthode post- peut être mise en cache	<ul style="list-style-type: none">- quantité limitée de données à passer- la requête n'est pas sécurisée- paramètres restent dans l'historique du navigateur car ils sont dans l'URL- seul les caractères ASCII sont autorisés

Syntaxe :

```
ma_Var = $_GET['variable'];
```

POST

La méthode POST répond au défaut de la méthode GET qui est l'affichage en clair de valeurs de variables dans l'URL, ainsi POST est utilisé pour récupérer des données sensibles comme le mot de passe ou encore des données à caractères personnelles.

Avantages	Inconvénient
<ul style="list-style-type: none">- Sécurisée- grande quantité de donnée possible- les données sont pas affichées en clair- les paramètres ne sont pas enregistrés dans l'historique- variables pas visibles dans l'URL	<ul style="list-style-type: none">- moins efficace que la méthode GET

Syntaxe pour récupérer une valeur passée avec la méthode post:

```
ma_Var = $_POST['variable'];
```


Explications

Searchbar

```

if(isset($_POST['rechercher'])&&(!empty($_POST['search']))) {
    //affichageRecherche($bdd);
    try{
        $search = "%".$_POST['search']."%";

        $sql = $bdd->prepare(" SELECT idJeu, jeu.nomJeu, jeu.ageMini, jeu.prixAchat, jeu.dateAchat, description, picture, plateforme.plateforme, editeur
        FROM JEU, PLATEFORME, EDETEUR
        WHERE jeu.idplateforme = plateforme.idplateforme
        AND jeu.idEditeur = editeur.idEditeur
        AND nomJeu LIKE ? ");
        $sql->execute([$search]);

        affichage($sql);
    }catch(Exception $e){
        Die ('Erreur :'. $e->getMessage());
    };
}

```

```
if(isset($_POST['rechercher'])&&(!empty($_POST['search'])))
```

Pour valider une recherche il faut que le champ soit rempli d'au moins 1 caractère et qu'on ai cliqué sur le bouton rechercher. Sans ces 2 conditions le script n'est pas exécuté. On utilise le && pour l'opérateur logique ET

```

try{ action }catch(Exception $e){
    Die ('Erreur :'. $e->getMessage());
};

```

On utilise le try catch pour entourer des instructions qui vont effectuer une action sur la base de données. En l'occurrence ici on recherche des des valeurs dans le champ nomJeu de la base de données.

catch sert à attraper l'erreur si la connexion ou une erreur survient lors de l'échange avec la base de données. Cette erreur est ensuite affichée avec un message plus user friendly qu'un gros tableau qui empêche l'affichage de la page et qui mentionne l'erreur.

```
$search = "%".$_POST['search']."%";
```

j'ai utilisé wamp pour ce projet et le SGBD MySQL. Avec MySQL le caractère % désigne l'équivalent du * en msdos qui signifie n'importe quel caractère et peu importe la longueur. Autrement dit on récupère la valeur entrée par l'utilisateur et on demande à la base de donnée tous les noms de jeux qui contiennent les caractères que l'utilisateur a entrés.

```
$sql = $bdd->prepare(" requête SQL ");  
$sql->execute([$search]);
```

on prépare un PDOStatement qui prépare la requête. L'objet PDO permet la connexion à une base de données, alors que l'objet PDOStatement représente une requête préparée, prête à être exécutée avec différentes valeurs de paramètres

La méthode "prepare" permet de préparer une requête SQL. Cela permet d'éviter les attaques de type injection SQL qui peuvent être utilisées pour compromettre la sécurité d'une application.

La méthode "execute" est utilisée pour exécuter la requête préparée en y passant les valeurs des paramètres à remplacer dans la requête