**Une image contenant texte, clipart

Description générée automatiquementConcepteur Développeur d’Applications**

**Evaluation Rattrapage**

**PHP/HTML/CSS/JS : Application Web**

**Durée : 2 h**

**Documents autorisés : Oui**

**Internet autorisé : Oui**

**Consignes :**

Ce document est à compléter et à mettre à la racine de votre espace réseau dans le dossier **EVAL\_PHP\_Rattr\_[Prénom]**

L’évaluation porte sur **vos connaissances en matière de développement web**.

* Les réponses doivent être précises. Mettre des extraits courts de code (uniquement ceux qui répondent à la question), préciser le fichier et la méthode s’il y a lieu.

**Sujet :**

**/!\ AVANT TOUTE CHOSE JE M’EXCUSE PAR AVANCE DES FAUTES DE FRAPPES ET DES ERREURS DE GRAMMAIRES/CONJUGAISON /!\**

# Quels sont les points techniques qui permettent de passer de :

## La page connexion au catalogue

Dans le fichier ActionConnexion.php, on redirige vers la page catalogue que quand la connexion est réussie avec cette instruction :

header("location:index.php?page=Catalogue");

## La liste des articles au formulaire de modification

Dans le fichier ListeArticles.php, on fait un echo avec la route vers le formulaire de modification, comme ceci :

echo '<div class="caseListe"> <a href="index.php?page=FormArticles&mode=Modifier&id='.$unObjet->getIdArticle().'"><i class="fas fa-pen"></i></a></div>';

La partie « page=FormArticles » c’est le nom de la route qu’on met dans le tableau routes dans l’index.php qui se trouve à la racine du projet.

"FormArticles" => ["PHP/VIEW/FORM/", "FormArticles", "Formulaire Articles", 2, false],

Ce tableau associatif contient des tableaux qui comporte :

* Le dossier dans lequel se trouve le fichier php
* Le nom du fichier php
* Le titre qu’on souhaite faire apparaitre lors de l’affichage de cette page
* Le role de l’utilisateur pour gérer l’accès à cette page.
* Un booléen pour savoir si c’est une api ou non, dans le cas où c’est une api on affichera que le contenu sans head, header,nav, footer.

Cette route précise à l’index quel fichier PHP on doit appeler et grâce à la méthode afficherPage on peut afficher cette page dans le navigateur, cette méthode se trouve dans le fichier Outils.php et nous servira pour chaque page qu’on souhaite afficher.

*Je ne la montre pas parce qu’elle est trop longue*

La raison pour laquelle on précise le mode et l’id, c’est parce qu’on va en avoir besoin dans le fichier FormArticles.php.

En effet celui-ci sert pour différents types de traitement, il sert pour l’affichage, l’ajout, la suppression et la modification.

if (isset($\_GET['id'])) {

    $elm = ArticlesManager::findById($\_GET['id']);

} else {

    $elm = new Articles();

}

Pour pouvoir entrer les valeurs dans les inputs quand on est en mode modification, il faut récupérer les informations de l’enregistrement sur lequel on se trouve, en envoyant l’id on peut récupérer l’objet complet grâce à son manager qui va appeler la méthode findbyId

    public static function findById($id)

    {

        return DAO::select(Articles::getAttributes(),"Articles",["IdArticle" => $id])[0];

    }

Qui va simplement prendre en paramètres l’id et appeler la méthode select du DAO qui lui est le seul à pouvoir faire des requêtes en base de données.

Pour renvoyer un objet de type Articles le select a besoin :

* D’un tableau de colonnes, dans le cas présent on a une méthode statique dans toutes les classes qui permettent de renvoyer un tableau avec le nom des colonnes en base de données.

private static $\_attributes=["IdArticle","LibelleArticle","DescriptionArticle","PrixArticle","Photos","IdTypeArticle"];

    public static function getAttributes()

    {

        return self::$\_attributes;

}

* Le nom de la table su laquelle on souhaite faire la requete SQL
* Un tableau qui servira pour les conditions dans le cas présent on cherche l’idArticle.

Le [0] est présent parce que la méthode select renvoie toujours un tableau même s’il n’y a qu’une données.

*Je ne montre pas la méthode par ce qu’elle est trop longue.*

Et donc grâce à cet objet Articles je peux faire l’afficher ce que je veux sur cet enregistrement de la base de données.

## Comment s’effectue le retour

Le retour c’est juste un lien qui renvoie vers la page qu’on veut, grâce à la balise « a » qui redirige au clique sur celle-ci, voici un exemple :

<a href="index.php?page=Accueil"><button><i class="fas fa-sign-out-alt fa-rotate-180"></i></button></a>

# La liste des articles me fournit un IdTypeArticle, comment récupérer un texte ?

La classe DAO permet de faire des requêtes en base de données.  
Chaque classe à son Manager associé sauf si on a pas besoin de récupérer des données dans la base de données.

Pour chaque article, il faut appeler le manager TypesArticles et appeler le findById qui retournera un objet TypesArticles dans lequel on pourra afficher ce qu’on veut par exemple le libelle de cette manière :

echo '<div class="caseListe left color5  gridCase">'.TypesArticlesManager::findById($unObjet->getIdTypeArticle())->getLibelleTypeArticle()

'</div>';

Dans le projet d’exemple on appelle la méthode toString : qui retourne le libelle du type article

public function toString()

    {

        return $this->getLibelleTypeArticle();

    }

# Dans le formulaire article

## Comment faire apparaitre l’image ?

Quand on est dans une formulaire Articles on a forcément un objet Articles qui est récupéré il est soit vide, soit plein.

if (isset($\_GET['id'])) {

    $elm = ArticlesManager::findById($\_GET['id']);

} else {

    $elm = new Articles();

}

Pour faire apparaitre l’image il suffit de créer une div et de mettre une balise image en précisant la source, cette source c’est le dossier IMG qui est sur le serveur suivi du nom de l’image avec son extension.

Le nom de l’image et l’extension on le stock dans l’attribut qui se trouve dans la classe Articles.php

private $\_Photos;

Il suffit donc d’appeler le getter getPhotos() et il va renvoyer ce qui est contenu dans l’attribut $\_Photos, le nom de l’image suivi de son extension qui est enregistré en base de données.

public function getPhotos()

    {

        return $this->\_Photos;

    }

Ce qui donne un affichage de cette manière écrite dans FormArticles.php.

echo '<div class="caseForm donneeForm"><img src="./IMG/'.$elm->getPhotos().'" /></div>';

NB : soit dit en passant la première lettre du nom des attributs ne doit pas être une majuscule.

## Comment transformer l’input TypeArticle pour avoir une liste déroulante ?

Pour créer un select facilement le framework met à disposition une fonction dans le fichier Outils.php, qui est un fichier qui regroupe différentes fonctions utiles et souvent utilisées dans le framework.

creerSelect(?int $valeur, string $table, array $nomColonnes, ?string $attributs = "", array $condition = null, string $orderBy = null)

Cette fonction prend plusieurs paramètres :

* L’id sur lequel on veut que ce soit selected par défaut, ça sert notamment en mode afficher modifier ou supprimer. il peut être null dans le cas où on est en mode ajout par exemple.
* Le nom de la table sur laquelle on fait la requete
* Le nom des colonnes qui nous importe et donc celle dont on souhaite faire afficher leur valeur dans le select.
* C’est un string qui va permettre de mettre des classes par exemple, l’attribut disabled aussi par exemple.
* Les conditions qui est un tableau associatif, qui est pareil que le select du DAO qui sert à composer la partie après le WHERE dans une requête SQL.
* Le ORDER BY qui est un string, servira à classer par ordre croissant ou décroissant en fonction des colonnes qu’on souhaite voir afficher.

Dans le FormArticles.php pour afficher une liste déroulante qui contient les libelles de types d’articles il suffit de l’écrire de cette façon :

creerSelect($elm->getIdTypeArticle(),"TypesArticles",["LibelleTypeArticle"],$disabled)

$elm->getIdTypeArticle() qui retourne un id ou null si l’objet et vide.

TypesArticles qui est le nom de la table sur laquelle on veut faire le select

Le tableau qui contient le nom de la colonne LibelleTypeArticle

Et pour finir le $disabled qui est une variable qui est soit un espace soit qui contient disabled afin de mettre le select en disabled ou non suivant le mode du formulaire.

$disabled = " ";

switch ($mode) {

    case "Afficher":

    case "Supprimer":

        $disabled = " disabled ";

        break;

}

## Quels fichiers, quelles méthodes entre en jeu ?

Ayant détaillé au-dessus je fais un récapitulatif de ce qui est appelé dans ce fichier FormArticles.php

Tout d’abord, si l’id est dans l’URL au moment de la redirection on va appeler le manager ArticlesManager.Class.php pour lui demander de nous récupérer l’objet Articles en utilisant sa méthode findById() avec l’id en passé en paramètre.

Cette méthode retourne un objet Articles donc va utiliser la méthode hydrate() qui va permettre de construire l’objet Articles.

Cet objet sera donc instancié par la classe Articles qui se trouve dans le fichier Articles.php

if (isset($\_GET['id'])) {

    $elm = ArticlesManager::findById($\_GET['id']);

} else {

    $elm = new Articles();

}

Ensuite on peut appeler la méthode CreerSelect() qui se trouve dans le fichier Outils.php

*Elle est trop longue pour que je l’affiche ici.*

## Quels sont les modifications à apporter pour permettre le changement d’image ?

Pour permettre le changement d’image on met l’input caché qui contient le nom de la photo et on extension.

<input type="hidden" name="AnciennePhotos" value="'.$elm->getPhotos().'">

Le reste se passe en js, on charge le js dans le footer dans le fichier footer.php .

if($page[1]="FormArticles") echo ' <script src="./JS/FormArticles.js"></script>';

Pour être honnête dans le js j’ai compris ce qu’il se passait.  
  
Mais je n’ai pas compris ces lignes.

inputFile = document.querySelector("input[type=file]");

if (inputFile!=undefined)

{

    inputFile.addEventListener("change", choixImage);

}

Comment on peut ajouter un eventListener si l’élement est introuvable dans la DOM ?

Sinon dans le js on ajoute la function

function choixImage(e)

Qui va simplement regarder si l’input contient un fichier ou non et s’il n’en contient pas on va charger l’image de base qui est déjà enregistré en base de données et on va modifier la source de la balise image dans le formulaire et ajouter le lien de l’image.

Si c’est une nouvelle image on va mettre input modifImage pour qu’après dans le ficher action on fasse le traitement nécessaire en cas de changement.

Dans ActionArticles.php

On check si l’input avec le nom modifimage existe

if (isset($\_POST["modifImage"]))

et s’il existe on supprime l’ancienne image du serveur

unlink("IMG/" . $\_POST['AnciennePhotos']);

$elm->setPhotos(chargerImage());

Et on met l’autre image.

function chargerImage()

{

    if (is\_uploaded\_file($\_FILES["Photos"]["tmp\_name"]))

    {

        $leNom = uniqid('jpg\_') . '.jpg';

        move\_uploaded\_file($\_FILES['Photos']['tmp\_name'], 'IMG/' . $leNom);

    }

    return $leNom;

}

Cette fonction va enregistrer l’image sous un nom unique pour ne jamais melanger les image entre elles et surtout aussi pas sécurité.

DANS LE CAS ou l’image ne change pas

else $elm->setPhotos($\_POST['AnciennePhotos']);

On remet ce lien d’image et on va persister à l’aide du DAO.

# Dans les classes, au niveau des setters, que veut dire ?int ?

?int veut dire qu’on attend soit null, soit un int.

# Traduction : quels sont les points techniques qui entrent en jeu ? (A tous les niveaux)

On a une table texte dans la base de données.

On a un texteManager qui va chercher en fonction du code qu’on a mis en paramètre le résultat de la traduction.

La fonction texte dans outils.php :

function texte($codeTexte)

{

    $retour=TextesManager::findByCodes($\_SESSION['lang'], $codeTexte);;

    if ($retour==false) return $codeTexte;

    return $retour;

}

Comme on peut le voir le manager demande le code de la lang par défaut c’est ‘FR’ et le code texte qui est le paramètre de la fonction texte.

Si la requête retourne false c’est que le manager n’a pas trouvé l’enregistrement en base de données.

Dans ce cas-là on afficher le code texte on ne va pas afficher une erreur.

Sinon on retourne la traduction enregistrée en base de données.

On utilise la fonction de cette manière par exemple :

echo '<label for=LibelleArticle class="caseForm labelForm">'.texte("LibelleArticle").'</label>';

Cette fonction est appelable partout comme toutes les fonctions qui se trouve dans le fichier outils.php.

# Dans quel(s) contexte(s) sert le fichier DAO.Class.php

Le ficher DAO sert à partir du moment où on a besoin de faire une requête en base de données, il est appelé par les managers et eux sont appelés par pleins de fichier comme les listes, les formulaires, les actions etc ..

# Utilisateurs : A quel endroit intervenir pour forcer le rôle et le mot de passe ?

Au moment de l’inscription, on ne demande pas à l’utilisateur de saisir son rôle mais du coup notre objet utilisateur est partiellement remplis.

Donc il faut le remplir « sois même » et donc dans le fichier ActionInscription.php il faut écrire ceci :

$u = new Utilisateurs($\_POST);

        $u -> setRole(1);

        $u->setMotDePasse(crypte($u->getMotDePasse()));

# Panier Utilisateur : Quelles sont les actions à mener pour récupérer les lignes du panier du client ?

Dans le fichier ListeLignespaniers.php

On récupère le panier du client de cette façon :

$cmd = PaniersManager::getList(null, ["idClient" => $\_SESSION['utilisateur']->getIdUtilisateur()]);

if (!$cmd)

    echo '<div class="gridListe center">Pas de panier en cours</div>';

else {

    $objets = LignesPaniersManager::getList(null, ["idPanier" => $cmd[0]->getIdPanier()]);

//Some Code…

}

Dans le cas où il a un panier on va récuperer la liste des ignes paniers qui sont lié à son panier.

Ce qui lie les tables paniers et Lignespaniers c’est l’idPanier qui est une Foreign Key dans la table Lignespaniers.

# Comment gérer les listes vides ? A quel(s) niveau(x) ?

Ici par exemple si on récupère une liste vide, il suffit de faire un if pour voir si le tableau est vide on affiche un message sinon ça veut dire qu’il est rempli donc on fait le traitement derrière

if (!$cmd)

    echo '<div class="gridListe center">Pas de panier en cours</div>';

else {

    $objets = LignesPaniersManager::getList(null, ["idPanier" => $cmd[0]->getIdPanier()]);

//Some Code…

}

# Comment fixer le footer en bas de la page et permettre de voir toute la page ?

On met le footer en position fixed.

Le css du footer  dans le fichier style.css :

footer{

    background-color: var(--color5);

    color: var(--color1);

    font-size: 0.8em;

    position: fixed;

    bottom:0;

    width: 100%;

     padding: 2em;

}

Le css du main dans le fichier style.css :

main{

    padding-bottom: 20vh;

}

Avec ces instructions, les informations ne passent pas en dessous du footer, elle reste toujours au-dessus.

# Présentation sous forme de Grid :

## Où intervenir, comment le paramétrer ?

Pour utiliser les grids il faut utiliser un élément HTML par exemple une div et lui attribuer une classe CSS pour utiliser le comportement d’une grid.

echo '<div class="grid-col-8 gridListe">';

dans le fichier CSS grid.css il y’a tout les types de grids dont l’application avait beosin durant la génération du projet.

.grid-col-8 {

    display : grid;

    grid-template-columns : 2fr 2fr 2fr 2fr 2fr 0.5fr 0.5fr 0.5fr;

}

Il suffit de mettre l’attribut display grid, de spécifier l a taille des colonnes en « fr » qui veut dire « fragment » pour répartir comme on veut la taille des colonnes.

Et ensuite tous les éléments qui seront dans cette div se mettront de façon à former une grille avec le nombre de colonnes qui ont été spécifié dans l’attribut grid-template-columns .

## Comment obtenir une case qui fusionne 2 cases adjacentes ?

Dans le fichier grids.css qui sert pour les grids.

Il faut utiliser cet attribut qui permet de faire en sorte de fusionner deux cases adjacentes.

grid-column:1/3;

On met cet attribut dans une classe css et on attribue la classe a un élément HTML qui se trouve dans une grid.

# Gestion des erreurs : que faut il faire pour mettre en place une gestion d’erreur ? A quel endroit ?

J’avoue qu’il me semble que c’est Nils qui l’a fait et j’ais spécialement étudié le code.

A ce que j’ai compris, lors de la connexion dans le fichier ActionConnexion.php si l’adresseMail ou le Mdp est incorrect on renvoie l’utilisateur vers la page Erreur.php.

header("location:index.php?page=Erreur&cible=Connexion&codeErreur=MdpIncorrect");

la cible correspond à la page qui a appelé la page Erreur et le codeErreur on met le code qu’on veut parce qu’il sera traité dans la page Erreur.

Dans la page Erreur.php on affiche le code de l’erreur et le titre de la page Erreur .

 if (isset($\_GET['codeErreur'])) {

            echo texte($\_GET['codeErreur']);

        }else{

            echo texte($titre);

        }

if (isset($\_GET['cible'])) {//si on a une page où retourner, on y va

                echo '<a href="index.php?page='.$\_GET["cible"].'"><i class="fas fa-sign-out-alt fa-rotate-180"></i></a>';

            }else{//sinon, go connexion

                echo '<a href="index.php?page=Connexion"><i class="fas fa-sign-out-alt fa-rotate-180"></i></a>';

            }

Et ensuite on va mettre un bouton retour qui lui va rediriger vers la page cible c’est-à-dire la page où l’on était avant l’erreur.

Si on veut ajouter d’autres erreur c’est simple il suffit d’ajouter cette ligne en mettant la cible c’est la page dans laquelle on ajoute cet header et le codeErreur c’est simplement un code Erreur qu’on choisit et on fait en sorte de mettre un nom différent si l’erreur est différente des autres erreurs.

header("location:index.php?page=Erreur&cible=Connexion&codeErreur=MdpIncorrect");