

Secteur Tertiaire Informatique
Filière « Étude et développement »

Séquence « Développer des pages Web »

TP PHP

Apprentissage

Mise en pratique

Évaluation



Mise en pratique : Découverte de PHP

Afpa © 2019 – Section Tertiaire Informatique – Filière « Étude et développement »

TABLE DES MATIÈRES

1.	INSTRUCTION PHP ET LANGAGE HTML	4
2.	LISTES, TABLEAUX INDEXÉS ET TABLEAUX ASSOCIATIFS	5
3.	TRAITEMENTS SUR LES ÉLÉMENTS D'UN TABLEAU	6
4.	CLASSES ET OBJETS	6
5.	MODULES ET MODULES DE CLASSE.....	8
6.	LES ERREURS ET LES EXCEPTIONS	9
7.	VALIDATION D'UNE DATE POUR UN FORMULAIRE.....	10
8.	REDIRECTION À L'ISSUE D'UN TRAITEMENT.....	10
9.	SYSTÈME CRUD SUR UNE TABLE SIMPLE.....	11
10.	SYSTÈME CRUD EN ARCHITECTURE MVC	13
11.	AUTO RETRO CHIC EN ARCHITECTURE MVC.....	14
12.	SYSTÈME CRUD POUR LA SOCIÉTÉ ABI.....	18
12.1	AJOUT D'UN NOUVEAU CLIENT	19
12.2	MODIFICATION D'UN CLIENT	20
12.3	SUPPRESSION D'UN CLIENT	21
12.4	MESSAGES D'ERREUR	22
13.	SYSTÈME D'AUTHENTIFICATION EN AJAX POUR ABI.COM ...	24
14.	FORMULAIRE DE CONNEXION EN ARCHITECTURE MVC.....	25
15.	CRÉATION D'UNE CLASSE POUR PRÉSENTER UNE TABLE MYSQL.....	26
16.	CLASSE, HÉRITAGE ET CLONAGE	29

17.	FORMULAIRE DE CONTACT EN PHP-POO-AJAX-MYSQL ET ENVOI DE MAIL.....	30
18.	UPLOADER UNE IMAGE VERS LE SERVEUR	34
19.	PRÉSENTATION PAGINÉE D'UNE TABLE MYSQL	35
20.	ATTACHEMENT DE DONNÉES PAR FORMULAIRE ET STOCKAGE DANS UNE BASE MYSQL	37
21.	AUTHENTIFICATION, SESSION SÉCURISÉE, AJAX ET PROCÉDURES STOCKÉES.....	39
22.	CRÉATION D'UN COMPOSANT DE SESSION	43
23.	SESSION ET AUTHENTIFICATION D'UTILISATEURS	44
24.	JSON ET AUTO COMPLÉTION	45
25.	COMPOSANT JQUERY DATATABLES, PHP-MVC	46
26.	ENVOI DE MAIL AVEC LE COMPOSANT PHPMAILER.....	48
27.	ATELIER TWIG.....	48

1. INSTRUCTION PHP ET LANGAGE HTML

Dans ce TP, nous allons étudier PHP dans un contexte HTML avec différents styles de rédaction.

Une table de multiplication									
1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

1 – Créer un programme PHP **mult_html.php** qui affiche une table de multiplication 10*10. Efforcez-vous d'utiliser au maximum les balises HTML et employez le moins possible les balises `<?php` et `?>`.

2 – Ecrivez un nouveau programme **mult_php.php** de multiplication mais qui n'utilise aucune balise HTML.

3 – Comparez les deux styles.

4 – Trouvez un moyen de n'utiliser qu'une seule fois l'instruction echo (ou print) à partir du second programme (**mult_sans_echos.php**).

2. LISTES, TABLEAUX INDEXÉS ET TABLEAUX ASSOCIATIFS

La taille des Tableaux en PHP n'est pas définie à leur création, les index peuvent être numériques ou alphanumériques, et le langage propose de nombreuses façons de les exploiter. Seule difficulté notoire, le passage d'un tableau à une fonction se fait par valeur.

- 1- Trouvez un moyen d'échanger les valeurs de deux variables **\$a** et **\$b** en une seule instruction, vous pourrez utiliser l'élément de langage PHP **list ()**. Créez un script **tableaux.php** pour le tester.
- 2- Déclarez un tableau **\$pays** indexé par des entiers contenant les noms de trois pays.
- 3- Affichez ce tableau à l'aide de l'instruction **var_dump**.
- 4- Parcourez le tableau à l'aide d'une boucle **for**, vous pourrez utiliser la fonction **count ()**.
- 5- Parcourez de nouveau le tableau à l'aide d'une boucle **foreach**.
- 6- Pour quelle raison n'est-il pas approprié d'utiliser une boucle **do ... while**.
- 7- Ajoutez des index sous forme de chaînes de caractères pour associer les capitales aux pays du tableau **\$pays**.
- 8- Combien vaut l'expression **count(\$pays)** ?
- 9- Comment parcourir ce tableau pour afficher chaque pays et sa capitale ?
- 10- Ajouter au script en cours une fonction **enumerer(\$t)** qui prend le tableau comme paramètre et qui affiche la valeur de chaque index alphanumérique.
- 11 Insérez une valeur **\$t[« capitale »] = « pays »** en créant une fonction **ajouter(\$t)** puis afficher les nouvelles valeurs du tableau résultant.
- 12 Affichez par **var_dump** le tableau après l'exécution de la fonction. Qu'en concluez-vous ?

3. TRAITEMENTS SUR LES ÉLÉMENTS D'UN TABLEAU

Créez un tableau contenant une liste d'adresses e-mail.

Extrayez le nom de serveur de ces données, puis réalisez des statistiques sur les occurrences de chaque fournisseur d'accès.

```
$tab=array("php7@free.fr","sacha8@gmail.com","ariel3@wanadoo.fr",  
"webmestre@wanadoo.fr","marcelduchamp9@gmail.com","picasso69@gmail.com",  
"vangogh6@gmail.com","matis63@free.fr","degas45@wanadoo.fr");
```

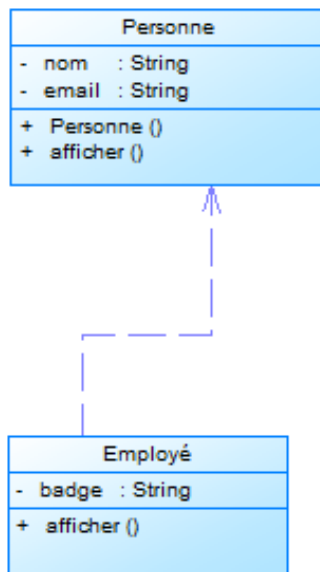
Résultat de sortie :

Fournisseur d'accès : free.fr =22.22 %
Fournisseur d'accès : gmail.com =44.44 %
Fournisseur d'accès : wanadoo.fr =33.33 %

Indices : Vous pourrez utiliser à bon escient les fonctions PHP suivantes : **explode()**, **array_count_values()**, **count()** et **round()**.

4. CLASSES ET OBJETS

Nous allons dans ce TP découvrir la syntaxe pour décrire une classe **Personne** exposant des champs et des méthodes. Cette classe sera ensuite étendue par le mécanisme d'héritage sous la forme d'une autre classe **Employe**.



- 1 – Créez un script **personnes.php** contenant une classe **Personne**. Déclarez dans cette classe deux champs **\$nom** et **\$email**.
- 2 – Hors de la classe, déclarez une variable **\$p1** qui est une instance de **Personne**. Affichez « **jean** » au champ **nom** de **p1** et jean@ailleurs.net au champ **email**.
- 3 – Effectuer un **var_dump** de la variable **p1**.
- 4 – Ajoutez une fonction **constructeur** à l'intérieur de la classe **Personne**. Initialisez les champs **nom** et **email** aux valeurs « **sans nom** », « **sans email** ».
- 5 – Déclarez une autre instance **p2** de la classe **Personne** puis affichez-la au travers d'un **var_dump**. Qu'en concluez-vous ?

6 – Ajoutez à la classe **Personne** une fonction **afficher()** qui imprime la valeur des champs **nom** et **email**. Appliquez cette méthode à l'instance **p1** puis à **p2**.

7 – Créez une classe **Employe** qui hérite de la classe **Personne**. Adjoignez-y un champ **\$badge**.

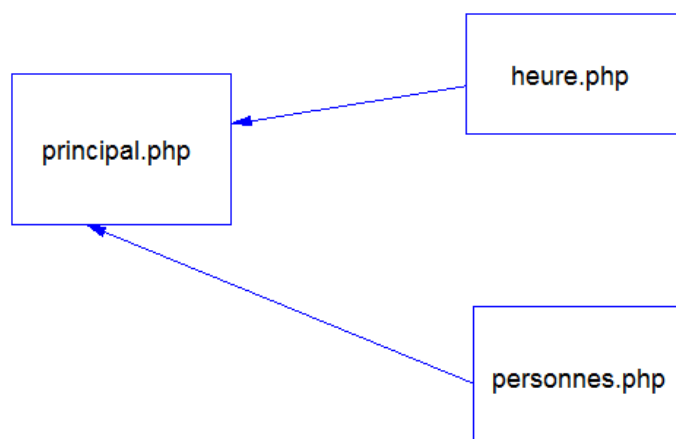
8 – Ajoutez une méthode **constructeur** à la classe **Employe** qui appelle la méthode **constructeur** de la classe parente **Personne** pour l'objet courant. Cette méthode initialise aussi le badge à « **numéro de badge inconnu** ».

9 – Déclarez une instance **\$e1** dans la classe **Employe**. Appliquez la méthode **afficher()** à cette instance. Qu'en concluez-vous ?

10 – Ecrivez une nouvelle version de la méthode **afficher()** dans la classe **Employe** qui tient compte du badge. Rappelez cette méthode pour l'objet **\$e1**. Interprétez le résultat.

5. MODULES ET MODULES DE CLASSE

L'objet de ce TP est de découvrir les conséquences du choix d'une portée de déclaration d'une variable ou d'une fonction dans un module inclus par la directive **include**.



- 1 – Créez un script **principal.php** contenant une page HTML vierge.
- 2 – Créez un autre script **heure.php** qui imprime l'heure au format de votre choix.
- 3 – Intégrez ce module dans le premier.
- 4 – Incluez à présent le script **personnes.php** réalisé lors du TP précédent. Qu'en concluez-vous ?
- 5 – Quelles règles proposeriez-vous pour segmenter une application en modules ?

6. LES ERREURS ET LES EXCEPTIONS

Tout programme informatique doit prendre en charge les erreurs déclenchées par l'utilisateur ou par les défaillances du système. Une erreur qui n'est pas prise en charge provoque généralement l'interruption du script PHP alors qu'un mécanisme standard, l'instruction **try ... catch finally** prévient ce type de blocage. Une erreur gérée devient une exception.

- 1 – Créez un fichier **messages.txt** contenant les entrées suivantes :

Clé	Message
titre	Bienvenue
invite	Quel est votre nom ?

Optez pour une représentation simple de la base de messages, telle que clé=message.

- 2 – Créez un fichier **modele.php** contenant les éléments suivants :

Tableau	\$messages
Fonction	LoadMessages()
Fonction	getMessage(\$key)

- 3 – Ajoutez une page **erreur.php** incluant le module précédent et utilisant la fonction **getMessage(\$key)** pour afficher les différents messages accessibles.

- 4 – Vérifiez le bon fonctionnement de cette page (**erreur.php**).

- 5 – Changez le nom du fichier lu par **loadMessage()** de manière à provoquer une erreur lors du chargement.

- 6 – Utilisez l'instruction **try ... catch** pour contrôler le chargement du fichier.

- 7 – Déclenchez vous-même des exceptions à l'aide de l'instruction **throw** pour améliorer la robustesse du programme.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

7. VALIDATION D'UNE DATE POUR UN FORMULAIRE

La validation d'un champ particulier contenant une date demande un peu de travail. Le format d'une date évolue peu mais le besoin, lui, est constant. Nous prévoyons ici un programme de vérification qui pourra être utilisé dans différentes applications PHP.

1 – Créez une page PHP auto référente contenant un formulaire, un champ de type texte (« jour »), et un bouton pour soumettre la requête.

2 - Définissez une fonction PHP, **verifier(\$d)**, qui sera chargée de vérifier le champ dont le nom est passé en paramètre.

3 – Commencez par afficher la valeur du champ correspondant : un simple **echo** suffira.

4 – Concevez un masque d'expression régulière pour vérifier le format **jj/mm/aaaa**.

5 – Modifiez la fonction **verifier()** pour qu'elle renvoie un message d'erreur lorsque le champ ne valide pas le masque.

6 – Ajoutez à votre code une vérification de la date, en fait il faut contrôler cette fois si elle correspond bien à une date effective.

8. REDIRECTION À L'ISSUE D'UN TRAITEMENT

Lorsqu'une page PHP valide les données d'un formulaire, elle doit généralement afficher un message à l'utilisateur. Ce message peut faire partie de la page PHP qui a assuré le traitement mais aussi être intégré à une autre page. Généralement, les formulaires exposés par une page **form1.php** postent les données sur eux-mêmes, ce qui simplifie l'établissement d'un rapport d'erreurs de saisie.

Il faut par conséquent aider le programmeur à se diriger vers une autre page.

Nous allons au travers de ce TP mettre en place une logique pour traiter un formulaire et pour activer les différentes pages qui composent l'application à laquelle appartient le formulaire.



1 – Créez une page **form1.php**

Placez dans cette page un formulaire auto référant, deux champs **email** et **password**, ainsi qu'un bouton de validation de type **submit**.

2 – Déterminez si la page est publiée initialement ou s'il s'agit d'un formulaire qui revient (post back) suite à l'action sur le bouton **submit**. Utilisez pour cela une variable **\$msg** qui affichera « Saisie obligatoire » lors de la première publication de la page.

3 – Dans le second cas, vérifiez que les champs **email** et **password** sont bien renseignés et qu'ils correspondent à des couples de valeurs prises dans le tableau qui suit :

email	password
jean_valjean@academie.net	hugo
steve_ostin@lesseries.org	3md
david_banner@marvel.com	hulk

Utilisez une variable **\$erreur** positionnées à **true** ou **false**. Procédez à un rapport d'erreur en cas d'échec.

4 – Si l'identification a réussi, redirigez l'utilisateur vers **form2.php** qui vérifie la présence d'un **cookie** généré par **form1.php**.

5 – **form2.php** affiche un message de bienvenue.

9. SYSTÈME CRUD SUR UNE TABLE SIMPLE

Le But de ce TP est de réaliser un système **CRUD** complet en utilisant le langage **PHP**. Le SGBD choisit est **MySQL**.

La page centrale de votre application devra exposer un simple tableau qui comportera une liste de Stagiaires. Vous devrez requêter sur une unique table **membres** issue de la Base de données **formation** dont le script de création vous est fourni dans les ressources liées à ce TP.

Vous n'aurez pas besoin de soigner ni la présentation des pages, ni la validation et la sécurisation des données, le but de cet exercice est avant tout de vous permettre d'utiliser les différentes solutions que proposent la classe **PDO** en terme de requête SQL.

La gestion des contrôles de formulaire et autres levées d'exceptions seront vues ultérieurement.

Vous utiliserez exclusivement les **requêtes préparées** afin de prémunir votre application contre d'hypothétiques attaques par **injection SQL**.

Les divers captures d'écran vous aideront dans cette mission ...

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

Liste des Stagiaires

ID Membre	Prénom Membre	Nom Membre	Suppression
3	Mehdi	HOUAM	Supprimer
4	Julie	MIDON	Supprimer
5	Shaquille	ROLLAN	Supprimer
8	Brandon	BERNABEN	Supprimer
9	Thomas	BORDAS	Supprimer
10	Léo	GAONAC'H	Supprimer
12	Julien	LAGARDE-MARTIAL	Supprimer
14	Christophe	MESTDAGH	Supprimer
15	Jason	RAFFY	Supprimer
16	Nicolas	RIVA	Supprimer
55	Pierre-Xavier	VERCELLY	Supprimer
56	Olivier	FOIDEFOND	Supprimer
57	Clément	AMOUROUX	Supprimer
60	Romaric	MEGERT	Supprimer
Ajouter un Stagiaire			

Modifier un Stagiaire

Prénom

Nom

Ajout d'un Stagiaire

Prénom

Nom

10. SYSTÈME CRUD EN ARCHITECTURE MVC

Le But de ce TP est de réaliser un système **CRUD** complet en utilisant le langage PHP dans une architecture **MVC**. À travers le **Modèle** vous devrez requêter sur une Base de Données incluse dans le SGBD **MySQL**.

La page centrale de votre application devra exposer un simple tableau qui comportera une liste de Stagiaires. Vous devrez requêter sur une unique table **membres** issue de la Base de données **formation** dont le script de création vous est fourni dans les ressources liées à ce livret de TP.

Je vous donne en archive les scripts qui amorcent une présentation de l'architecture requise à travers la récupération des données stagiaires et leur affichage à travers une vue.

Je traite aussi la suppression d'un stagiaire en respectant l'architecture MVC.

À vous de gérer l'**ajout** et la **modification** des données stagiaires en respectant ce **modèle MVC**.

Vous n'aurez pas besoin de soigner ni la présentation des pages, ni la validation et la sécurisation des données, le but de cet exercice est avant tout d'évaluer que vous avez bien assimilé le modèle MVC dans le cadre du développement d'une application web reposant sur les technologies **PHP/MySQL**. Les diverses requêtes devront être **préparées** en utilisant **PDO**.

Les divers captures d'écran vous aideront dans cette mission ...

Liste des Stagiaires

ID Membre	Prénom Membre	Nom Membre	Suppression
3	Mehdi	HOUAM	Supprimer
4	Julie	MIDON	Supprimer
5	Shaquille	ROLLAN	Supprimer
8	Brandon	BERNABEN	Supprimer
9	Thomas	BORDAS	Supprimer
10	Léo	GAONAC'H	Supprimer
12	Julien	LAGARDE-MARTIAL	Supprimer
14	Christophe	MESTDAGH	Supprimer
15	Jason	RAFFY	Supprimer
16	Nicolas	RIVA	Supprimer
55	Pierre-Xavier	VERCELLY	Supprimer
56	Olivier	FOIDEFOND	Supprimer
57	Clément	AMOUROUX	Supprimer
60	Romaric	MEGERT	Supprimer
Ajouter un Stagiaire			

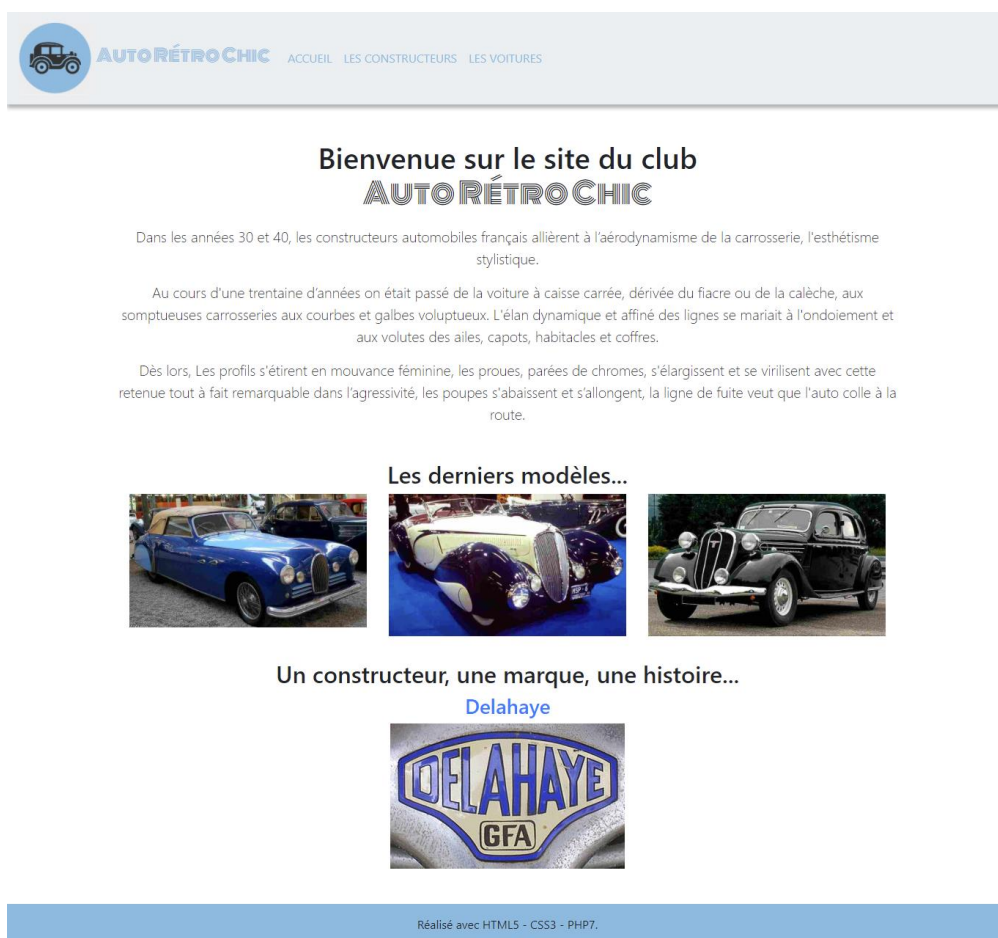
<h2>Modifier un Stagiaire</h2> <p>Prénom <input type="text" value="Mehdi"/></p> <p>Nom <input type="text" value="HOUAM"/></p> <p><input type="button" value="Envoyer"/> <input type="button" value="Annuler"/></p>	<h2>Ajout d'un Stagiaire</h2> <p>Prénom <input type="text"/></p> <p>Nom <input type="text"/></p> <p><input type="button" value="Envoyer"/> <input type="button" value="Annuler"/></p>
--	---

11. AUTO RETRO CHIC EN ARCHITECTURE MVC

Pour cet exercice vous allez devoir développer un site internet **responsif** qui expose ses pages à partir d'une architecture **MVC** élaborée en **PHP**. Les données textuelles et les libellés des images sont stockés dans une base **MySQL** pour laquelle je vous fournis le script de création **autoretrochic.sql**

Le responsif pourra être obtenu avec le Framework **Bootstrap**. Pour les faibles résolutions le menu se transformera en *Burger* ;-)

Voici un aperçu de la page d'accueil :



Sur cette page d'accueil est exposé un titre suivi d'*Auto Retro Chic* écrit avec la police de caractères :

```
<link href="https://fonts.googleapis.com/css?family=Monoton&display=swap" rel="stylesheet">
```

```
font-family: 'Monoton', cursive;
```

Le texte suivant pour vous éviter d'avoir à le saisir :

Dans les années 30 et 40, les constructeurs automobiles français allient à l'aérodynamisme de la carrosserie, l'esthétisme stylistique.

Au cours d'une trentaine d'années on était passé de la voiture à caisse carrée, dérivée du fiacre ou de la calèche, aux somptueuses

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

carrosseries aux courbes et galbes voluptueux. L'élan dynamique et affiné des lignes se mariait à l'ondoiement et aux volutes des ailes, capots, habitacles et coffres.

Dès lors, les profils s'étirent en mouvance féminine, les proues, parées de chromes, s'élargissent et se virilisent avec cette retenue tout à fait remarquable dans l'agressivité, les poupes s'abaissent et s'allongent, la ligne de fuite veut que l'auto colle à la route.

Ensuite, une section dans laquelle s'affiche les 3 dernières images de voitures selon leur **id** respectif dans la table **t_voiture**.

Ensuite, une section dans laquelle s'affiche un constructeur choisit de façon **aléatoire** dans la table **t_construct**. Vous pourrez vous intéresser pour cela à la fonction SQL **RAND ()**.

Le header et le footer de la page appartiennent au layout général.

Les liens **Les Constructeurs** et **Les Voitures** du Menu exposent tour à tour tous les constructeurs et toutes les voitures comme il suit :



AUTORÉTROCHIC

[ACCUEIL](#) [LES CONSTRUCTEURS](#) [LES VOITURES](#)

Bienvenue sur le site du club AUTORÉTROCHIC

La liste des constructeurs



Réalisé avec HTML5 - CSS3 - PHP7.



AUTORÉTROCHIC

[ACCUEIL](#) [LES CONSTRUCTEURS](#) [LES VOITURES](#)

Bienvenue sur le site du club AUTORÉTROCHIC

La galerie des voitures








Réalisé avec HTML5 - CSS3 - PHP7.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

Puis pour chaque **constructeur** et **voiture** une page de détail sera exposée :

<div>ACCUEIL - LES CONSTRUCTEURS - LES VOITURES</div> <div><h3>Chenard&Walcker</h3><p>Chenard et Walcker est une marque d'automobiles Française créée par Lucien Chenard et Henry Walcker en 1899. Quatrième constructeur Français en volume vers 1922 sous la direction de l'ingénieur Lucien Chenard, il est le premier vainqueur de la première édition des 24 Heures du Mans en 1923 et collectionne les victoires en compétition automobile. Son succès commercial est assuré par ses luxueuses automobiles de sport et de tourisme. Outre la gamme automobile, des camions sont produits à partir de 1931. La société est filialisée par la Société des usines Chausson en 1936 et la production automobile cesse en 1940. Après la Seconde Guerre mondiale, la société produira des camions et utilitaires commercialisés par la marque Peugeot jusqu'en 1955.</p><div>Réalisé avec HTML5 - CSS3 - PHP7.</div></div>	<div>ACCUEIL - LES CONSTRUCTEURS - LES VOITURES</div> <div><h3>CHENARD&WALCKER AIGLE 8 U16</h3><p>1937</p><p>La Chenard & Walcker Aigle 8 U16 fut produite en 1937 en 15 exemplaires. caractéristiques: Le modèle U16 (phares en dehors et capot à l'ancienne) avec le V8 Ford, 3,6 litres, est une présérie de 15 automobiles qui a servi de test pour relancer en 37 la marque CW qui avait fait faillite 4 mois avant.</p><div>Réalisé avec HTML5 - CSS3 - PHP7.</div></div>
---	---

Vous devrez, à chaque fois que ça sera nécessaire, prévoir une levée **d'Exception** et renvoyer à l'utilisateur un **message d'erreur** approprié dans par exemple une **vue** adaptée à cet effet.

12. SYSTÈME CRUD POUR LA SOCIÉTÉ ABI

Les commerciaux de la société de service informatique **ABI** effectue une collecte manuscrite des informations sur les Entreprises clientes qu'ils démarchent.

La société **ABI** a décidé de mettre en place une solution informatisée pour améliorer leur productivité.

Vous allez devoir développer une partie de cette application maison **ABI.COM**

Vous devrez implémenter le code nécessaire pour réaliser les différentes pages qui suivent. En liaison avec ce TP, le script de création et d'initialisation de la Base de données vous est fourni **abicom.sql**

Pour commencer la page d'accueil, elle dispose d'un champ de recherche qui porte sur la Raison Sociale de l'Entreprise.

Au-dessous, la liste des différentes entreprises qui correspond au critère de recherche. En cliquant sur le logo **ABI**, l'utilisateur recharge la page.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

A partir de cet écran, l'utilisateur peut accéder à un formulaire d'ajout d'un nouveau Client en suivant le lien : [Ajouter un Nouveau Client](#)

Il peut supprimer un Client de la liste en suivant le lien : [Suppression de ce Client](#)

Il peut modifier les données d'un Client en suivant le lien de la [Raison Sociale](#) de la liste.



Liste des Clients

Raison Sociale :

Ajouter un Nouveau Client

ID Client	Raison Sociale	Code Postal Client	Ville Client	CA (€)	Téléphone Client	Type Client	Nature Client	Supprimer un Client
2	AgriCorp	19033	Brive	15000	0555231245	Public	Secondaire	Suppression de ce Client
3	AgriMoon	19500	Objat	80000	0555121415	Public	Principale	Suppression de ce Client
4	AgriGel	87000	Limoges	40000	0555457458	Privé	Principale	Suppression de ce Client
10	Silice	75008	Paris	100000	0612121212	Public	Principale	Suppression de ce Client
11	Dastin	81000	Narbonne	45000	2323232323	Privé	Principale	Suppression de ce Client
13	Accenture Agro	31000	Toulouse	800000	0145477896	Privé	Secondaire	Suppression de ce Client
18	Xander	92300	Levallois Perret	180000	0356895874	Public	Secondaire	Suppression de ce Client
19	Brut	75002	Paris	1000000	0212457889	Privé	Secondaire	Suppression de ce Client

**ABI.COM**

Qui sommes-nous ?

- Rejoignez-nous
- Actualité
- Contact

12.1 AJOUT D'UN NOUVEAU CLIENT

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

Retour

Ajout d'un Nouveau Client

Raison sociale	<input style="width: 95%;" type="text"/>	
Type Client	<div style="background-color: white; color: black; padding: 2px 5px;">Public ▼</div>	
Téléphone Client	<input style="width: 95%;" type="text"/>	
Adresse Client	<input style="width: 95%;" type="text" value="25 Rue des Allouettes"/>	
Ville	<input style="width: 95%;" type="text"/>	
Code Postale	<input style="width: 95%;" type="text"/>	
Activité	<div style="background-color: white; color: black; padding: 2px 5px;">Administration ▼</div>	
Nature	<div style="background-color: white; color: black; padding: 2px 5px;">Principale ▼</div>	
CA	<input style="width: 95%;" type="text"/>	
Effectif	<input style="width: 95%;" type="text"/>	
Commentaires Commerciaux		
Valider Annuler		

ABI.COM

Qui sommes-nous ?

Rejoignez-nous

Actualité

Contact

Vous devrez effectuer tous les tests nécessaires afin que les données envoyées sur le serveur de BDD soient en adéquation avec le modèle.

12.2 MODIFICATION D'UN CLIENT

À partir de la liste des Clients, lorsque l'utilisateur clique sur le nom de l'Entreprise (Raison Sociale) il accède à une page de modification qui lui permet de modifier l'intégralité des informations qui concernent cette Entreprise :

Sur cette page de modification, les diverses **listes déroulantes** doivent être évidemment **initialisées** en fonction des données enregistrées dans la Base de données.

Avant de traiter la requête de modification, vous devrez effectuer là aussi tous les **contrôles** susceptibles d'assurer la conformité des données.

[Retour](#)



Modification du Client

Numéro de Client	<input type="text" value="10"/>
Raison sociale	<input type="text" value="Silice"/>
Type Client	<input type="text" value="Public"/>
Téléphone Client	<input type="text" value="0612121212"/>
Adresse Client	<input type="text" value="79 Rue des Maturins"/>
Ville	<input type="text" value="Paris"/>
Code Postale	<input type="text" value="75008"/>
Activité	<input type="text" value="Industrie"/>
Nature	<input type="text" value="Principale"/>
CA	<input type="text" value="100000"/>
Effectif	<input type="text" value="150"/>
Commentaires Commerciaux	<div style="border: 1px solid black; padding: 5px; min-height: 40px;">En négociation</div>
Valider Annuler	


ABI.COM

Qui sommes-nous ?

[Rejoignez-nous](#)
[Actualité](#)
[Contact](#)

12.3 SUPPRESSION D'UN CLIENT

Lorsque l'utilisateur choisit de supprimer un client, il sollicite un *script en aveugle* qui supprime de façon effective le client de la BDD et qui redirige immédiatement après sur la page d'accueil (Liste des Clients). Avant de rendre la suppression effective, vous déclencherez une **boîte de confirmation** en **JavaScript** pour demander à l'utilisateur s'il veut ou non supprimer ce Client.

9011CopFor - We...
Présentation des no...
localhost:8080 indique
Vous êtes certain de vouloir supprimer le Client : Xander ?
OK
Annuler

hier config.xml...
Cordova - Construir...

2	AgriCorp	19033						Ancienne	Suppression de ce Client
3	AgriMoon	19500						Principale	Suppression de ce Client
4	AgriGel	87000	Limoges	40000	0555457458	Public	Principale	Suppression de ce Client	
10	Silice	75008	Paris	100000	0612121699	Public	Principale	Suppression de ce Client	
11	Dastin	81000	Narbonne	45000	2323232323	Privé	Principale	Suppression de ce Client	
13	Accenture Agro	31000	Toulouse	800000	0145477896	Privé	Secondaire	Suppression de ce Client	
18	Xander	92300	Levallois Perret	180000	0356895874	Public	Ancienne	Suppression de ce Client	

Toutes vos requêtes seront **préparées** et vous devrez utiliser **PDO**.

12.4 MESSAGES D'ERREUR

Des messages d'erreur seront renvoyés à l'utilisateur lorsque les formats de données saisies ne correspondront pas avec ceux prévus en Bdd :

[Retour](#)

Ajout d'un Nouveau Client

Raison sociale	<input type="text" value="Test"/>	
Type Client	<input type="text" value="Public"/>	
Téléphone Client	<input type="text" value="06785419"/>	Numéro de Téléphone invalide.
Adresse Client	<input type="text" value="25Rue des Lapins"/>	Veuillez saisir une adresse postale valide
Ville	<input type="text" value="23 Le Saillant"/>	Veuillez saisir un nom de ville.
Code Postale	<input type="text" value="19"/>	Veuillez saisir un code postal valide.
Activité	<input type="text" value="Administration"/>	
Nature	<input type="text" value="Principale"/>	
CA	<input type="text" value="d"/>	Veuillez saisir un nombre.
Effectif	<input type="text" value="d"/>	Veuillez saisir un nombre.
Commentaires Commerciaux	<input type="text" value="Commentaires"/>	
<input type="button" value="Valider"/>		<input type="button" value="Annuler"/>

**ABI.COM**

Qui sommes-nous ?

[Rejoignez-nous](#)[Actualité](#)[Contact](#)

13. SYSTÈME D'AUTHENTIFICATION EN AJAX POUR ABI.COM

Vous allez devoir ajouter au projet ABI.com un système d'authentification reposant sur un appel **Ajax**, vous pourrez pour cela ajouter la bibliothèque **JQuery** à votre projet.

Voici une capture d'écran de ce module d'authentification :



Lorsque le **login** et le **mot de passe** ne correspondent pas à ceux utilisés dans la Table **users** du SGBD un message sera renvoyé en Front à l'utilisateur.

Vous pourrez utiliser la méthode JQuery **serialize ()** pour transmettre l'ensemble des données du formulaire au script de traitement.



Pour aller plus loin dans ce TP, vous pourrez ajouter un module de **création de compte**, avec **hashage** du mot de passe. Vous pourrez aussi prévoir un lien ou un bouton de **déconnexion** dans vos pages. Vous pourrez aussi envisager de créer une variable de **Session** qui vous permettra d'afficher ou non des parties du site en fonction du **rôle** de l'utilisateur. On pourra par exemple envisager que le **RespCom** pourra tout faire et que le **RespDev** ne pourra pas supprimer des Clients, qu'il ne pourra pas ajouter ni modifier une fiche Client

14. FORMULAIRE DE CONNEXION EN ARCHITECTURE MVC

Le but de ce TP est de réaliser un **formulaire** de **connexion** et **d'authentification** avec au préalable la **création** d'un **compte utilisateur** en **architecture MVC**.

Vous devrez **sécuriser** cette « *petite application* », tant du côté **Front** que du côté **Back**.

Je vous expose ci-dessous les écrans que vous devrez réaliser. Ça vous aidera dans la progression de votre réalisation :

The image shows two side-by-side wireframe screenshots of a web application interface. The left screenshot is titled "Connexion MVC" in pink. It features three input fields labeled "Utilisateur :", "Email :", and "Mot de passe :". Below these fields are two buttons: "Valider" and "Annuler". A link "Créer un compte" is visible in the top right corner. The right screenshot is titled "Ajouter un Utilisateur" in pink. It features four input fields labeled "Utilisateur :", "Mot de passe :", "Email :", and "Fonction :". Below these fields are two buttons: "Enregistrer" and "Annuler". A link "Retour à l'accueil" is visible in the bottom left corner.

The image shows a detailed wireframe of the "Ajouter un Utilisateur" form. The title "Ajouter un Utilisateur" is at the top in large pink font. Below it are four input fields with corresponding red validation messages to their right: "Utilisateur : [input] Veuillez entrer un nom d'utilisateur valide.", "Mot de passe : [input] Veuillez entrer un mot de passe valide.", "Email : [input] Veuillez entrer une adresse email valide.", and "Fonction : [input] Veuillez saisir une fonction.". At the bottom of the form are two buttons: "Enregistrer" and "Annuler". A link "Retour à l'accueil" is located at the bottom left.

Du côté **Front**, vous contrôlerez à la saisie la conformité de l'adresse **email**.

Vous contrôlerez aussi que les champs soient bien **tous renseignés**.

L'identifiant de l'utilisateur devra commencer systématiquement par **une majuscule** et les autres caractères devront être en **minuscules**.

Du côté **Back**, vous contrôlerez encore une fois que **tous les champs** soient effectivement bien **renseignés**.

Vous contrôlerez aussi la validité de l'adresse **email** en la confrontant à un modèle **d'expression régulière** adéquat.

Vous contrôlerez aussi que les champs **Utilisateur** et **Fonction** soient conformes à un modèle d'expression régulière. (Chaîne de caractères de **minimum 3** et **maximum 30** caractères, accentués ou non).

Concernant le **Mot de passe**, il devra comporter au moins **6 caractères**, au moins une lettre **majuscule**, au moins une lettre **minuscule**, au moins un caractère spécial : **\$ @ % * + - _ !**, et au moins un **chiffre**.

Tous ces contrôles devront générer l'apparition de **messages d'erreur** au niveau de l'interface utilisateur. (Cf. captures d'écran ci-dessus).

Un **même utilisateur** ne pourra créer **qu'un seul compte** pour une adresse **email donnée**.

Par souci de sécurité, vous devrez implémenter une solution qui permettra de parer l'application contre les attaques **XSS** éventuelles.

Vous devrez aussi **encrypter** le **Mot de passe** avant son insertion en base de données. L'authentification nécessitera donc un **décryptage** de ce mot de passe. (Cf. le livret *Securite_en_PHP*).

Enfin, toutes les **requêtes** sur la Base de données devront utiliser les **requêtes préparées de PDO**, ceci pour éviter les **injections SQL**.

Après son authentification, l'utilisateur arrivera sur la page d'accueil du site qui affichera l'identifiant de l'Utilisateur, sa fonction et son adresse email. Ces données pourront être stockées dans des variables de **session** sécurisées.

15. CRÉATION D'UNE CLASSE POUR PRÉSENTER UNE TABLE MYSQL

Les programmeurs PHP écrivent continuellement les mêmes lignes pour obtenir des résultats finalement très proches. Le scénario est si bien connu que l'on gagne à les placer dans une classe plutôt que dans un simple fichier **include**. Cette classe aide à opérer la distinction entre le domaine de la présentation HTML et le domaine de la manipulation des bases de données.

Nous vous proposons dans ce TP de réaliser une telle classe pour représenter une table **MySQL** et ses enregistrements au travers de pages PHP. La classe contient des fonctions indépendantes de la présentation telles que la suppression, l'exploration détaillée, la modification et l'ajout d'enregistrements.

Créez sous **MySQL** une base **Guide** et une table **Restaurant** contenant les champs suivants :

Champ	Type
id	Entier calculé automatiquement, clé primaire.
nom	texte
adresse	texte
prix	double
commentaires	texte
note	double
visite	Texte (Ou Date)

Insérez quelques enregistrements.

1 – Concevez une page PHP pour présenter les données sous forme tabulaire.

2 – Concevez une classe **MyTable** dans un fichier **bases.php** pour initialiser la connexion dans le constructeur et pour assurer la gestion des erreurs éventuelles (Utilisez **PDO**), et une méthode **reqSelection(\$requete)** qui permet d'exécuter une requête SQL de Sélection passée en paramètre sans toutefois afficher les données.

3 – Ajoutez une méthode **rendre_html()** à la classe pour afficher les données sous forme tabulaire. Prévoyez aussi de faire afficher les en-têtes des colonnes.

4 – Modifiez la méthode **rendre_html()** pour insérer un lien vers une **URL** paramétrée pour chaque enregistrement (par exemple **detail.php ?id=8**). Par exemple création d'un lien hypertexte sur le **Nom** du restaurant.

5 – Ajoutez une méthode **detail_html(\$id)** à la classe **MyTable** permettant de générer l'affichage du détail dans la page **detail.php**. Faire en sorte que cette méthode génère aussi un affichage permettant de réaliser la modification des données.

6 – Ajouter une méthode de suppression **suppression_enreg(\$id)** à la classe **MyTable** permettant de supprimer un enregistrement. Il faudra mettre en place un message de confirmation de la suppression avant de la rendre effective (Contrôle de confirmation JavaScript avant Suppression).

7 – Ajoutez une méthode de modification **modification_enreg(\$id)** à la classe **MyTable** permettant de modifier un enregistrement. La requête Update devra être préparée pour sécuriser l'application des injections de SQL.

8 – Finalement, ajoutez une méthode permettant l'affichage d'un formulaire de saisie d'un nouveau restaurant **ajout_html()**. Ajoutez une méthode, **ajout_enreg()**, permettant d'ajouter un nouvel enregistrement à la Table Restaurant.

Cette action viendra conclure le cycle **CRUD** au complet à travers une méthode de développement purement orientée objet avec **PHP/MySQL**. Cette approche reste

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

purement pédagogique car dans les faits nous allons arranger le code pour obtenir un découpage plus adapté qui séparera la couche présentation des données. Ainsi le rendu HTML ne sera plus obtenu à travers l'exposition d'une méthode de la classe.

Captures d'Ecran de la liste des Restos, du détail et de la modification d'un Resto.

WebAppli Les Restos

Ajout d'un Resto

ID	Nom	Adresse	Prix €	Commentaires	Note	Visite	Suppression	Modification
2	La Marinade	54, chemin des Amarantes 44000 Nantes	28	Excellents poissons. Service convenable. Cave bien fournie	14	18/05/2008	<button>Supprimer</button>	<button>Modifier</button>
3	Chez Zalbier	30, Boulevard Gouvion Saint Cyr 75017 Paris	40	Spécialités algéroises de premier choix. Pâtisseries délicates, tajines envoûtantes. Le décor participe au dépaysement. Evasion garantie	15	30/01/2019	<button>Supprimer</button>	<button>Modifier</button>
4	En Cuisine	25, rue des Ardoises 19300 Brive	55	Spécialités Corrèziennes remises au goût du jour. Produits de qualité. Excellente prestation	18	28/06/2016	<button>Supprimer</button>	<button>Modifier</button>

WebAppli Les Restos

La Marinade

Adresse :	54, chemin des Amarantes 44000 Nantes
Prix :	28
Commentaires :	Excellents poissons. Service convenable. Cave bien fournie
Note :	14
Date de la visite :	18/05/2008

[Retour à la liste](#)

WebAppli Les Restos

Nom :	<input type="text" value="La Marinade"/>
Adresse :	<input type="text" value="54, chemin des Amarantes 44000 Nantes"/>
Prix :	<input type="text" value="28"/>
Commentaires :	<input type="text" value="Excellents poissons. Service convenable. Cave bien fournie"/>
Note :	<input type="text" value="14"/>
Date de la visite :	<input type="text" value="18/05/2008"/>
<input type="button" value="Valider"/> <input type="button" value="Annuler"/>	

[Retour à la liste](#)

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

16. CLASSE, HÉRITAGE ET CLONAGE

Créez une classe nommée **Form** représentant un formulaire HTML.

Le constructeur doit créer le code d'en-tête du formulaire en utilisant les éléments `<form>` et `<fieldset>`.

Une méthode **setText()** doit permettre d'ajouter une zone de texte.

Une méthode **setSubmit()** doit permettre d'ajouter un bouton d'envoi.

Une méthode **setReset()** doit permettre d'ajouter un bouton reset.

Les paramètres de ces méthodes doivent correspondre aux attributs des éléments HTML correspondants.

La méthode **getForm()** doit retourner tout le code HTML de création du formulaire.

Créer enfin une **méthode statique afficherCommentaire()** dans la classe **Form** qui renverra le message suivant « *Ce que vous pouvez faire avec des Classes !!!* »

Créez des objets **Form**, et ajoutez-y deux zones de texte et un bouton d'envoi et un bouton reset. Testez l'affichage obtenu.

Faite un appel à la méthode statique **afficherCommentaire()** de la classe **Form**.

Créez une sous-classe nommée **Form2** en dérivant la classe **Form**.

Cette nouvelle classe doit permettre de créer des formulaires ayant **en plus** des **boutons radio** et des **cases à cocher**.

Elle doit donc avoir les méthodes supplémentaires qui correspondent à ces créations.

Créez des objets, et testez le résultat.

Faite un appel à la méthode statique **afficherCommentaire()** en sollicitant cette fois la classe **Form2**. Que remarquez-vous ?

Créez un objet à partir de la classe **Form2**, puis créez-en un **clone**.

Modifiez certaines caractéristiques de cet objet, et affichez les deux formulaires obtenus.

Création de formulaires par Classe

Ce que vous pouvez faire avec des Classes !!!

Accès au site

Votre nom :

Votre code :

Envoyer

Annuler

Ce que vous pouvez faire avec des Classes !!!

Coordonnées et sports préférés

Votre nom :

Votre code :

Homme ☒ Femme ☐

Tennis ☐

Equitation ☐

Football ☐

Envoyer

Annuler

17. FORMULAIRE DE CONTACT EN PHP-POO-AJAX-MYSQL ET ENVOI DE MAIL

1.1 – Soumission classique d'un formulaire

Le but de ce TP est de vous permettre de gérer un **formulaire de contact** en utilisant la **POO**. De plus vous devrez gérer l'envoi d'un **email** en utilisant ici le composant **sendmail**.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

Formulaire de Contact - PHP-POO

Nom

Prénom

Email

Message

Annuler Envoyer

Pour ce faire, vous allez devoir prévoir un script **contact.class.php** dans lequel sera implémentée une **classe Contact** qui possèdera les **propriétés** adaptées à notre problématique.

Vous devrez gérer les **contrôles de validation** du **formulaire** avec affichage de **message d'erreur** le cas échéant. Et lorsque **l'insertion** des données dans la base et que le **mail** seront **envoyés** vous devrez afficher un message pour en informer l'utilisateur.

De plus, cette classe permettra **d'insérer** les données du formulaire dans une **table MySQL** que vous devrez créer. Pour cela, vous devrez prévoir une **première méthode** qui fera ce travail.

Cette **classe** permettra aussi de vous envoyer un **mail** avec les informations de contact issues du formulaire. Vous devrez donc prévoir une **deuxième méthode** qui permettra de réaliser cet envoi.

Pour l'envoi d'un email, vous pourrez utiliser le composant **sendmail** pour PHP, vous devrez étudier sa mise en place et vous devrez vous renseigner sur la façon de le faire fonctionner depuis votre **serveur local**, en utilisant les services de votre messagerie par exemple.

Pour cela vous devrez modifier les scripts **sendmail.ini** du composant et **php.ini** de votre serveur **Apache**.

<https://stackoverflow.com/questions/15965376/how-to-configure-xampp-to-send-mail-from-localhost/18185233#18185233>

La solution alternative à celle-ci sera d'utiliser le composant **PHPMailer** qui permet l'envoi de mail depuis une application Web en locale sans nécessiter

l'utilisation et le paramétrage d'un serveur SMTP.

<https://github.com/PHPMailer/PHPMailer>



1.2 – Soumission d'un formulaire par requête AJAX.

Dans cette deuxième partie, vous allez devoir réutiliser le travail effectué ci-dessus mais cette fois en l'adaptant dans le cadre d'une **soumission** par **requête AJAX** du **formulaire**.

Vous effectuerez les **contrôles** qui vous semblent nécessaires côté **Front** et vous **sécuriserez** aussi côté **Back** les données **avant** leur **insertion** en **base de Données**.

Vous devrez prévoir une récupération des **messages d'erreur** venant du **Back** pour un **affichage cumulé** en **Front**. (Cf. capture d'écran suivante). Pour cette version vous n'aurez pas besoin d'implémenter la partie relative à l'envoi d'un email.

Traitement d'un Formulaire de Contact en AJAX - PHP

- Vous devez saisir un Nom
- Vous devez saisir un Email
- Vous devez saisir un Message

Nom

Prénom

Email

Message

Traitement d'un Formulaire de Contact en AJAX - PHP

Données insérées avec succès !!!

Nom

Prénom

Email

Message

Bonus : Dans une **ultime** version, vous pourrez mettre en place un système de récupération des **messages d'erreur** issues du **Back** pour leur affichage en **Front** fidèle à ce que vous avez déjà fait dans la première partie de ce TP. (Messages d'erreur en rouge positionnées à droite des **labels** concernés (**Nom**, **Email**, **Message**)).

Traitement d'un Formulaire de Contact en AJAX - PHP

Nom Vous devez saisir un Nom

Prénom

Email Vous devez saisir un Email

Message Vous devez saisir un Message

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

18. UPLOADER UNE IMAGE VERS LE SERVEUR

Avec ce petit TP, je vous propose de mettre en place un **système d'édition** d'un **article** de **Blog** pour lequel l'utilisateur pourra lier une **image** afin d'enrichir son contenu éditorial.

Pour commencer, vous devrez prévoir la création d'une Base de données **annonce** dans laquelle vous devrez créer une seule table **annonces** composée des champs (**id_annonce** 'Int(11) Auto-Increment', **titre** 'Varchar(150)', **message** 'Text', **nom_image** 'Varchar(150)').

Vous devrez prévoir un système de contrôle des extensions des fichiers avant de les *uploader* vers le serveur. Les images possibles seront exclusivement de type : '.png', '.gif', '.jpg', '.jpeg', sinon un message d'erreur devra avertir l'utilisateur des extensions possibles avant l'insertion.

Pour l'interface utilisateur les captures d'écran suivantes vous renseigneront mieux qu'un long discours ...

<h3>Upload d'une image vers le Serveur</h3> <p>Titre de l'annonce</p> <input type="text"/> <p>Message de l'annonce</p> <div></div> <p>Choisir une image</p> <div>Choisir un fichier Aucun fichier choisi</div> <p>Ajouter</p>	<p>Upload effectué avec succès !</p> <h3>Upload d'une image vers le Serveur</h3> <p>Titre de l'annonce</p> <input type="text" value="Il était une fois un petit Pingouin ..."/> <p>Message de l'annonce</p> <div></div> <p>Choisir une image</p> <div>Choisir un fichier Aucun fichier choisi</div> <p>Ajouter Rendu sur le Front</p>
---	--

Rendu de la page générée lorsque l'utilisateur cliquera sur le lien hypertexte : [Rendu sur le Front](#) :

Rendu sur le Front

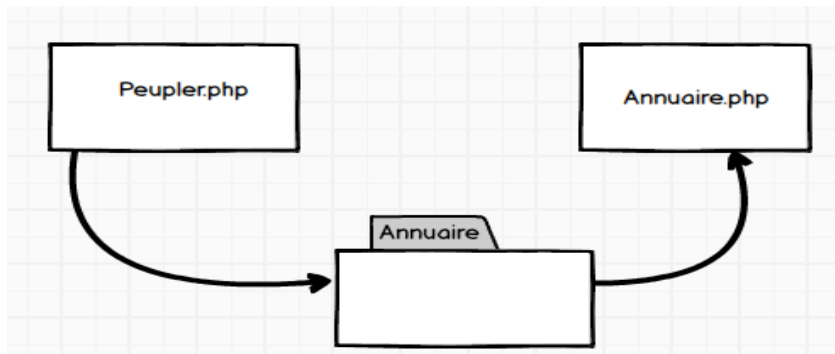
Il était une fois un petit Pingouin ...



C'est l'histoire d'un petit pingouin perdu sur la banquise ...

19. PRÉSENTATION PAGINÉE D'UNE TABLE MYSQL

Il est souvent nécessaire de fractionner la lecture et l'affichage d'enregistrements issus d'une base de données, les performances et la lisibilité s'améliorent alors nettement. La construction d'un dispositif permettant ce fractionnement est l'objet de ce TP.



1 – Créez une base **Annuaire** à l'aide de PhpMyAdmin ou d'un script sql, ainsi qu'une table **Personne** contenant les champs **ID** (auto_increment), **NOM** (varchar(80)), **PRENOM** (varchar(80)), **TELEPHONE** (varchar(10)) et **EMAIL**(varchar(80)).

2 – Créez un script **peupler.php**, non nécessairement sous la forme d'une page web, chargé d'alimenter la table. Pour cela, préparez 3 chaînes de caractères, l'une contenant des prénoms séparés par des virgules, une autre avec des noms de famille et la dernière avec des noms de domaine (gmail.com, orange.fr, yahoo.com ...). Utilisez le découpage (**explode**) pour générer des tableaux.

Utilisez ensuite une fonction de génération automatique de coordonnées, associant un prénom, un nom, un numéro de téléphone à 10 chiffres et une adresse email complète

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

(a.nom@gmail.com). Chaque entrée générée est insérée dans la base. Générez au moins une centaine de lignes.

3 – Construisez une page **annuaire.php** chargée de présenter l'intégralité des lignes. Cherchez une solution très simple, très concise.

4 – Testez la vitesse d'exécution et proposez des améliorations. Inutile d'afficher l'heure au dixième de seconde près.

5 – Utilisez des paramètres sur la chaîne d'interrogation (query string) pour contrôler le nombre d'enregistrements présentés ainsi que l'enregistrement de départ. Vous pourrez présenter 10 enregistrements à chaque fois.

6 – Estimez la différence d'exécution. Il est toujours inutile d'afficher l'heure.

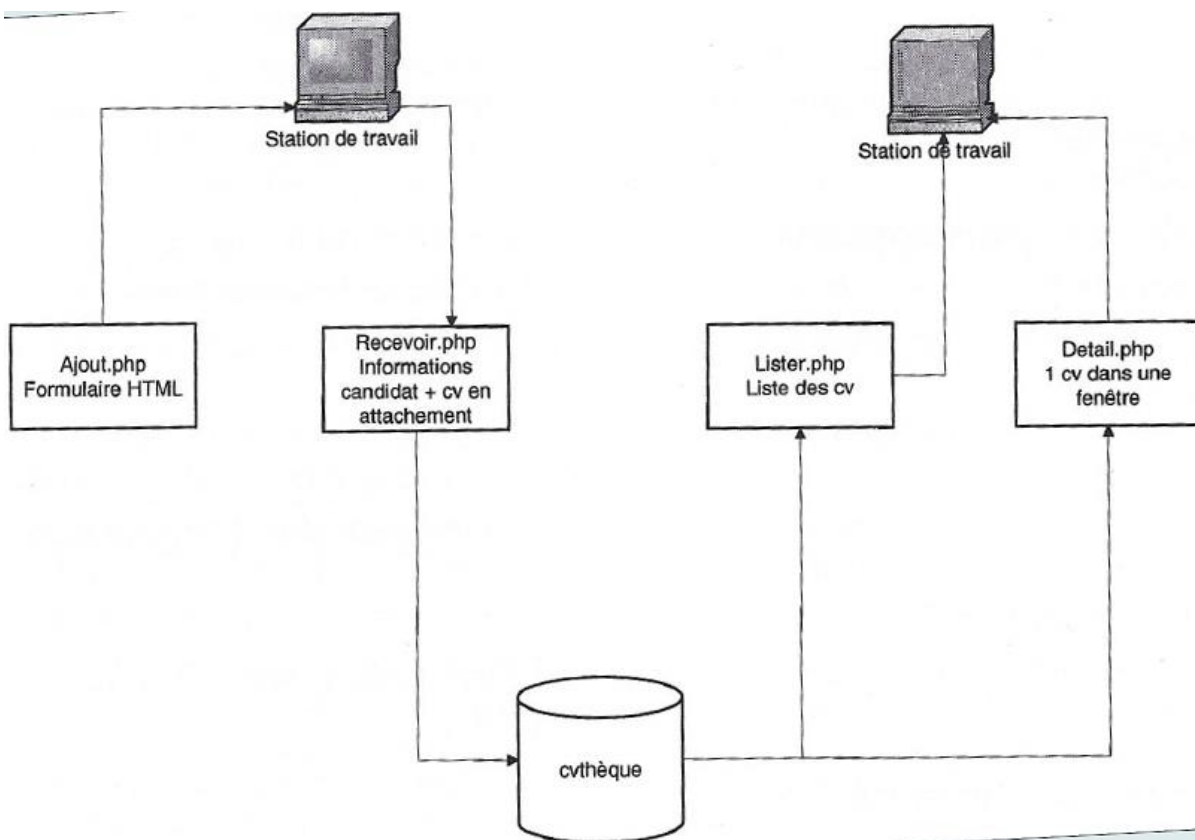


Annuaire			
10 20 30 40 50 60 70 80 90 Suivant			
Hoche	Maxime	0767643706	m.hoche@orange.fr
Nicaud	Cherry	0935917835	c.nicaud@orange.fr
Delacre	Paul	0699894442	p.delacre@hotmail.com
Lescure	Pierre	0701055388	p.lescore@yahoo.com
Durand	Marguerite	0587949098	m.durand@wanadoo.fr
Restoueix	Florian	0773994491	f.restoueix@orange.fr
Charlemagne	Simone	0957280329	s.charlemagne@hotmail.com
Dupont	Francis	0654956762	f.dupont@gmail.com
Chemin	Fred	0535652633	f.chemin@hotmail.com
Delacre	Simone	0832967083	s.delacre@wanadoo.fr

20. ATTACHEMENT DE DONNÉES PAR FORMULAIRE ET STOCKAGE DANS UNE BASE MySQL

PHP est employé dans quantité d'applications pour lesquelles on a besoin de recueillir des fichiers attachés à des formulaires : récupération de CV, messagerie électronique sur le web. Toutefois, la technique de récupération de l'attachement diffère de ce qu'on a l'habitude de déployer pour recevoir des champs de formulaires.

Il est choisi dans ce TP d'étudier un système pour recueillir des CV avec **fichier en attachement** stockés dans une base **MySQL**. Une interface permet ensuite de découvrir les données enregistrées.



1 – Créez une base de données MySQL, **cvtheque**. Créez aussi une table **Personne** avec les champs suivants :

Champ	Type	Valeur
ID	Identifiant	Auto_incrément
NOM	Texte	
EMAIL	Texte	

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

EMPLOI	Enumération	CDD, CDI, Stage
CV	Longblob	
CV_TYPE	Texte	
NOM_FICHER_CV	Texte	
JOUR	Date	

2 – Créez une page **ajout.php**, contenant un formulaire postant sur **recevoir.php**. Le formulaire contient les champs **nom**, **email**, **emploi** (sous la forme de boutons radios), ainsi qu'un champ de type téléchargement de fichier, **cv**. (Les CV seront au format PDF).

3 – Créez la page **recevoir.php**, affichant les paramètres du fichier reçu.

4 – Récupérez le fichier dans une chaîne et affichez-le temporairement.

5 – Modifiez **recevoir.php** pour insérer l'enregistrement. Il faut également calculer la date du jour.

6 – Créez la page **lister.php** pour afficher la liste des personnes ayant envoyé leur CV.

7 – Pour chaque enregistrement, ajoutez un lien vers **detail.php** pour découvrir un CV. L'activation du lien doit déclencher l'ouverture d'une nouvelle fenêtre.

8 – Créez **detail.php**.

9 – Ajoutez une page **menu.php** qui sera insérée en en-tête.

Indications :

2 – L'astuce consiste à prévoir un schéma d'encodage pour le formulaire. Généralement, on utilise **enctype= «multipart/form-data »**.

3 – L'environnement PHP crée un fichier temporaire dont le nom est donné par la variable **\$cv**. D'autres variables sont créées, **\$cv_name**, **\$cv_type**, **\$cv_size**.

4 – Il faut utiliser la fonction **file_get_contents()** pour lire le fichier temporaire. Inutile de sauver le fichier, son contenu sera transféré dans la BDD.

5 – Pour calculer la date, on peut utiliser **time()** et **date()** en choisissant un format qui respecte le schéma supporté par MySQL (année/mois/jour).

8 – Dans **detail.php**, il est inutile de prévoir des balises html. Il vaut mieux définir un en-tête http **Content-Type** à partir du type de fichier contenu dans le **BLOB**. On aura évidemment pris la précaution de sauvegarder cette valeur.

Ajouter Consulter

Ajouter un CV

Nom	<input type="text"/>
Email	<input type="text"/>
Emploi recherché	<input type="radio"/> CDI <input type="radio"/> CDD <input type="radio"/> Stage
CV (Format PDF uniquement)	<input type="text" value="Choisir un fichier"/> Aucun fichier choisi
<input type="button" value="Envoyer"/> <input type="button" value="Annuler"/>	

Ajouter Consulter

Liste des Candidats

MASSET	a.masset173@laposte.net	Stage	CV (pdf)	<input type="button" value="Supprimer"/>
DELACRE	flo.del27@outlook.com	Stage	CV (pdf)	<input type="button" value="Supprimer"/>
NICAUD	nicaud@gmail.com	Stage	CV (pdf)	<input type="button" value="Supprimer"/>

21. AUTHENTIFICATION, SESSION SÉCURISÉE, AJAX ET PROCÉDURES STOCKÉES

L'objectif de ce TP est de mettre en œuvre d'une part la **sécurité** sur les **variables** de **session** rencontrée lors de votre lecture attentive du document *Securite_en_PHP.pdf* et d'autre part la consommation de **Procédures Stockées** sur le SGBD **MySQL**.

Pour commencer, vous allez devoir rajouter à votre base de données **Magasin**, 2 Tables supplémentaires.

Une Table **Utilisateur** qui stockera les **login** et **mot de passe** des utilisateurs de l'application.

Une deuxième table **Role** qui fixera le rôle de l'utilisateur sur l'application. Il y aura 3 rôles prédéfinis possibles (**Admin**, **Editeur**, **Lecteur**).

Vous pourrez créer votre propre jeu d'essai ;-)

Vous devrez créer une « *classe technique* » **Database PHP** qui centralisera toutes les méthodes nécessaires à la **connexion/déconnexion** à la BDD **Magasin**.

Puis toutes les méthodes permettant de **requêter** sur cette BDD, ces **méthodes** seront ici toutes **statiques** ;-)

La **page d'authentification** sera très classique :



**Connexion au
Magasin**

Login utilisateur

Entrer le login d'utilisateur

Mot de passe

Entrer le mot de passe

LOGIN

À la validation, vous contrôlerez que l'utilisateur est bien inséré dans la table **Utilisateur**. Vous effectuerez les contrôles de saisie usuels. Vous mettrez en place un **système de ticket** afin de protéger la session utilisateur.

Vous devrez initialiser une **variable de session** qui stockera le rôle de l'utilisateur, ainsi **l'espace membre** s'adaptera aux droits de **l'utilisateur authentifié**.

La première page accessible après la phase d'authentification sera celle qui présentera la liste des articles.

Si l'utilisateur est un simple **lecteur**, cette liste devra être conforme à :

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

Liste des Articles

ID Article	Désignation	Prix Unitaire (€)	Catégorie	Quantité Vendue
CA300	Canon EOS 3000V zoom 28/80	329.00	photo	1
CAS07	Cassette DV60 par 5	26.90	divers	10
CP100	Camescope Panasonic SV-AV 100	1490.00	video	2
CS330	Caméscope Sony DCR-PC330	1629.00	video	4
DEL30	Portable Dell X300	1715.00	informatique	2
DVD75	DVD vierge par 3	17.50	divers	12
HP497	PC Bureau HP497 écran TFT	1500.00	informatique	2
NIK55	Nikon F55+zoom 28/80	269.00	photo	1
NIK80	Nikon F80	479.00	photo	5
SAX15	Portable Samsung X15 XVM	1999.00	informatique	8
SOXMP	PC Portable Sony Z1-XMP	2399.00	informatique	4

Si l'utilisateur est un **éditeur** ou un **administrateur** cette page devra être conforme à :

Liste des Articles

[Ajouter un Article](#)

ID Article	Désignation	Prix Unitaire (€)	Catégorie	Quantité Vendue	Modifier	Supprimer
CA300	Canon EOS 3000V zoom 28/80	329.00	photo	1	Modification	Suppression
CAS07	Cassette DV60 par 5	26.90	divers	10	Modification	Suppression
CP100	Camescope Panasonic SV-AV 100	1490.00	video	2	Modification	Suppression
CS330	Caméscope Sony DCR-PC330	1629.00	video	4	Modification	Suppression
DEL30	Portable Dell X300	1715.00	informatique	2	Modification	Suppression
DVD75	DVD vierge par 3	17.50	divers	12	Modification	Suppression
HP497	PC Bureau HP497 écran TFT	1500.00	informatique	2	Modification	Suppression
NIK55	Nikon F55+zoom 28/80	269.00	photo	1	Modification	Suppression
NIK80	Nikon F80	479.00	photo	5	Modification	Suppression
SAX15	Portable Samsung X15 XVM	1999.00	informatique	8	Modification	Suppression
SOXMP	PC Portable Sony Z1-XMP	2399.00	informatique	4	Modification	Suppression

Dans les deux cas, la requête **SQL** sollicitée sera une **procédure stockée** que vous aurez implémentée dans votre **SGBD MySQL** depuis votre interface **PHPMyAdmin**.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

La **Quantité Vendue** pour un **article** donné sera obtenue en tenant compte de toutes les **commandes** passées.

Lorsque l'utilisateur *survolera* le lien hypertexte correspondant à la **catégorie** de l'article, une **requête AJAX** sera envoyée sur le serveur et une **info Bulle** apparaîtra en retour sur l'interface utilisateur pour afficher le *Gain total* obtenu sur les ventes de tous les articles appartenant à la **catégorie** mise en jeu.

Liste des Articles						
Ajouter un Article						
ID Article	Désignation	Prix Unitaire (€)	Catégorie	Quantité Vendue	Modifier	Supprimer
CA300	Canon EOS 3000V zoom 28/80	329.00	photo	10	Modification	Suppression
CAS07	Cassette DV60 par 5	26.90	divers		Modification	Suppression
CP100	Camescope Panasonic SV-AV 100	1490.00	video	2	Modification	Suppression
CS330	Caméscope Sony DCR-PC330	1629.00	video	4	Modification	Suppression
DEL30	Portable Dell X300	1715.00	informatique	2	Modification	Suppression
DVD75	DVD vierge par 3	17.50	divers	12	Modification	Suppression
HP497	PC Bureau HP497 écran TFT	1500.00	informatique	2	Modification	Suppression
NIK55	Nikon F55+zoom 28/80	269.00	photo	1	Modification	Suppression
NIK80	Nikon F80	479.00	photo	5	Modification	Suppression
SAX15	Portable Samsung X15 XVM	1999.00	informatique	8	Modification	Suppression
SOXMP	PC Portable Sony Z1-XMP	2399.00	informatique	4	Modification	Suppression

Pour obtenir ces informations vous devrez prévoir une *nouvelle procédure stockée* avec 2 paramètres (un paramètre **IN** qui correspondra à la **catégorie** et un paramètre **OUT** qui stockera le **prix total** de **tous les articles** vendus dans cette **catégorie**).

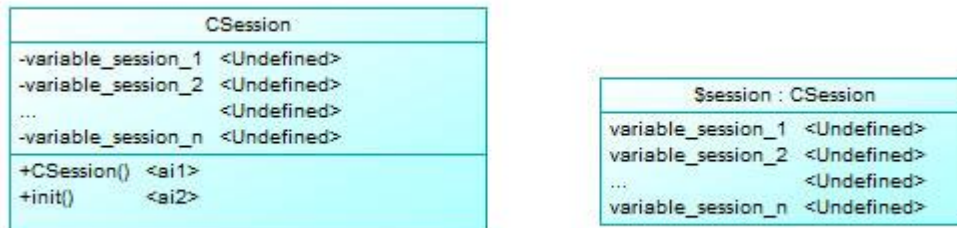
Je ne vous demande pas ici d'implémenter les parties relatives à l'**Ajout**, la **Modification** et la **Suppression** d'un **Article**.

Mais si vous voulez obtenir un **système CRUD** complet sur *la gestion des Articles* de ce magasin, libre à vous d'emprunter ce chemin ...

Bon courage ;-)

22. CRÉATION D'UN COMPOSANT DE SESSION

Le mécanisme des **sessions**, c'est-à-dire de mémorisation de variables entre deux transactions **http** s'est stabilisé depuis la version de PHP4. Toutefois, les programmeurs ont parfois du mal à séparer dans leurs scripts les **variables de session** des **variables globales**. L'idée de ce TP est d'utiliser un composant appelé session, qui stockera l'ensemble des variables mémorisées. On retrouve alors le modèle de Microsoft ASP.



- 1 – Créez un programme **session.php** qui définit une classe **CSession**.
- 2 – Ajoutez une méthode **init()** pour vérifier si une instance de **CSession** est enregistrée en session PHP. Dans la négative, elle est créée et enregistrée sous le nom **session**.
- 3 – En dehors de la classe **CSession** (mais dans le fichier **session.php**), ajoutez l'instanciation de la classe **CSession** et l'appel de la méthode **init()** pour cette instance.
- 4 – Créez une page **test1_sess.php**, dotée d'un formulaire regroupant un champ texte **message** et un bouton de validation. Le formulaire est auto référant.
- 5 – Dans ce script de traitement du formulaire, incluez le fichier **session.php**.
- 6 – En procédant par affectation, ajoutez un membre à l'objet **\$session** pour sauvegarder le message saisi par l'utilisateur.
- 7 – Testez la page et recherchez le fichier temporaire qui contient les variables de session de l'utilisateur. Ouvrez ce fichier dans un éditeur de texte. (Fichier visible dans le répertoire **/tmp** d'Apache).
- 8 – Ajoutez un lien à la page **test1_sess.php** pour charger une autre page, **test2_sess.php**.
- 9 – Créez la page **test2_sess.php**, commencez un script pour récupérer une instance de composant.
- 10 – Extrayez le message et affichez-le.

Indications :

2 – La méthode **init()** concentre les lignes délicates du programme. Utilisez **session_start()**, **\$_SESSION** et **empty()** pour parvenir à vos fins. N'oubliez pas de définir la variable **\$session** comme étant globale.

Mise en pratique – Découverte de PHP

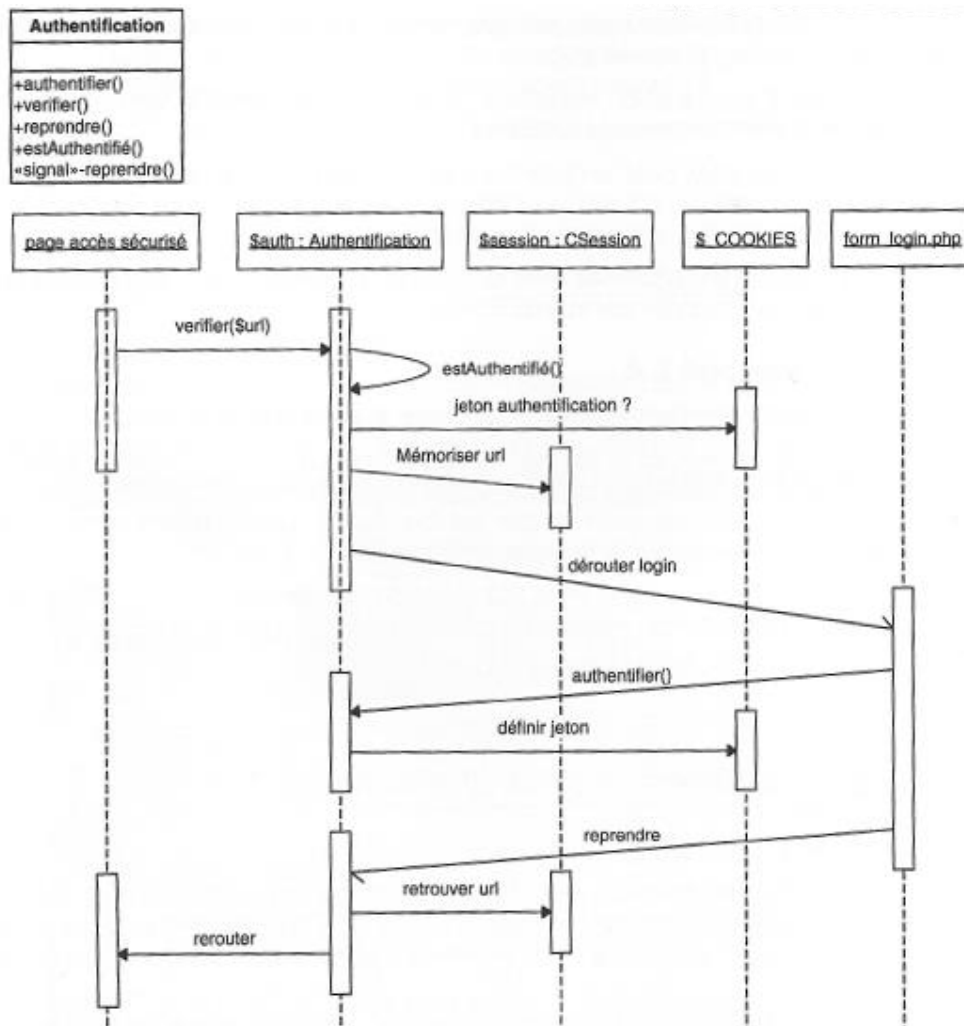
Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

3 – Il faut choisir un nom de circonstance pour votre instance, **\$composant** est un bon choix. Placez les deux lignes en dessous de la définition de la classe.

23. SESSION ET AUTHENTIFICATION D'UTILISATEURS

Nous proposons dans ce TP une solution toute applicative d'authentification. L'idée est d'authentifier un utilisateur à l'aide d'informations de connexions vérifiées dans une BDD. Lorsque l'utilisateur est authentifié, il reçoit un jeton sous la forme d'un cookie. Les pages qui ont besoin d'autorisations spéciales pour fonctionner feront appel au système d'authentification pour vérifier la présence de ce jeton et l'utilisateur sera temporairement dérouter vers un formulaire de connexion si le jeton n'est pas disponible.

Une simple page sert de test pour ce TP, mais ce système pourra être repris dans des applications plus ambitieuses.



- 1 – Dans la base Annuaire du TP sur la Pagination (13 – Présentation paginée d'une Table MySQL), créez une table **Utilisateur** formée des colonnes **idu**, **login** et **mdp**. Ajoutez quelques enregistrements d'essai.
- 2 – Créez une classe **Authentification** dans la page **auth.php**.
- 3 – Ajoutez une méthode **estAuthentifie()** qui détermine la présence d'un cookie nommé « **PHP_AUTH** ». Cette méthode renvoie **true** si le cookie a comme valeur le **numéro de session** concaténé avec la chaîne « **ABC** ».
- 4 – Ajoutez une méthode **authentifier()** qui admet comme paramètres un nom d'utilisateur et un mot de passe. Cette méthode provoque l'écriture d'un cookie **session** qui sera lu par **estAuthentifie** si le couple **nom/mot de passe** est vérifié dans la base **Annuaire**. La méthode renvoie **true** ou **false**, selon les cas.
- 5 – Ecrivez une méthode **verifier(\$url)** qui teste l'authentification d'un utilisateur et le déroute vers **form_login.php** si nécessaire, en mémorisant en variable de session l'URL demandée. Cette méthode renvoie **true** ou **false**, selon le cas.
- 6 – Ecrivez également une méthode **reprendre** qui affiche l'URL demandée si l'authentification a réussi.
- 7 – Dans le fichier **auth.php**, créez une instance nommée **auth** de la classe **Authentification**.
- 8 – Mettez au point **form_login.php** présentant un formulaire et traitant les crédits apportés par l'utilisateur. Lors de la publication du formulaire, la page doit utiliser l'instance **auth** pour appeler **authentifier()** puis **reprendre()**.
- 9 – Elaborez une page **test_auth.php** qui se serve de la classe **Authentification**.

24. JSON ET AUTO COMPLÉTION

Dans ce TP nous proposons de mettre en place un système **d'Auto-complétion** de champs de formulaire.

Dans un premier temps il faudra générer un fichier JSON qui reprend les champs de la Table **personne** de la BDD **Annuaire**.

- 1 – Implémentez un script PHP **generatejson.php** qui devra générer un fichier JSON : **Annuaire.json** à partir de la table **personne** de la BDD **Annuaire** déjà rencontrée.
 - 2 – Créez un formulaire constitué de trois champs textes (**id**, **nom**, **prénom**, **téléphone** et **email**), le champ **id** sera désactivé.
- L'Auto-complétion devra se faire sur le champ **nom** de la personne et lorsqu'une personne sera sélectionnée les champs **id**, **prénom**, **téléphone** et **email** seront automatiquement renseignés avec les bonnes données.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

Pour mener à bien cette mission, vous pourrez mettre en place un système d'interrogation des données du fichier JSON (**Annuaire.json**) reposant sur **AJAX** (en utilisant par exemple **JQuery**).

Id	
Nom	Com
Prénom	Comin Nathalie
Téléphone	Comin Fred
Email	Comin Marguerite
	Comin Fred
	Comin Raoul
	Comin Florian
	Comin Nicolas
	Comin Raoul
	Comin Mila
	Comin Simone

25. COMPOSANT JQUERY DATATABLES, PHP-MVC

Avec ce TP, vous allez avoir l'occasion de prendre en main le composant **Datables** de **JQuery** et de le **customiser** pour obtenir une présentation et des fonctionnalités spécifiques.

Ce composant permet de créer facilement des **listes organisées** en **tableau paginé**, chaque entête des colonnes du tableau permet de disposer d'une fonction de **tri** sur les valeurs qu'elles contiennent.

Laissant à l'utilisateur le choix d'afficher son tableau par 5, 10, 20, 25 ou plus de lignes d'enregistrements. Un champs de recherche est immédiatement disponible.

Vous devez prendre en main ce composant et étudier sa mise en place en consultant sa documentation officielle : <https://datatables.net/>

Le composant propose une version par **CDN**, je vous conseille de l'utiliser même si ça rajoute une **dépendance** à votre **projet Web**.

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

DATATABLES

Afficher 5 éléments

Rechercher :

Nom	Prenom	Téléphone	Email	Modifier	Supprimer
Adam	Marie	0566554408	m.adam@wanadoo.fr	Modification	Suppression
Adam	Raoul	0801624600	r.adam@hotmail.com	Modification	Suppression
Adam	Paul	0766004963	p.adam@hotmail.com	Modification	Suppression
Adam	Paul	0569912643	p.adam@wanadoo.fr	Modification	Suppression
Adam	Cherry	0982157723	c.adam@yahoo.com	Modification	Suppression

Affichage de l'élément 1 à 5 sur 96 éléments

Précédent 1 2 3 4 5 ... 20 Suivant

[Ajouter une Personne](#)

Pour arriver à ce résultat, il vous faudra tout d'abord gérer la **langue** pour obtenir la langue **française**.

Puis vous devrez **customiser** l'apparence de la vue. Vous pourrez vous aider pour cela de <https://datatables.net/manual/styling/theme-creator>

Vous pourrez utiliser la Base **annuaire** déjà rencontrée dans un précédent TP, le script de création de cette BDD vous est fourni avec ce TP (**annuaire.sql**).

Vous pourrez ainsi tester vos scripts dans des conditions acceptables.

À partir de là, vous pourrez implémenter le système **CRUD complet** dans le cadre d'une **architecture MVC**.

26. ENVOI DE MAIL AVEC LE COMPOSANT PHPMAILER

Le but de ce TP est d'utiliser le composant **PHPMailer** afin de pouvoir envoyer des emails depuis une application tierce.

Ce composant est très robuste et complet, il est donc préférable de l'utiliser plutôt que la fonction **mail()** native de PHP. Il permet l'envoi de messages avec des fichiers attachés, au contenu HTML, il permet d'utiliser le service SSL, ...

Dans un premier temps, vous devrez récupérer la dernière version de **PHPMailer** à l'adresse :

<https://github.com/PHPMailer/PHPMailer/releases>

Après cela, vous devrez mettre en place ce composant dans votre application.

Vous devrez créer un script (**sendmail.php**) faisant référence à ce composant et vous devrez implémenter la partie du script qui vous permettra d'envoyer un message (email) à une adresse mail de votre choix. Vous devrez faire en sorte que l'on puisse attacher un fichier en attachement de ce message.

Toute la difficulté de ce TP réside dans la mise en place d'un composant et de son paramétrage par rapport à un service SMTP pour le rendre opérationnel même sur votre serveur local.

27. ATELIER TWIG

Twig est un générateur de Template en **PHP**, il peut être utilisé avec un Framework (c'est le générateur de Template attitré de **Symfony**) mais aussi dans un projet web en PHP indépendant de tout Framework.

Ce TP sera l'occasion de découvrir l'installation de Twig dans un projet web PHP sans avoir recours à un Framework.

Pour commencer, vous devez installer **Composer** sur votre ordinateur. **Composer** est un **gestionnaire de dépendances** pour les projets PHP. C'est l'alter-ego de **Npm** pour les projets Front qui reposent sur **Node.js**.

Si vous travaillez avec un OS **Windows**, il existe un installateur qui vous permettra d'installer Composer **globalement** sur votre machine. (RDV sur <https://getcomposer.org/>)

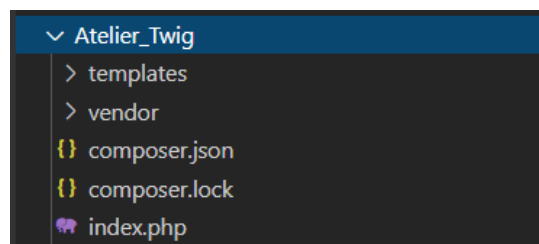
Lorsque Composer sera installé sur votre ordinateur. Vous devrez créer par exemple un répertoire de projet sur votre serveur web : **Atelier_Twig**

En suite, vous devrez ouvrir une fenêtre **Terminale** au niveau de ce dossier et vous exécuterez la ligne de commande :

```
> composer require "twig/twig:^3.0"
```

Ce qui aura pour effet d'installer Twig pour votre projet. A la fin du processus d'installation, un dossier **vendor** aura été créé ainsi qu'un fichier **composer.json** et aussi un fichier **composer.lock** qui permet de verrouiller toutes les versions des dépendances installées par notre projet.

Vous allez devoir créer un script : **index.php** ainsi qu'un dossier **templates** :



Le script *index.php* devra contenir le code suivant :

```
require_once 'vendor/autoload.php';

$loader = new \Twig\Loader\FilesystemLoader('templates');
$twig = new \Twig\Environment($loader, [
    'cache' => false,
]);

$prenom = "Sacha";
$nom = "restoueix";
$fonction = "Formateur DWM";

echo $twig->render('home.html.twig', [
    'prenom' => $prenom,
    'nom' => $nom,
    'fonction' => $fonction
]);
```

```
?>
```

Dans le dossier **templates** vous allez créer le fichier *home.html.twig* qui comportera le code suivant :

```
<h1>Bienvenue {{prenom}} {{nom | upper}}, vous êtes {{fonction}} </h1>

Date du jour : {{ "now"|date('d/m/Y') }}
```

Lorsque vous chargerez le script *index.php* dans votre Navigateur vous obtiendrez ce rendu :

Bienvenue Sacha RESTOUEIX, vous êtes Formateur DWWM

Date du jour : 31/01/2021

Voilà, vous avez réalisé votre première *mini application* qui repose sur le générateur de vue **Twig**.

Twig offre un véritable langage de programmation avec des **boucles**, des **conditions**, des **filtres** ...

Il permet d'alléger le code de façon significative par rapport au code PHP qui est très verbeux. Au final c'est bien du code PHP qui est généré mais il est optimisé.

De façon native, **Twig** offre une protection des données en échappant par défaut toutes les valeurs des variables qui comportent des caractères spéciaux. Ce qui permet de lutter efficacement contre les **injections XSS**.

Si vous ne voulez pas que Twig échappe vos variables qui contiennent du code HTML, il faut lui préciser en utilisant le filtre **raw** :

```
{{ ma_variable_html|raw }}
```

Si vous voulez utiliser un commentaire dans votre script.twig vous devez utiliser la syntaxe suivante :

```
{# Ceci est un commentaire Twig #}
```

Lorsque vous voulez **déclarer des variables** et les **concaténer** dans Twig :

```
{% set message = 'Salut' %}
{% set nom = 'Sacha' %}

{{ message ~ nom|lower }}    {# Salut sacha #}
```

Les Structures de contrôle avec Twig :

Pour utiliser une structure de contrôle **If ...elseif ... else ... endif** avec Twig :

Si vous avez déclaré au préalable une variable **\$membre** comme il suit au niveau de votre contrôleur :

```
$membre = ["age" => 52];
```

Du côté Template vous pourrez utiliser la **structure de contrôle** suivante en utilisant la syntaxe **Twig** :

```
{% if membre.age < 12 %}  
    Il faut avoir au moins 12 ans pour ce film.  
{% elseif membre.age < 18 %}  
    OK bon film.  
{% else %}  
    Un peu vieux pour voir ce film non ?  
{% endif %}
```

Pour utiliser la boucle **For** avec la syntaxe **Twig** :

Dans votre contrôleur définissez la liste :

```
$liste = ["Tigre", "Lion", "Panthère", "Ocelot", "Jaguar"];  
  
echo $twig->render('home.html.twig', [  
    'felins' => $liste  
]);
```

Dans votre Template utilisez la syntaxe Twig pour itérer sur cette collection :

```
<ul>  
    {% for felin in felins %}  
        <li>{{ felin }}</li>  
    {% else %}  
        <li>Pas de félin trouvé.</li>  
    {% endfor %}  
</ul>
```

La boucle **For** définit une variable **loop** qui possède différents **attributs** qui peuvent être très utiles :

```
{{ loop.index }}
```

Le numéro de l'itération courante (en commençant par 1).

```
{{ loop.length }}
```

Le nombre total d'itérations dans la boucle.

...

Ainsi en utilisant notre tableau de félins on peut définir un rendu côté Vue en utilisant le test suivant :

```
{% for felin in felins %}
  <span class="{% if loop.index is even %}pair{% else %}impair{% endif %}">
    {{ felin }}
  </span>
{% endfor %}
```

On peut aussi utiliser le test

```
{% if var is defined %} ... {% endif %}
```

Qui permet d'alléger la syntaxe PHP équivalente :

```
<?php
if (isset($var))
{
    ...
}
?>
```

La Notion d'Héritage avec Twig :

Parlons maintenant de **l'héritage** à la mode **Twig** :

Dans beaucoup de langages nous avons la possibilité **d'inclure** du code afin d'éviter sa répétition. Notre projet peu à tout moment devenir un projet conséquent avec le temps et si les bonnes pratiques ne sont pas mises en œuvre dès le départ il sera extrêmement difficile de faire demi-tour.

Si on doit changer une valeur à un endroit dans notre code qui est recopié partout, nous allons devoir la changer autant de fois qu'il y a de copié/collé.

Il faut avouer que c'est plutôt dommage quand on sait qu'avec **Twig** on peut facilement centraliser notre code en un fichier puis **inclure** ce fichier chaque fois que bon nous semble.

L'héritage avec **Twig** repose sur le même concept que celui du développement classique, il n'y a rien de bien nouveau : nous avons établi une charte graphique pour notre site et nous voulons l'utiliser partout dans notre application.

À la façon du développement, dans lequel nous avons une **classe** qui sera étendue par sa/ses filles, avec **Twig** nous allons avoir un Template **père** dans lequel il y aura des

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »

"**blocks**" à remplir et il sera étendu et rempli par son/ses **fils** pour former une page complète.

La différence réside dans le fait que là où *l'include* en PHP ne permet que la modification à l'endroit où il est écrit, le fils d'un Template Twig peut remplir plusieurs "blocks" (plusieurs trous).

Le Template **père** s'appelle en général le "*layout*". On aura par exemple le fichier *layout.html.twig* :

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>{% block title %}Notre site{% endblock %}</title>
  </head>

  <body>
    {% block body %}
    {% endblock %}
  </body>
</html>
```

Voici un exemple de Template **fils** nommé accueil qui comblerait le block body :

```
{% extends "layout.html.twig" %}

{% block title %}{{ parent() }} - Accueil{% endblock %}

{% block body %}
  Voici le body de notre page.
{% endblock %}
```

Dans le **fils** nous pouvons voir le mot "**extends**" qui permet de faire référence au Template **père**.

Nous pouvons aussi voir le "**block title**" et le "**block body**" dans le Template **père** que nous retrouvons dans le **fils**.

Comme expliqué, les blocks dans le **fils** servent à remplir le Template **père**, le block title **fils** prend alors la place du block title dans le Template **père**.

Nous pouvons donc en déduire que le nom d'un block se définit par {% block son_nom %} et il est délimité par ce même block et {% endblock %}.

Dans le **fils** nous retrouvons cette même notation avec un {% block son_nom %} et {% endblock %}.

Twig offre aussi la possibilité de mettre en **cache** les modèles compilés sur le système. Ainsi, la compilation est faite une seule fois et cela améliore la performance de notre site.

Pour cela il vous suffit de créer un répertoire **repCache** à la racine de votre projet et d'indiquer à Twig que vous voulez activer la mise en cache sur votre projet.

Pour activer le système de cache de Twig, nous devons renseigner le nom du répertoire de cache dans notre fichier « *index.php* ».

```
...
$twig = new Twig_Environment($loader, [
    'cache' => 'repCache',
]);
...
```

Vous verrez qu'après avoir actualisé votre **Navigateur**, un dossier est créé automatiquement dans le dossier de mise cache **repCache** et que dans ce dossier nous avons un fichier « *php* » qui contient le contenu de notre page.

Afin de bien prendre en main toutes ces notions, je vous propose de réaliser ce petit exercice.

Pour que vos modifications soient systématiquement prises en compte pendant la phase de développement, je vous conseille, avant de démarrer, de supprimer la mise en cache offerte par **Twig** en revenant à :

```
'cache' => false,
```

Vous allez devoir implémenter un petit site web qui présentera les différents langages du Web, sans être exhaustif.

Une barre de navigation qui sera commune à toutes les vues offrira une navigation aisée sur le site. Les différentes rubriques [**HTML**, **CSS**, **JavaScript**, **JQuery**] seront contenues dans un **tableau** que vous devrez parcourir pour créer cette barre de navigation.

Vous devrez jouer sur **l'héritage de Template** qu'offre **Twig** pour implémenter cette partie de la solution.

Vous devrez intégrer à votre solution une feuille de style générale, vous intégrerez aussi **Bootstrap** ou un thème Bootstrap de votre choix, enfin vous devrez intégrer la Bibliothèque **JQuery** à cette application.

Le rendu général aura cette forme :



Chaque section devra présenter le langage qui lui est affecté.

Ce TP vous a permis de vous familiariser avec la syntaxe de **Twig** et avec **Composer** ce qui sera un acquis important avant d'aborder l'étude du célèbre Framework PHP **Symfony**.

CREDITS

ŒUVRE COLLECTIVE DE l'AFPA

Sous le pilotage de la DIIP et du centre d'ingénierie sectoriel Tertiaire-Services

Équipe de conception (IF, formateur, mediatiseur)

Formateur : Alexandre RESTOUEIX

Date de mise à jour : 08/08/2021

Reproduction interdite

Article L 122-4 du code de la propriété intellectuelle.

« Toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droits ou ayants cause est illicite. Il en est de même pour la traduction, l'adaptation ou la reproduction par un art ou un procédé quelconque. »

Mise en pratique – Découverte de PHP

Afpa © 2021 – Section Tertiaire Informatique – Filière « Étude et développement »