Devoir sur Table - (1h30)

Développement Web - PHP

NOM : PRÉNOM :

1 Questions de Cours

1.1 Expliquez le rôle de chacun des composants suivants : navigateur, base de données, serveur web. Décrivez les spécificités de LAMP/MAMP/WAMP par rapport à l'architecture générale théorique.

— Navigateur : logiciel permettant au client d'effectuer des requêtes à un serveur web... c'est à dire, il permet de naviguer sur le web.
— Serveur Web : logiciel permettant de répondre aux requêtes d'un navigateur web en lui envoyant les ressources qu'il demande, ou éventuellement un message d'erreur
— Base de Données : un conteneur de données organisées (n'est pas un SGBD)

LAMP/MAMP/WAMP : Les 3 suites logicielles permettent d'installer sur un seul système (Linux, Mac, Windows) un serveur web et un SGBD.

1.2 Expliquez des points de vue utilisateur et développeur les différences entre un site web statique et dynamique. Citer quelques langages nécessaires pour faire un site web dynamique.

Du point de vue utilisateur, un site web statique ne changera jamais lorsque l'on rafraichit la page/répète une même requête. À l'inverse, un site dynamique peut éventuellement changer pour une même requête (exemple : stocks d'objets évoluent).

Du point de vue développeur, le site statique est écrit avec un langage de présentation : on écrit exactement ce qui doit être affiché sans aucune logique métier. Dans un site dynamique, on écrit une logique métier/des règles qui permettent de faire évoluer différentes valeurs affichées.

Site dynamique : PHP, Javascript, Java

1.3 Citez les 3 versions majeures du protocole HTTP. Expliquez très succinctement la différence entre HTTP et HTTPS.

HTTP 1.0, HTTP 1.1, HTTP/2

HTTP transmet toutes les informations en clair, tandis que HTTPS chiffre les communications entre le client web et le serveur web. On peut vérifier l'authenticiter du serveur web avec les certificats fournis avec HTTPS. S pour Secure.

1.4 PHP est-il un langage fortement ou faiblement typé ? Expliquez la différence.

PHP est un langage faiblement typé.

Les langages fortement typés imposent que les types soient rigoureusement les mêmes que ceux utilisés dans les opérations/fonctions. Une comparaison entre un nombre et une chaîne de carcatères ne peut pas fonctionner tel quel.

Les langages faiblements types ne vérifient pas les types avant d'effectuer des opérations, ils utilisent le plus simple/commun aux valeurs données en paramètres. Par exemple, comparer une chaîne de caractères et un nombre sera effectué en comparant des chaînes de caractères uniquement.

1.5 Expliquez comment PHP est utilisé dans le cadre d'un serveur web.

PHP est un pré-processeur, c'est à dire qu'il va lire un fichier et modifier certaines valeurs, pour générer un fichier de sortie qui sera lu/utilisé par un autre logiciel ou service.

Le serveur web reçoit des requêtes et y répond en renvoyant les ressources demandées. PHP est pafois appelé par le serveur web pour traiter les ressources avant de les envoyer aux clients. Le serveur web fournit le nom de la ressource et des variables à PHP, PHP fait les traitements, puis PHP renvoie le résultat au serveur web (qui transmet au client la réponse).

1.6 Expliquez la différence de fonctionnement au niveau du serveur web entre les méthodes GET et POST.

Le navigateur, pour envoyer des données au serveur web, peut utiliser 2 méthodes : GET et POST.

GET : l'URL vers la ressource est étendue d'un **?** qui sépare les ressources visées des données transmises. Les variables sont séparées par des **&**. (format général : http ://domain.com/ressource ?var=42& i=4). (On ne devrait pas envoyer plus de 256 octets par cette méthode.) Le serveur web reçoit donc une URL contenant les données supplémentaires. Le serveur web enregistre dans les logs/journaux les URL demandées, il ne faut donc jamais envoyer de données sensibles par cette méthode. (les admins du serveur web peuvent lire les données)

POST : Le navigateur envoie les données dans le corps de la réponse, elles ne sont donc pas enregistrées dans les logs/journaux du serveur web. (Il n'y a pas de limite pour la taille des données envoyées en POST.)

1.7 Expliquez la différence d'utilisation en PHP des méthodes GET et POST. Indiquez les limites de chaque méthode, s'il y en a.

En PHP, les variables envoyées avec GET et POST se trouvent dans des tableaux associatifs. Un formulaire envoie les données selon la méthode indiquée, le serveur web les récupère et les insère dans des tableaux associatifs pour PHP. Les variables GET et POST sont des superglobales.

```
$_GET['variable']
$_POST['variable']
```

1.8 Expliquez la différence entre les fonctions *strip_tags* et *htmlspecialchars*.

- htmlspecialchars (chaine, flags) : Conversion des caractères spéciaux en entités HTML. Remplace par exemple & (ET commercial) en &amp ;
- strip_tags (chaine, allowableTags) : Supprime les balises HTML et PHP d'une chaîne. Les commentaires HTML et PHP sont également supprimés

```
$str = "This is some <b> bold </b> text.";

echo htmlspecialchars($str);
// affichera : This is some &lt;b&gt; bold &lt;/b&gt; text.
```

```
$text = `
```

```
<p>Paragraph.</p><!-- Comment --><a href="#fragment">Other text</a>
';

// Autorise <p> et <a>
echo strip_tags($text, '<p><a>');
// affichera :
// <p>Paragraph.</p> <a href="#fragment">Other text</a>
```

1.9 Expliquez les différentes parties des URL suivantes, et ce qu'un serveur web standard comprendra :

1. **`https://univ-paris.fr`** : protocole : http (web), domaine : univ-paris.fr, ressource : aucune, donc la racine /
2. **`http://www2.website.jp:3864/homepages/portal/`** : protocole : http (web), domaine : www2.website.jp, port : 3864, ressource : /homepages/portal/
3. **`http://www.dailynews.co.uk/article/2864`** : protocole : http (web), domaine : www.dailynews.co ressource : /article/2864
4. **`http://www.monsite.fr/files-storage/data.video/index.php`** : protocole : http (web), domaine : www.monsite.fr, ressource : files-storage/data.video/index.php (fichier : index.php)
5. **`ftp://server01.datastore.bigcorp.com/contents/fabrice/`** : protocole : ftp, domaine : server01.datastore.bigcorp.com, ressource : /contents/fabrice/ (dossier)
6. **`rtmp://live.streamcorp.xyz:1337/flux-video/film/4242.01`** : protocole : rtmp (streaming), live.streamcorp.xyz, ressource : /flux-video/film/4242.01

1.10 Remplir le tableau avec la valeur booléenne de retour que chaque fonction renverrai pour chaque valeur de variable en entrée.

Paramètre	**`isset()`**	**`empty()`**	**`is_null()`**
`null`	false	true	true
[**`unset($var)`**]	false	true	error/true
`42`	true	false	false
`0`	true	true	false
`""`	true	true	false
`" "` [un espace]	true	false	false

1.11 Remplir le tableau avec les codes HTTP ou leur description.

Code HTTP	Description
200	OK
302 307	Temporary Redirect
301 308	Permanent Redirect
403	Le client n'a pas le droit d'accéder à la ressource
404	URL/Ressource demandée n'existe pas
500	Internal Server Error
502	Bad Gateway

1.12 Remplir le tableau avec les familles de codes HTTP et leur description.

Code HTTP	Description
100	Informations sur l'état de la connection
200	Réponses positives
300	Redirections
400	Erreurs côté client/dans la requête envoyée
500	Erreurs côté serveur

1.13 Expliquez les différents niveaux de visibilité des variables. Donnez un exemple succinct pour chaque niveau.

— Variables locales : n'existent que dans le scope local à la fonction
— Variables globales : elles existent dans l'ensemble des fonctions appelées par le script principal
— Variables superglobales : elles existent dans l'ensemble de PHP

1.14 Expliquer les différences entre les *sessions* et les *cookies*.

— Sessions : variables stockées côté serveur. Le numéro de session est envoyé à l'utilisateur qui le rappelle à chaque requête vers le serveur web. (Très utile pour stocker les gros objets sans avoir à les renvoyer au serveur web à chaque requête.)
— Cookies : variables stockées côté client. Les variables et valeurs sont renvoyées constament au serveur web. (Utile pour identifier l'utilisateur. Déconseillé pour stocker les gros objets, car il faut les renvoyer à chaque requête)

1.15 Dans quel cas vaut-il mieux utiliser des *sessions* ? Dans quel cas vaut-il mieux utiliser des *cookies* ?

— Sessions : Très utile pour stocker les gros objets sans avoir à les renvoyer au serveur web à chaque requête.
— Cookies : Utile pour identifier l'utilisateur. Déconseillé pour stocker les gros objets, car il faut les renvoyer à chaque requête.

1.16 Expliquez ce qu'est un SGBD par rapport à une BDD. Indiquez le nom du standard permettant d'interroger un SGBD.

— BDD/DB : Base De Données/DataBase, l'ensemble des données organisées dans un fichier (ou en mémoire) selon un schéma
— SGBD/DBMS : Système de Gestion de Base de Données/DataBase Management System, le logiciel/programme permettant d'interroger et modifier des bases de données
SQL (Standard Query Language) est le langage standard de requête de base de données.

1.17 Que signifie l'acronyme ACID ? Quelles sont les 2 étapes qui peuvent alternativement se produire à la fin d'une transaction ? Donnez un exemple de transaction (du point de vue métier, puis du point de vue technique).

ACID : Atomicité, Cohérence, Isolation, Durabilité
À la fin d'une transaction, il se produit soit un COMMIT, soit un ROLLBACK.
Exemple de transaction : un achat en ligne (métier). L'achat se découpe en plusieurs étapes techniques : sélection des objets selon les stocks disponibles, création d'une commande, retrait des objets des stocks.

1.18 Qu'est-ce que :

1. **`L'Atomicité`** : transaction se déroule intégralement ou pas du tout
2. **`La Cohérence`** : la base de données doit passer d'un état cohérent à un autre état cohérent
3. **`L'Isolation`** : une transaction ne peut pas dépendre d'une autre transaction
4. **`La Durabilité`** : les modifications de la base de données à la fin de chaque transaction sont sauvegardées et doivent résister aux crashs/une reprise dans le dernier état cohérent en cas de crash doit être possible

1.19 Qu'est-ce qu'un moteur de stockage dans une base de données ? Citer 3 exemples et leurs caractéristiques.

Le moteur de stockage d'une base de données est le le logiciel/programme/code qui permet d'organiser les données dans la base. Chaque moteur a des caractéristiques spécifiques (écriture ou lecture rapide, ...).

- MyISAM : lecture rapide, pas de clé étrangère
- InnoDB : gère ACID et les clés étrangères
- Memory : aucune écriture sur disque, uniquement en mémoire... ultra rapide mais ne sauvegarde rien en cas de crash

2 Développement

2.1 Ce code devrait afficher le contenu du tableau, mais il ne fonctionne pas. Corrigez les erreurs de syntaxe pour que le tableau soit correctement rempli et affiché.

```
1  <html>
2  <body>
3  <?php
4
5  my_tab = array();
6  my_tab[1] = Fabrice;
7  my_tab[2] = Ali;
8  my_tab[3] = Floriane;
9
10 foreach ( i  in  my_tab )
11 {
12    echo('i \n');
13 }
14
15 ?>
16 </body>
17 </html>
```

CORRECTION :

```
<html>
<body>
<?php

$my_tab = array();       // <== $
$my_tab[1] = "Fabrice";    // <== $ " "
$my_tab[2] = "Ali";      // <== $ " "
$my_tab[3] = "Floriane";   // <== $ " "

foreach ( $my_tab as $i ) // <== $my_tab as $i
{
   echo("$i \n");      // <== " $ "
}
```

```
?>
</body>
</html>
```

2.2 Ce code devrait compter le nombre de lignes contenant la valeur "Floriane", mais il ne fonctionne pas. Corrigez les erreurs pour que le compte soit bon.

```
1  <html>
2  <body>
3  <?php
4
5  // Fait une requete SQL et l'ecrit dans $result
6  require(functions.php)
7
8  $results = my_fun();
9
10 for ( $i = 1; $i <= length($results); $i-- )
11 {
12   if ($results[$i] == "Floriane")
13     $count++;
14 }
15
16 echo "$count";
17 ?>
18 </body>
19 </html>
```

CORRECTION :

```
<html>
<body>
<?php

// Contient la fonction "my_fun" qui fait une requete SQL et
//  renvoie son resultat
require("functions.php"); // <== " " ;

$results = my_fun();
$count = 0;          // <== $count = 0;

for ($i = 0; $i < count($results); $i++) // <== 0 < count/sizeof $i++
{
        if ($results[$i] == "Floriane")
                $count++;
}

echo "$count";
```

```
?>
</body>
</html>
```

2.3 Cet ancien code écrit avec l'extension obsolète *MySQL* doit être mis à jour. Écrivez la nouvelle version en utilisant PDO ou MySQLi.

```php
<?php
function MyConnection($host, $db)
{
  $dbms = mysql_connect($host, "login", "pass");
  if ($dbms)
  {
      mysql_select_db($db $dbms);
  }
  return ($dbms);
}

function CloseMyConnection($sgbd)
{
  mysql_close($sgbd);
}

function MakeQuery($sql, $db)
{
  $result = FALSE;
  if ($db)
  {
    $result = mysql_query($sql, $db);
  }
  return ($result);
}

function IterResSql($results)
{
  $tab = array();
  $i = 0;
  if ($results)
  {
    while ($row = mysql_fetch_assoc($results))
    {
      $tab[$i] = array();
      $tab[$i]['prenom'] = $row['prenom'];
      $tab[$i]['nom'] = $row['nom'];
      $tab[$i]['adresse'] = $row['addresse'];
      $tab[$i]['age'] = $row['age'];
      $i = $i + 1;
    }
  }
  return ($tab);
```

```
44 }
45
46 $co = MyConnection("localhost", "MaBDD");
47 $requete = "SELECT * FROM clients";
48 $res = MakeQuery($requete, $co);
49 $tab = IterResSql($res);
50 CloseMyConnection($co);
51 ?>
```