Internet et Outils (IO2)
Contrôle terminal de session 1
12 mai 2020

Le barème est donné à titre indicatif.

Lisez bien tout le sujet avant de vous mettre au travail. Certaines questions dépendent d'autres, mais vous pouvez les aborder même sans avoir terminé les questions précédentes.

Dans les réponses aux questions suivantes, veillez à utiliser les fonctions PHP pour factoriser votre code, pouvoir le réutiliser et le rendre lisible.

Le but est de réaliser un mini-site pour l'annotation d'articles. Un utilisateur, en se connectant sur le site, a accès à un certain nombre d'articles. Il/Elle devra alors choisir l'article à lire. Le site permet à l'utilisateur d'écrire des commentaires sur l'article lu, qu'il pourra ensuite accéder, relire ou supprimer. Les bases de données n'étant pas au programme de ce contrôle, les commentaires seront stockés dans la session de l'utilisateur. Ils seront donc disponibles le temps de la session ; quand l'utilisateur se déconnecte tout les commentaires seront perdus.

Toutes les pages du site doivent être rattachées à la même feuille de style `monStyle.css` (qu'il n'est pas nécessaire d'écrire), et doivent comporter, en pied de page, l'adresse email du Webmaster.

PARTIE 1

Question 1 — *2 points* Ecrire une fonction `page_login($login)` qui affiche un formulaire demandant login et mot de passe, le formulaire est pré-rempli avec le login passé en paramètre. La page affiche en plus un message d'erreur ("login et mot de passe non reconnus") si ce paramètre a une valeur. Le formulaire de login sera traité par la page `home.php`, à écrire dans le prochain exercice.

Question 2 — *5 points* Ecrire une page `home.php`. Si la page est demandée avec des données venant du formulaire de login, elle doit vérifier que login et mdp sont valides (pour simplifier, le seul login valide est "user0" avec mdp "user0"). Si c'est le cas la page stocke le login dans une variable de session. Ensuite l'accès à la page est protégé de la façon suivante. Si la variable de session contenant le login est indéfinie le script affiche la page de login (par la fonction définie à l'exercice précédent), possiblement pré-rempli avec la tentative de login en cours, ensuite le script termine. Si en revanche la variable de session contient le login alors la page est affichée.

Pensez à définir des fonctions pour toutes ces vérifications car **toutes les page du site devront commencer par faire la même vérification** : elles doivent vérifier que la variable de session contient bien un login et, si ce n'est pas le cas, elles doivent terminer en affichant le formulaire de login.

Le contenu de la page home est le suivant : Un titre affiche "Choisissez un article", ensuite une liste déroulante permet de choisir entre quatre articles à lire (la liste affiche les titres des quatre articles, choisissez les quatre titres vous mêmes, soyez inventifs). Chaque article a aussi un identifiant de `0` à `3` qui n'est pas affiché. Ensuite un bouton "Lire" envoie l'article choisi (par la méthode `GET`) à la page `lecture.php` que vous devez écrire au prochain exercice. Veillez à insérer suffisamment d'information dans la liste déroulante pour que la page `lecture.php` reçoive l'identifiant de l'article à lire.

Bonus. En fin de page ajouter un lien *Déconnexion*. Écrire le script `deconnexion.php` pointé par ce lien, qui s'occupe de détruire la session courante, et rediriger vers la page home.

Question 3 — *5 points* Écrire la page `lecture.php`. Le script commence par vérifier qu'un login est bien présent dans la variable de session correspondante, et termine en affichant la page de login, si ce n'est pas le cas. Si en revanche c'est le cas, la page est affichée de la façon suivante.

D'abord le script initialise un tableau de quatre elements ; à l'indice `i` le tableau contient le texte de l'article d'identifiant `i` (cette étape simule une lecture de la base de données, qui serait plus réaliste) - peu importe les textes, les choisir au hasard.

Ensuite le script récupère l'identifiant de l'article à lire dans une variable `$article`, et affiche le texte correspondant dans une balise `<article>` de la page. Cette balise est suivie d'une liste de commentaires, chacun dans une balise `<p>`. Les commentaires à afficher se trouvent dans la variable de session `$_SESSION[commentaires]`, initialement vide. Cette variable est un tableau à deux dimensions : en position [i][j] la variable contiendra le commentaire numéro `j` sur l'article d'identifiant `i`. la variable de session `$_SESSION[commentaires]` est initialement vide et sera remplie au fur et à mesure par les commentaires de l'utilisateur comme indiqué dans le prochain exercice.

Attention : la page `lecture.php` affiche seulement les commentaires pour l'article courant `$article` (s'il y en a). À la suite des commentaires la page affiche un bouton "Ajouter un commentaire", qui sera traité par le script `commentaire.php` à écrire pour l'exercice suivant. Veillez à envoyer à ce script l'identifiant de l'article `$article`.

PARTIE 2

Question 4 — *4 points* Écrire le script `commentaire.php`. Il commence par vérifier qu'un login est bien présent dans la variable de session correspondante, et termine en affichant la page de login, si ce n'est pas le cas. Si en revanche c'est le cas, il affiche une *text area* dans lequel l'utilisateur pourra écrire son commentaire, suivie d'un bouton "Enregistrer". Ce formulaire est traité par le script `commentaire.php` lui même. N'oubliez pas que ce formulaire doit re-envoyer au script `commentaire.php` également l'identifiant de l'article.

Le script `commentaire.php` reçoit donc en général deux paramètres : l'identifiant de l'article `$article` et possiblement le contenu de la *text area* `$text`. Seulement si `$text` est indéfinie la *text area* sera affichée comme indiqué plus haut. En revanche si `$text` contient déjà une valeur, au lieu d'afficher la *text area*, le script enregistre le nouveau commentaire `$text` parmi les commentaires de l'article `$article`, dans le tableau `$_SESSION[commentaires]` ; ensuite il redirige vers la page `lecture.php` (de cette façon la page `lecture.php` affichera de suite le commentaire qui vient d'être ajouté). N'oubliez pas que la redirection doit également envoyer à la page `lecture.php` l'identifiant `$article` en paramètre.

Question 5 — *4 ponts* Modifiez la page `lecture.php`, qui s'appellera maintenant `lecture1.php`, pour qu'elle affiche, avant chaque commentaire une case à cocher. Un bouton "Supprimer" sera aussi ajouté à coté du bouton "Ajouter un commentaire". Le bouton "Supprimer" doit envoyer la valeur des cases cochées, ainsi que l'identifiant de l'article à la page `lecture1.php` elle même, par la méthode `GET`.

Le script `lecture1.php` reçoit donc en général deux paramètres : l'identifiant de l'article et possiblement les cases cochées. Si la variable correspondante aux cases cochées est définie, le script s'occupe de supprimer les commentaires correspondants dans `$_SESSION[commentaires]` avant d'afficher la page comme à la Question 3 (avec en plus les cases à cocher et le bouton "Supprimer"), sinon elle affiche directement la page.

Se rappeler que sur un tableau php `$tableau` la fonction `unset($tableau[i])` efface la case `i` du tableau, sans changer les indices des autres cases.