DWWM Vernon 2022
Évaluation PHP

Travaux à réaliser du 01/08/2022 au 15/08/2022

Dans le cadre de l'évaluation PHP, vous traiterez l'un des deux sujets au choix :

- Création d'un blog
- Terminer le projet Alibobo.

Travaux à réaliser :

- L'ensemble des travaux sont à fournir au plus tard le 15/08/2022 à 19h00 (date et heure à laquelle vous me fournirez l'URL de votre dépôt)
- L'exercice est **individuel**
- Vous devrez utiliser Git (GitHub ou GitLab) en respectant les règles suivantes
    - utiliser au minimum deux branches (*main* et *develop*)
    - utiliser les branches *features* et les *pull requests*
    - créer un dossier « _documents » dans lequel vous placerez le script permettant l'import de la base de données renseignée et le fichier de la modélisation de votre base de données
    - dans ce dossier, vous devrez également indiquer les accès à votre application (compte utilisateur, compte administrateur, *etc.*)
    - créer un fichier README.md indiquant comment déployer votre application sur un serveur (sans rien oublier)
    - L'évaluation sera faite en clonant les scripts de la branche *main/master* et en faisant un import de votre base de données renseignées.

Code :

- l'application sera codée en PHP 8.×
- idéalement, faites de la programmation orientée objet (vu ce qui est demandé, c'est mieux), je veux au moins une classe
- vous utiliserez MySQL ou MariaDB pour votre base de données
- si vous le jugez nécessaire, vous pouvez utiliser plus d'une base de données
- même si ce n'est pas noté, un peu de CSS est le bienvenu.

Quelques conseils (toujours les mêmes) :

- adopter le principe KISS *(keep it simple stupid)*
- user et abuser du RTFM *(read the fucking manual)*
- si des dizaines de personnes ou plus se cassent la tête pour écrire de la documentation, c'est qu'il y a une bonne raison
- un *mail catcher* pourrait être utile
- Google n'a pas la réponse à tous vos problèmes

- Stackoverflow non plus
- OpenClassrooms encore moins
- faites des *commits* atomiques
- faites des *commits* atomiques avec des messages de commit digne de ce nom (ça n'est pas le canal le plus adapté pour parler des vos états d'âmes ou de vos problèmes existentiels)
- pensez au *.gitignore* avant de faire votre premier *commit*
- ne commencez pas le 14, ça va se voir
- si vous avez une question, demandez à un être humain susceptible de vous répondre
- planifiez votre travail
- dans 95% des cas, la réponse est dans ce que nous avons vu en cours
- ce qui compte, c'est la démarche
- je préfère une moitié de sujet traité comme il faut, que la totalité d'un site qui ne marche pas
- si vous faites un WordPress, je le verrai
- si vous pompez sur un site, je le verrai
- si vous me rendez le même projet à 10, je le verrai.

Sujet 1 - Réalisation d'un blog

Votre client, un célèbre influenceur de la région de Sous-Parsat dans la Creuse, souhaite que vous lui développiez une application lui permettant de publier des articles et autres contenus. Malgré vos explications acharnées pour lui expliquer qu'un WordPress fera très bien l'affaire, ledit client est prêt à vous payer le développement de son propre CMS parce que c'est son projet !

Bien que plus à l'aise dans la communication digitale pour vendre l'eau de ses poissons rouges et poster des *selfies* en débardeur pour ses fans du Club des Cheveux d'Argent de Saint-Avit-le-Pauvre, votre client vous a néanmoins communiqué un cahier des charges fonctionnel.

Fonctionnalités à développer

1. Fonctionnalités du site

Le site a pour vocation de permettre la rédaction de contenus et leur commentaire par des utilisateurs qui auront nécessairement un compte et des niveaux d'accès limitant leurs actions possibles.
Chaque opération liée à des (comptes) utilisateurs devra faire l'objet d'un envoi de mail au compte concerné.
N'importe quel utilisateur peut s'inscrire sur le site et aura par défaut le rôle "utilisateur inscrit".
Seul l'administrateur aura la possibilité de modifier le rôle d'un utilisateur. L'utilisateur en question sera notifié par mail du changement de situation de son compte.
L'administrateur et les rédacteurs devront pouvoir visualiser les notes affectées aux articles (nombre de notes et moyenne des notes).

2. Gestion des rôles

Votre client souhaite avoir la possibilité de gérer les niveaux de compte suivante :

- ***Visiteur***
  Pas de compte en soi, est en capacité de lire les articles publics, mais ne peut ni leur affecter une note, ni écrire un commentaire, ni répondre à un commentaire ;
- ***Utilisateur inscrit***
  Possède un compte, peut mettre une note à un article, rédiger un commentaire et répondre à un commentaire
- ***Modérateur***
  A les mêmes droit que l'utilisateur inscrit, mais peut en plus modifier ou supprimer un commentaire ou une réponse à un commentaire d'un autre utilisateur (mais pas d'un autre modérateur)

- ***Rédacteur***
  A les mêmes droits que le modérateur, mais peut rédiger et publier des articles.
- ***Administrateur***
  Possède la totalité des droits, incluant l'ajout, la suppression, la gestion ou le bannissement temporaire d'un utilisateur.

3. **Interface utilisateur**

   Interface par défaut pour chaque catégorie d'utilisateurs, intégrant les fonctionnalités suivantes :
   - navigation classique (accueil, mentions légales, formulaire de contact, formulaire d'inscription)
   - inscription
   - connexion/déconnexion
   - afficher les derniers articles
   - noter un article (si connecté)
   - afficher les catégories des articles
   - afficher les articles par catégories (un article peut être dans plusieurs catégories).

4. **Interface administrateur**

   Cette interface sera accessible aux modérateurs, rédacteurs et administrateurs. Les fonctionnalités seront à afficher en fonction du rôle de l'utilisateur connecté. Outre les fonctions classiques de CRUD, les fonctionnalités seront à proposer en fonction des droits des utilisateurs.
   - CRUD utilisateurs avec affectation des droits
   - CRUD de contenus en fonction des droits
   - gestion des commentaires.