RAPPORT DE PROJET
Tech Note

Groupe :
BENAIS-HUGOT Charles,
LOUSTAU-CARRERE Guillaume

2015 - 2016

Table des matières

1. Introduction

Dans le cadre du projet d'Architecture et programmation du Web (HLIN607) de Licence 3 Informatique, dont le but est de mettre en place un site web permettant la consultation, le dépôt et le commentaire de technotes (de petits articles techniques), nous avons conçu une base de donnée relationnelle ainsi qu'une architecture web à l'aide des langages informatiques PHP, Javascript, HTML5 & CSS.

2. Base de données

2.1 Diagramme

Pour décrire l'implémentation de la base de données, un MCD (modèle conceptuel des données) a été réalisé qui montre les différentes tables, leurs relations, et les cardinalités associées. Ensuite, pour permettre la mise en place de la base de données proprement dite, un MLD (modèle logique des données) a été réalisé qui décrit les clés étrangères et les tables relationnelles à créer en fonction des cardinalités des associations.

2.1.1 MCD

FIGURE 2.1 – Modèle conceptuel des données

2.1.2 MLD

FIGURE 2.2 – Modèle logique des données

2.2 Table

2.2.1 Role

Il existe trois types de membres : les visiteurs, les membres et les administrateurs. id_role : type int Clé primaire de la table, permet d'identifier chaque rôle. libelle_role : type VARCHAR (255 caractères maximum) Nom du rôle, pour permettre de savoir à quel rôle correspond l'id_role. Contenu de la table :

2.2.2 Membre

Les membres sont ou non enregistrés dans la base de données. Les membres non enregistrés sont reconnus comme Visiteur (id_role = 3) id_membre : type int Cle primaire de la table

permet de différencier chaque membre email : type VARCHAR (200 caractères maximum) Permet au membre de se connecter pseudo : type VARCHAR (30 caractères maximum) Identifiant sous lequel le membre sera vu par les autres utilisateurs mot_de_passe : type VARCHAR (35 caractères maximum) Contient le mot de passe permettant au membre de se connecter, codé grace à la fonction md5 en php. id_role : type int Permet de gérer les droits d'un membre grace à la table rôle_action. date_creation : type datetime Date d'inscription du membre sur le site.

2.2.3 Techquestion

Les technotes et les questions gérant les mêmes attributs, j'ai décidé de les réunir dans une seule table. La table contient donc les technotes et les questions. id_techquestion : Type int Clé primaire de la table. Permet de différencier chaque entrée dans la table. titre : type VARCHAR (100 caractères maximum) Titre du technote ou de la question contenu : Type text Contenu du technote ou de la question . id_type : type int Permet de différencier au sein de la table les technotes des questions. Valeur 1 pour les technotes, 2 pour les questions date_creation : type datetime Permet de garder en mémoire la date de création du technote ou de la question afin de pouvoir effectuer une recherche par date. id_membre : Type int Cle étrangère de la table membre, permet de connaitre l'auteur d'un technote ou d'une question. statut : type int Pour les technotes : Permet de préciser si le technote est publié (valeur 1) ou en cours de rédaction (valeur 0). Pour les questions : Permet d'indiquer l'avanceée de leur résolution (valeur 1 : résolu , valeur 0 : non résolu)

2.2.4 Commentaires

Technotes et questions peuvent faire l'objet de commentaires de la part des Membres et Administrateurs id_commentaire : type int Cle primaire de la table permet de différencier chaque commentaire contenu : type VARCHAR (400 caractères maximum) Contenu du commentaire date_creation : type datetime Date et heure de l'enregistrement du commentaire id_techquestion : type int Cle étrangère de la table techquestion , permet de savoir à quel technote ou question est lié le commentaire. id_membre : type int Cle étrangère de la table membre, permet de connaitre l'auteur du commentaire. id_pere : type int Permet une arborescence des commentaires à un niveau : un commentaire peut être une réponse à un commentaire existant. id_pere permet de connaître le commentaire lié. Sinon il sera égal à 0.

2.2.5 Mot_clé

Technotes et questions sont liés à une liste de mots-clés qui permettra une recherche aisée et rapide. id_motcle : type int Cle primaire de la table , permet de différencier chaque mot-clé. libelle : type VARCHAR(200 caractères maximum) Libellé du mot-clé date_creation : type datetime Permet de connaitre la date de création d'un mot-clé id_pere : type int Un mot-clé peut appartenir à une famille de mots-clés. L'id_pere permet de connaître la famille d'un mot-clé. Le but était de développer une ontologie verticale des mots-clés mais elle n'a pas été implémenté dans la recherche. id_synonime : type int Pour l'ontoglogique horizontale (implementé) valide : type int Permet de préciser la validation d'un mot clé : Valeur 0 : non validé, Valeur 1 : Validé

2.2.6 Mc_tp

Cette table permet de relier les techquestions à leurs mots-clés id_techquestion : Clé primaire, étrangère de la table techquestion id_motcle : Clé primaire, étrangère de la table mot_cle origine : Représente l'origine du mot-clé, choix de l'utilisateur (Valeur 1) ou synonyme de son choix (Valeur 0).

2.2.7 Action

La table décrit les différentes actions sur lesquelles chaque rôle peut intervenir id_action : type int Clé primaire de la table, permet d'identifier chaque action. libelle_action : type VARCHAR (255 caractères maximum) Libellé de l'action, pour permettre de savoir à quelle action correspond l'id_action. Contenu de la table :

2.2.8 Role_action

Cette table décrit les interactions entre rôle et action telles que décrites dans le programme. Pour chaque couple (rôle, action), on décrit les autorisations d'ajout, modification, suppression et consultation id_role Clé primaire étrangère de la table rôle id_action Clé primaire étrangère de la table acton ajouter : type int Valeur 1 permet l'ajout - Valeur 0 interdit l'ajout modifier : type int Valeur 1 permet la modification - Valeur 0 interdit la modification supprimer : type int Valeur 1 permet la suppression - Valeur 0 interdit la suppression consulter : type int Valeur 1 permet la consultation - Valeur 0 interdit la consultation Contenu de la table :

3. Architecture du programme

Le développement Web implique une architecture Client/Serveur. Le cahier des charges imposait l'utilisation de PHP, HTML et Javascript. Après avoir sommairement décrit l'architecture, je vous présenterai l'arborescence des fichiers, ainsi que les choix de conception qui ont permis de respecter le cahier des charges.

3.1 Architecture Client/Serveur

L'architecture Client/Serveur est l'architecture utilisée pour tous les développements Web. Les langages utilisés peuvent être interprétés côté Client ou côté Serveur. Dans notre cas, le PHP est interprété côté serveur, alors que le javascript est interprété côté Client. Le PHP, sigle de HyperText Preprocessor, langage de script côté serveur, est un langage de programmation de page web, libre, gratuit, simple d'utilisation et d'installation. Le javascript, langage de script côté client, est un langage de programmation qui permet d'alléger le travail du serveur. Fonctionnement de l'architecture Client/Serveur : Le client envoie une requête au serveur sous la forme d'une url. Le serveur demande à l'interpréteur PHP de fabriquer la page demandée. L'interpréteur envoie éventuellement, les demandes de requêtes SQL à la base de données MySql. La base de données exécute les requêtes avant de renvoyer les résultats. L'interpréteur PHP prépare alors la page Html qu'il fera parvenir au client.

3.2 Arborescence de Fichiers

3.2.1 Index.php

Le fichier index.php est le seul fichier appelé quel que soit la page demandée. Il joue le rôle de «contrôleur». Le champ «action» permet de déterminer les opérations à effectuer et la page à afficher : Les pages sont donc appelées soit au moyen d'un lien du type «index.php ?action=<action>» (méthode GET), soit au moyen d'un formulaire possédant un champ caché nommé action (méthode POST). A chaque appel de la page index, la valeur de la variable action va déterminer les actions à accomplir.

3.2.2 Config.php

Ce fichier contient les informations de connexion à la base de données et crée une instance de la classe PDO qui permettra de faire tous les appels à la base de données. Il est appelé par index.php.

3.2.3 Fonction.php

Ce fichier définit différentes fonctions qui sont utilisés par le programme. Il est appelé par index.php. La fonction envoyer_mail permet d'envoyer un email. Elle est utilisé lors de l'inscription d'un membre et devrait l'être si un membre oublie son mot de passe. La fonction formaterDate permet de formater une date au format sql au format jour/mois/année.

De même la fonction formaterDateH ajoute l'heure au format précédent. Enfin, la fonction definirDroit permet de definir les droits lorsqu'un membre ou un visiteur fait un premier appel à index.php. Les droits sont alors défini dans les variables de la session et sont conservé toute la durée de la session.

3.2.4 Constante.php

Ce fichier définit quelques constantes et fait appel à la fonction definirDroit dans le cas ou les droits ne sont pas définis dans la session. Il est appelé par index.php, après la vérification des actions de connexion ou déconnexion.

3.2.5 Javascript.js

Ce fichier contient tout le code javascript du programme. Chaque formulaire s'est vu associé une fonction javascript qui permet de vérifier que les champs obligatoires sont correctement remplis. Le fichier est appelé par index.php dans l'en-tête de la page html.

3.2.6 Style.css

Ce fichier contient le code css. Le fichier est appelé par index.php dans l'en-tête de la page html.

3.2.7 Dossier methodes

Il contient les classes du programme : cl_membre.php : elle gère les membres. cl_techquestion.php : elle gère les technotes et les questions. cl_commentaire.php : elle gère les commentaires cl_motcle.php : elle gère les mots-clés. Chacun de ces fichiers est systématiquement appelé par index.php Chaque fichier se compose de deux classes : une classe décrivant l'objet et ses attributs et une classe permettant sa gestion. La classe décrivant l'objet contient : une méthode construct qui prend un dictionnaire en paramètres sous la forme tvaleurs['nom de l'attribut']=valeur de l'attribut. Une methode isNew qui permet de savoir si l'objet existe dans la base de données. Une méthode isValid qui permet d'initialiser les attributs si besoin avant l'enregistrement. Les méthodes set et get pour chaque attribut. La classe permettant de gérer l'objet contient : une méthode construct qui prend pour paramètre l'instance de base de données créé à chaque appel de la page index.php. Une méthode save qui va sauvegarder l'objet dans la base de données en le créant ou le modifiant. Les méthodes add et delete appelés par save. La méthode delete qui supprime l'objet La méthode getOne qui permet de récupérer toutes les données d'un objet Des méthodes spécifiques.

3.2.8 Dossier actions

Il contient les différentes actions permises par le programme. Chaque fichier fait les appels d'ajout, modification ou suppression dans la base de données telles que définit par la variable action. Le fichier index.php fait appel au fichier action associé à l'action demandée.

3.2.9 Dossier affichages

Il contient les différentes vues nécessaires au fonctionnement du programme, formulaires, listes ou détails des informations. Le fichier index.php appelle la vue utile selon l'action demandée.

3.2.10 Dosser ckeditor

Il contient les différents fichiers nécessaires au fonctionnement de ckeditor, un framework qui permet d'intégrer une barre d'outil à un textarea classique. L'utilisateur peut ainsi formater ses données.

3.3 Choix de conception

3.3.1 Utilisation des Sessions

Afin de gérer les droits des utilisateurs, j'ai utilisé les sessions PHP. La page index.php crée une session Visiteur lors du premier appel par un utilisateur. Lorsque l'utilisateur se connecte, la session Visiteur est détruite et une nouvelle session avec les droits associés au membre est créé. La session reste ouverte jusqu'à ce que le membre se déconnecte ou ferme le navigateur.

3.3.2 PDO

L'extension PHP Data Objects (PDO) définit une interface permettant d'accéder à une base de données depuis PHP. Elle nécessite PHP5,1 pour fonctionner. Elle est orientée objet, définie dans une classe PDO. On crée donc une instance d'objet PDO en passant en paramètre le SGBD utilisé, le nom de la base de données, celui de l'utilisateur et son mot de passe. Par exemple : $BDD = new PDO("mysql :host=$BDD_HOST ;dbname=$BDD_NAME ;charset=utf8, $BDD_USER, $BDD_PWD) ;

3.3.3 CKeditor

Je me suis servis de la classe CKeditor afin de satisfaire à la consigne de faire ressortir dans les technotes le code des explications. Pour cela j'ai déclaré a la place des textarea des objets de la classe CKeditor ce qui me permet d'avoir une barre d'outil associé à un textarea, lorsque l'utilisateur saisi du code il a juste a selectionner son code et le mettre en citation. Le css formatera ensuite le code qui sera délimité par des balises blockquote en début et fin de citation, ce qui permet de faire ressortir le code.

3.3.4 Ontologie des mots-clés

J'ai créé une ontologie horizontale de mots-clés en définissant des synonimes pour chaque mot-clé. Lorsqu'un membre saisit un mot-clé pour définir son technote ou sa question, le programme recherche si le mot-clé existe, le crée sinon. Le mot-clé créé a un statut non validé au départ. Lesa administrateurs, grâce à une interface de gestion des mots-clés peuvent afficher les mots-clés (en particulier les non validés qui sont affichés en rouge) et les associer à des synonimes avant de les valider. Lorsqu'un utilisateur fera une recherche, le programme recherchera non seulement le mot-clé saisit, mais aussi tous les mots-clés synonimes. Toute modification d'un mot-clé (ajout ou suppression d'un synonyme), ou suppression de mot-clé est répercutée sur la table mc_tq qui liste les relations entre mot-clé et technquestion. Lors d'une recherche, l'utilisateur peut demander plusieurs mots-clés en les séparant par une virgule. S'il préfixe un mot-clé par +, ce mot-clé devient obligatoire. Par exemple si l'utilisateur saisit : A, +B, +C, D, + E, F, la requête recherchera les technotes ou les questions de la facon suivante : « et (B ou syn B) et (C ou syn C) et (E ou syn E) et (1 ou A ou D ou F ou syn A ou syn B ou syn C) »

4. Conclusion

Plusieurs améliorations peuvent être apportées au site. En particulier : Vérification du champs email lors de la connexion (javascript) Gestion de l'ontologie verticale : un ID père existe dans la table mot-clé. Sa saisie est gérée dans le formulaire des mots-clés, mais il n'est pas utilisé dans les recherches. Création de la focntionnalité de gestion des droits qui permettrait de définir les droits accordés à chaque rôle.

Le développement de cette application m'a permis d'approfondir mes connaissances en PHP, Javascript, HTML, CSS et dans l'utilisation d'une base de données MySQL. Cette première expérience de développement WEB m'a beaucoup intéressée et m'a permis d'aborder de nombreuses notions (sessions, PDO, etc). Les recherches sur la notion d'ontologie m'ont captivé ; Difficiles à appréhender dans un premier temps, j'espère en avoir compris la théorie et su la mettre en partie en pratique.