Cher(e)s étudiant(e)s,

Il est demandé de rendre un rapport écrit par Groupe Projet Tutoré

pour le vendredi 13 janvier dernier délai. Le format du rapport (PDF

ou papier) est à voir avec votre enseignant.

Votre rapport doit contenir à minima :

\* Le sommaire,

\* L’'introduction,

\* La description du projet,

\* Les solutions mise en œuvre,

\* Les problèmes rencontrés et leurs résolutions,

\* Les algorithmes,

\* Une synthèse des comptes rendus des points de rendez-vous avec

l'enseignant et le client,

\* La conclusion,

\* En annexe : un peu de code, copies d’écran, etc …

\* Un glossaire,

\* Les

références bibliographiques.

Consigne pour le rapport :

--------------------------

l'ordre d'idée est de produire un document entre 15 et 20 pages (les

annexes sont en plus), sans trop de copies d'écran (si cela s'avère

nécessaire, il faudra mettre les copies d'écran en annexe et y faire

référence dans le corps du document). Le document sera composé en

Times 11 et s'il est rendu imprimé, ce sera en recto verso. Les codes

sources (des extraits uniquement, choisis judicieusement et commentés)

devront être mis en annexe.

-----------------------------------------------------------------------------

-----------------------------------------------------------------------------

Il est rappelé que la note tiendra compte :

\* De l’'état d’'avancement, du travail, ..., tout au long de votre

projet, qui a été contrôlé lors de chaque point avec votre enseignant

et votre client : 10/20

\* Du rapport écrit : 5/20

\* De l’exposé oral : 5/20

Projet Gestion de QCM

Client : Mme Françoise Gayral

Septembre 2016-Janvier 2017

Par les élèves de DUT Informatique :

Mme Lucie Godefert

Mme Célia Aït Mammar

M. Florian Kotecki

M. Nathan Gatto

M. Charlie Didier

Supervisé par :

Mme Françoise Gayral

Sommaire

* Introduction : qui sommes-nous, choix du projet
* Description du projet : problématique, attentes du client
* Solutions de mise en œuvre : choix des outils de travail, choix des méthodes de travail pour les différentes parties,répartition du travail et temps

->conception (le bla bla sur l’avant-dvpt)

-> développement(page par page avec les outils utilisés au début de chaque partie)

-parler de la base de données ?->N

-parler des choix de design ?->W

-partie questionneur(F,C remplissez ça)

-partie répondeur(my job.)

* Problèmes rencontrés et leurs résolutions : au cas par cas, ajoutez vos pb rencontrés sur votre code
* Les algorithmes : solutions techniques
* Synthèse des comptes rendus des points de rendez-vous avec l'enseignant et le client->à détailler
* Conclusion
* Annexes
* Glossaire
* Références

Introduction

Nous sommes une équipe de cinq étudiants en deuxième année de DUT Informatique à l’Université Paris 13. Dans le cadre de notre troisième semestre, nous avons été amenés à choisir un projet tuteuré parmi ceux proposés par les différents enseignants. Nous avons opté pour le projet proposé par Mme Gayral, la gestion de QCM. En effet, le sujet nous donnait déjà une idée de la conception possible du site, et de la manière de l’implémenter.

De plus, le développement s’appuyant sur des langages de programmation que nous maîtrisons, nous avons estimé ce projet plus accessible que d’autres nécessitant la découverte de langages méconnus.

Description du projet

Notre projet se présente sous la forme d’un logiciel web permettant de créer et de répondre à des QCM. Il est constitué de deux interfaces, prévues pour deux sortes d’utilisateurs différentes: le questionneur qui crée les QCM, et le répondeur qui répond aux QCM créés par les questionneurs. Ainsi un élève répondeur n’a accès qu’à l’interface « répondeur » alors qu’un enseignant questionneur a aussi accès à l’interface « questionneur ».

Les langages à utiliser étaient imposés par le client : notre projet s’articule donc autour de deux interfaces implémentées en html et css pour le développement front-end\*, ainsi qu’en php avec l’extension PDO\* pour le développement back-end\*. Il s’appuie sur une base de données sous PostgreSQL. Bien que le sujet fasse mention d’un développement en Javascript, à la demande de Mme Gayral ce langage n’a pas été utilisé car il n’a pas été abordé en cours.

Solutions de mise en œuvre

Conception

A la suite de l’affectation finale de nos projets, nous avons tout d’abord défini au mieux notre travail. Cette première phase a constitué l’étape initiale de la conception, avec le recensement des différentes tâches à effectuer. Les langages de programmation étant imposés, il seulement fallu choisir les outils adaptés à leur mise en œuvre. Une fois les tâches écrites sous forme de user stories\*, classées par ordre d’importance et leur chronologie déterminée (voir annexe Trello), nous avons réparti le travail entre les membres du groupe. Il a été décidé que le développement front-end serait assuré par Célia Aït Mammar, la création et la gestion de la base de données par Nathan Gatto, et le développement back-end par Lucie Godefert, Florian Kotecki et Charlie Didier.

Le site web est structuré suivant un schéma défini préalablement par une arborescence regroupant toutes les pages créées (voir annexe 3). Cette arborescence nous a servi pour répartir l’implémentation des pages entre les développeurs back-end.

Développement

Comme dit plus haut, l’implémentation a été scindée en deux parties, le développement de l’interface questionneur et le développement de l’interface répondeur à partir des pages html créées par le développeur front-end.

/////////////////////////////Parler des choix de conception de la bdd+des scripts/////////////////////////////

/////////////////////////////Parler des choix de conception du design-html,css/////////////////////////////

Partie Questionneur

1.parler des outils utilisés

2.page par page, décrire leur utilité, dire qui les a faites entre F et C.

Partie Répondeur-L-

Auteur : Lucie.

En tant que développeur back-end, j’ai développé l’interface répondeur en réponse aux exigences du client.

Afin de concevoir et d’implémenter cette partie, j’ai utilisé différents outils et logiciels libres. Le php a été écrit sur l’éditeur de texte libre Notepad ++. L’application web PhpPgAdmin ainsi que système de gestion de bases de données PostgreSQL installés sur la plateforme de développement WampServer m’a ont permis de faire fonctionner localement mes scripts php.

La partie répondeur est constituée de sept pages html, qui suivent un schéma linéaire du choix du QCM au résultat de l’élève.

Je vais aborder dans cette partie la fonction et l’utilité de chaque page de l’interface répondeur. Pour plus de clarté, j’explique sommairement le mode de fonctionnement des pages. Les détails plus précis sur le fonctionnement de certains codes sont eux disponibles dans la partie algorithme du compte-rendu.

Page 1.ChoixRD.php

Temps.

Le php de cette page sert à afficher les différents domaines de QCM présents dans la base de données. Le répondeur a le choix entre plusieurs catégories dans lesquelles seront classés les QCM. Cette page a été la première fonctionnalité développée pour cette partie, car le choix entre plusieurs domaines est le premier choix à faire pour le répondeur lors de la résolution d’un QCM.

Afin d’afficher les domaines, un requête préparée a été utilisée. L’affichage des domaines se fait à l’aide d’ ‘echo’ comme pour tous les affichages en php, et le passage à la page suivante (ChoixRDS.php) se fait avec un formulaire. Le formulaire prend en paramètres dans une variable $\_POST\* le nom et l’identifiant du domaine sur lequel clique le répondeur pour les utiliser ultérieurement.

Page 2.ChoixRDS.php

Temps.

Après avoir choisi le domaine, le répondeur est amené à choisir soit un sous-domaine, soit directement un QCM. Un domaine peut en effet contenir un ou plusieurs sous-domaines : par exemple, le domaine ‘Informatique’ pourra contenir les sous-domaines ‘php’, ‘Java’, ou encore ‘Programmation Orientée Objet’.

Les QCM présents sur la même page que les sous-domaines sont des QCM sans sous-domaine, c’est-à-dire qui regroupent des questions générales sur le domaine ciblé. Dans le cas de l’exemple ‘Informatique’, on pourra par exemple trouver dans un QCM sans sous-domaine des questions portant sur plusieurs thèmes différents en informatique, comme Java, C, ou SQL.

Ces QCM et ces sous-domaines s’affichent sur la page ChoixRDS.php grâce à deux requêtes préparées. Une condition ‘if’ encadre le code PHP et permet de vérifier si le domaine est renseigné dans le $\_POST, afin d’éviter les exceptions liées à l’absence éventuelle de cette valeur. Si le domaine est renseigné, une requête préparée sélectionne et affiche tous les sous-domaines liés à ce domaine.

Si le domaine ne contient pas de sous-domaines, le répondeur en est informé (voir les explications dans la partie Page 3.ChoixRQI.php pour plus de détails).

Un bouton ‘retour’ permet dans tous les cas de revenir à la page précédente si le répondeur s’est trompé de domaine, ou s’il a changé d’avis.

Une autre requête permet de sélectionner et d’afficher les QCM sans sous-domaine qui ont été rendus visibles par leur créateur (voir partie questionneur pour plus de détails sur la visibilité d’un QCM).

Les deux requètes utilisent la valeur contenue dans le $\_POST pour sélectionner les sous-domaines et QCM à afficher.

Lorsque le répondeur clique sur le sous-domaine qu’il a choisi, un formulaire prend en paramètre les noms du domaine courant et du sous domaine sélectionné, et fait le lien vers la page ChoixRQI.php.

Si le répondeur clique sur un QCM sans sous-domaine, un autre formulaire fait le lien vers la page Executer.php qui permet de répondre directement au QCM, avec en paramètre les valeurs de l’identifiant du QCM (id\_qcm) et le nom du domaine courant.

Page 3.ChoixRQI.php

Temps.

Dans le cas où le répondeur n’a pas choisi d’effectuer un QCM sans sous-domaine, il se trouve sur cette troisième page après le choix du sous-domaine et du domaine. De la même manière que la deuxième requête de la page précédente, cette page affiche les QCM présents dans le domaine et le sous-domaine courants, sélectionnés avec une requête préparée.

De même que pour le choix du sous-domaine sur la page ChoixRDS.php, si aucun QCM n’est répertorié dans le sous-domaine choisi, le répondeur en est informé. Pour cela, la requête affichant les QCM incrémente un compteur à chaque QCM trouvé dans la base. Si le compteur est égal à 0 à la fin de l’exécution de la requête, cela signifie qu’il n’y a aucun QCM dans ce sous-domaine. Le formulaire activé lors du choix d’un QCM par le répondeur fait le lien avec la page Executer.php en passant en paramètre les valeurs utiles, soit l’id qcm, et les noms du domaine et sous-domaine auxquels appartient le QCM choisi.

Page 4.Executer.php

Temps.

Après avoir choisi le QCM à effectuer, le répondeur arrive sur la page Executer.php. L’enregistrement de la date courante à l’aide de la fonction php prédéfinie time() constitue la première action de la page. Cette variable servira à calculer le temps mis par le répondeur pour effectuer le QCM. Le temps total alloué au QCM à effectuer est récupéré grâce à une première requête préparée. La page Executer.php permet d’afficher toutes les questions du QCM dont l’id a été passée en paramètre, et toutes les réponses possibles pour chaque question. Le QCM s’affiche par l’imbrication de deux requêtes préparées : la première affiche les questions du QCM dont l’id a été passée en $\_POST, et la seconde utilise l’id de la question affichée pour sélectionner toutes les réponses qui s’y rapportent.

Les questions et les réponses ont un mode d’affichage sécurisé par l’emploi de la fonction prédéfinie en php htmlspecialchars, qui permet dans ce cas d’afficher les données sous forme de chaine, sans interprétation du code. Par exemple, une réponse ‘</br>’ affichée avec htmlspecialchars ne sera pas interprétée par le script php comme un retour à la ligne, mais bien comme la chaine de caractères ‘</br>’.

Toutes les réponses possibles possèdent une case que le répondeur peut cocher pour les choisir ; les cases cochées sont enregistrées dans un formulaire dans un tableau de $\_POST, de même que la date de début du QCM et son identifiant.

Un bouton ‘submit’ lié au formulaire permet au répondeur de valider le QCM lorsqu’il est satisfait des réponses cochées, et fait le lien avec la page Statistique.php.

Tout le code php est encadré par une condition qui vérifie le bon déroulement du choix du QCM, et l’appel de la page EviteMessageFormulaire.php interdit le retour à la page précédente pour empêcher l’affichage d’éventuelles erreurs ou exceptions\* .

Page 5.Statistique.php

Temps.

Après avoir validé son QCM, le répondeur arrive sur la page Statistique.php. Cette page va calculer sa note et son temps, puis les enregistrer dans une table qui gardera le résultat.

La page Statistique.php affiche également l’identifiant du QCM, chaque question pour laquelle le répondeur a coché une ou plusieurs cases, et les réponses données. Dans le cas d’une bonne réponse, la réponse s’affiche en vert. Si le répondeur a mal répondu, sa réponse s’affiche en rouge, et la bonne réponse s’affiche en vert en-dessous. A chaque bonne réponse de la part du répondeur, le score total, ou note, s’incrémente de la valeur décidée par le questionneur à la création du QCM (voir la partie questionneur pour plus de détails). Des requêtes imbriquées sont là aussi utilisées pour afficher les différentes réponses de chaque question : la première sélectionne tout d’abord les questions grâce aux valeurs enregistrées dans le $\_POST. Pour chaque question, la seconde requête préparée sélectionne les réponses s’y rapportant. Quand une réponse cochée enregistrée dans le $\_POST correspond à une réponse correcte, le score est incrémenté. Sinon, si la réponse cochée est incorrecte, une troisième requête trouve et affiche les réponses justes, et le score n’est pas incrémenté. Enfin, l’incrémentation du score s’affiche à la fin du traitement de la question (ex : Score +1 si la question vaut un point).

La date actuelle enregistrée avec time() sert à calculer le temps passé sur le QCM, par une soustraction entre cette date actuelle et la date passée en $\_POST. Le répondeur est averti s’il a dépassé le temps alloué au QCM, et le score est décrémenté. Enfin, le résultat du répondeur est enregistré.

Lorsque le répondeur a fini d’effectuer son QCM et qu’il a vu la correction sur la page Statistique.php, Deux boutons lui permettent soit de refaire un QCM, soit d’aller consulter ses résultats aux QCM qu’il a fait précédemment sur la page Profil.php.

Page 6.Profil.php

Temps.

La page Statistique.php

Page 7.EviteMessageFomulaire.php

Temps.

La page EviteMessageFormulaire.php est utile pour empêcher le retour à une page précédente lorsque ce retour n’est pas souhaitable pour le bon déroulement de l’exécution d’un QCM. Par exemple, une fois le QCM validé, le répondeur ne doit pas pouvoir y revenir pour modifier ses réponses. Cette page évite également le message d’erreur ‘Confirmer le nouvel envoi du formulaire’ lors du retour arrière. Certaines des fonctionnalités de cette page n’ayant pas été abordées en cours, je me suis inspirée de l’exemple présent sur le site Openclassroom. EviteMessageFormulaire.php vérifie d’abord si des valeurs sont enregistrées en $\_POST et les sauvegarde dans un $\_SESSION\* le cas échéant. La location courante est sauvegardée. Puis lors de l’appel à la page précédente le répondeur est redirigé sur cette page mais n’a plus accès à la modification de quoi que ce soit. Les $\_POST sauvegardés sont ensuite remis dans $\_POST, et le script se termine par la destruction de la sauvegarde.

Problèmes rencontrés et leur résolution

Algorithmes

Synthèse des comptes rendus des points de rendez-vous avec l'enseignant et le client –L-

Tout au long du projet, les rendez-vous avec le client ont marqué les étapes de la conception puis de l’implémentation. Du 21 septembre 2016 au 5 janvier 2017, huit rendez-vous ont été pris régulièrement pour faire part de l’avancement du projet et des améliorations à y apporter.

Le premier rendez-vous, pris le 21/09, portait sur les Précisions sur la définition du sujet de projet« Gestion des QCM ».

Le deuxième rendez-vous, pris le 28/09, portait sur la Répartition des tâches, base de données

Le troisième rendez-vous, pris le 05/10, portait sur la présentation du site et la création de la base de données.

Le quatrième rendez-vous, pris le 19/10, portait sur l’hébergement du projet et la présentation de la fonctionnalité de tri des QCM

Le cinquième rendez-vous, pris le 09/11, portait sur la présentation de la connexion à la base de données, la création de question pour le rôle questionneur, le déroulement d’un QCM pour le rôle répondeur, et la modification du design et de la base de données.

Le sixième rendez-vous, pris le 30/11, portait sur

Le septième rendez-vous, pris le 21/09, portait

Le huitième rendez-vous, pris le 21/09, portait

Conclusion

En conclusion, malgré les problèmes rencontrés, le projet a atteint son stade de développement final. Même si des améliorations sont toujours possibles, le site est fonctionnel et répond aux exigences du client. Faire partie d’une équipe de développement nous a appris à travailler en coordination sous la direction d’un chef de projet, et à répartir les taches en fonction des compétences de chacun. La communication, essentielle entre les différentes parties et au sein du groupe projet, nous a permis d’arriver ensemble plus rapidement au but désiré.

Annexes

-1.Trello-> user stories

-2.shcéma relationnel base de données

-3.arborescence du site

Glossaire

Développement front-end : Partie du développement concernant la forme de l’application ou du logiciel. Le développeur front-end utilise des langages de structuration et de présentation comme html et css.

Développement back-end : Partie du développement concernant le fond de l’application ou du logiciel. Le développeur back-end utilise des langages de programmation comme php, javascript, ajax par exemple.

PDO (php avec l’extension) : PDO, ou PHP Data Objtects, est une extension du langage php, définissant une interface d’accès à une base de données, et permettant son utilisation au sein du php.

User stories (ou récit utilisateur): Phrase écrite dans un langage simple qui permet de décrire le contenu d’une fonctionnalité à développer sous la forme « En tant que <qui> je souhaite <quoi> afin de <pourquoi> ».

$\_POST : Tableau de variables en php, qui permet de stocker des valeurs et les passer d’une page html à une autre, sans faire apparaitre ces valeurs dans l’url de la page.

Exception : Erreurs lancées par PHP lorsqu’une anomalie de traitement est détectée.

$\_SESSION : De la même manière que $\_POST, $\_SESSION permet de stocker des valeurs, mais au lieu d’être accessible sur une seule page, ces valeurs sont accessibles sur toute page contenant la ligne ‘session\_start()’ dans le script php.