



UNIVERSITÉ
CAEN
NORMANDIE

Licence 3 Informatique

Rapport de Projet : **PartnerQuest**

CORNILLEAU Antoine
EL HADRI Ilias
LECOMTE Lucas
LELOUP Alexis

Table des matières :

Présentation du projet	3
Description du projet et des objectifs	3
Les fonctionnalités implémentées	4
Description des fonctionnalités	4
Organisation du projet	4
Les éléments techniques	6
Fonctionnement	6
Comparaison des données des quiz	6
Problèmes rencontrés	6
Des nouvelles parties	6
L'architecture du projet	7
Arborescence du projet (MVCR)	7
Diagramme de la base de donnée	7
Conclusion	8
Récapitulatif des fonctionnalités principales	8
Propositions d'améliorations	8

Liens utiles :

[GitHub](#)

[URL du projet](#)

1. Présentation du projet

1.1. Description du projet et des objectifs

Dans le cadre du module INF6D3 Projet Annuel, notre tuteur Marc Spaniol nous a confié la tâche de développer le projet Partner Quest. Le projet du PartnerQuest est une application web qui permet de tester la compatibilité de deux personnes à l'aide de quiz simple sur un thème précis. En se connectant à son compte utilisateur sur le site PartnerQuest, une personne peut créer un quiz, ensuite quand quelqu'un répond à ce quiz il peut comparer ses réponses avec les réponses d'un ami ayant répondu au même quiz. L'application analyse les réponses des deux participants et calcul une pourcentage de correspondance. Cette note est présentée à l'utilisateur sous la forme d'une barre de progression et le détail des réponses similaires est affiché en même temps. Le projet Partner Quest est développé en PHP et doit donc être mis en place sur un serveur.

1.2. Description de l'apprentissage derrière l'application

Le développement d'un tel projet nous permet de nous familiariser avec les langages PHP, HTML, CSS ainsi que la mise en place d'architecture Modèle Vue Contrôleur Routeur (MVCR).

De manière plus générale, on apprend à optimiser notre code, éviter la redondance, anticiper et corriger les bugs puis concevoir et comprendre le fonctionnement de l'architecture MVCR. Comme tout projet de groupe, ce travail développe nos facultés à travailler en équipe. En divisant les tâches pour accroître la productivité, il a fallut s'adapter aux forces et faiblesses de chacun pour réaliser l'objectif commun : rendre un projet cohérent avec le travail demandé.

2. Les fonctionnalités implémentées

2.1. Description des fonctionnalités

Toutes les spécifications du projet ont été discutées pendant deux réunions entre l'équipe de développement et le commanditaire du projet. La première qui a permis de définir la problématique du sujet et les différentes fonctionnalités à développer et la deuxième pendant laquelle nous avons déterminé le plan du rapport et convenu de la modélisation de la base de données.

Les fonctionnalités principales que nous avons développées pour ce projet étaient les suivantes :

- Un utilisateur peut s'inscrire et se connecter.
- Un utilisateur peut créer un quiz et le soumettre aux autres utilisateurs.
- Un utilisateur peut répondre à un quiz et enregistrer ses réponses.
- Deux utilisateurs doivent pouvoir comparer leurs résultats sur un même quiz.

En plus de cela nous avons ajouté d'autres fonctionnalités qui n'étaient pas explicitement demandées mais sans lesquelles la qualité de Partner Quest aurait été diminuée :

- Un utilisateur peut réinitialiser son mot de passe grâce à un code qui lui est envoyé par mail.
- Chaque utilisateur dispose d'un espace personnel dans lequel il peut modifier ses informations personnelles (adresse mail et mot de passe), mais il peut aussi y consulter les réponses des quiz effectués et les quiz qu'il a créés.
- Il est possible pour un utilisateur de modifier ou supprimer les quiz.
- L'administrateur peut supprimer des quiz, supprimer un utilisateur ou modifier le statut de celui-ci(admin/user).
- Les utilisateurs peuvent avoir une photo de profil.

Les utilisateurs peuvent également accéder à une page personnelle qui permet de visualiser les quiz créés par l'utilisateur et ceux auxquels ils ont répondu. Si un utilisateur n'est pas connecté, il ne peut voir que les différents quiz déjà créés sur le site. Pour répondre à un quizz, il faut créer ou se connecter à un compte utilisateur.

2.2. Organisation du projet

Pour mener à bien ce projet nous avons rencontré le commanditaire 2 fois. La première, avant le début de la conception de manière à s'assurer d'avoir compris le besoin de notre client. La seconde, 2 semaine après le premier rendez-vous pour nous assurer d'être sur la bonne voie et que la vision que nous avons de PartnerQuest corresponde à celle de Marc Spaniol.

Chaque membre du groupe a travaillé sur l'ensemble des fonctionnalités apportant son point de vue et son aide..

Priorité des phases de développement :

- Phase 1 : Conception du projet (UML).
- Phase 2 : Mise en place de la MVC et mise en place de la base de données SQL.
- Phase 4 : Développement des comptes (création de comptes, connexion, inscription).
- Phase 5 : Développement la création d'un quiz.
- Phase 6 : Développement de l'interface personnel.
- Phase 7 : Développement de l'interface social(amis et comparaison des quiz).
- Phase 7 : Développement de l'édition et la suppression d'un quiz existant.
- Phase 7 : Développement de l'interface administrateur.
- Phase 8 : Rédaction du rapport.

3. Eléments techniques

Notre projet se base sur un motif d'architecture logicielle MVCR (Model View Controller Router) ce qui lui confère une conception claire et efficace ainsi qu'une maintenance facilitée.

Pour la majorité des composants graphique nous utilisons le framework Bootstrap qui nous offre une collection d'outils utiles à la création du design de notre application web.

3.1. Fonctionnement de la MVC

L'architecture MVC s'articule autour de trois classes principales, ces classes sont initialisées grâce au fichier *index.php* :

Tout d'abord le routeur, celui-ci s'occupe de la redirection des pages WEB, il permet de changer la vue lorsque l'utilisateur clique sur un bouton et d'orienter vers la bonne redirection, il passe aussi les variables **\$_POST** résultantes des formulaires au contrôleur.

Ensuite, le contrôleur, celui-ci traite toute les données et s'occupe d'ordonner les requêtes SQL. Il a aussi pour rôle de vérifier les données saisies par un utilisateur, son statut et son droit d'accès à des pages afin d'éviter toute tentative de fraude. C'est l'élément central de l'architecture MVC.

Enfin, la vue, celle-ci gère l'affichage du code HTML. Elle est composée d'un squelette que l'on remplit avec les données voulues. Le squelette contient le header du fichier html, les imports de css et javascript, la barre de navigation et le pied de page.

3.2. Objets quiz et account

Deux objets facilitent la gestion des données en PHP, l'objet Quiz et Account.

Quiz:

L'objet Quiz est créé grâce à un Quiz Builder. Un quiz est composé:

- D'un nom
- Du nombre de questions.
- D'un thème.
- D'une description.
- De l'id de l'utilisateur qui l'a créé.
- D'une liste contenant les couples question/réponse.

Le **QuizBuilder** traite les données des formulaires et les normalise pour en faciliter le stockage, il vérifie aussi si il y a des erreurs comme par exemple une information manquante, si c'est le cas il signale l'erreur au contrôleur grace a la fonction isValid().

Account :

L'objet Account est créé grâce à un **UserBuilder**. Un Account est composé:

- D'un nom
- D'un prenom
- D'un mail.
- D'un mot de passe chiffré.
- D'un statut

Le UserBuilder permet la création de l'objet Account, cet objet est stocké en **\$_SESSION** afin de pouvoir obtenir à tout moment les données relatives à l'utilisateur.

\$_SESSION est un moyen simple pour stocker des données utilisateur en utilisant un identifiant de session unique. Elle est utilisée pour faire persister des informations entre plusieurs pages via des cookies de session. L'absence d'un identifiant ou d'un cookie de session indique à PHP de créer une nouvelle session, et génère ainsi un nouvel identifiant de session.

3.3. SQL côté PHP

Pour réaliser des requêtes PHP, nous utilisons PHP Data Objects (PDO), c'est une extension définissant l'interface pour accéder à une base de données avec PHP. Cet extension constitue une couche d'abstraction qui intervient entre l'application PHP et un système de gestion de base de données MySQL.

Elle permet de récupérer en une seule reprise tous les enregistrements de la table sous forme d'une variable PHP (array()) de type tableau à deux dimensions ce qui réduit le temps de traitement.

Pour gérer les requêtes MySQL via PDO, deux classes ont été mises en places:

AccountStorageMySQL gère toute les requêtes concernant les comptes utilisateurs, de la création à la suppression.

Pour manipuler un utilisateur, la classe utilise un objet Account comme vu précédemment.

QuizStorageMySQL gère toute les requêtes concernant les quiz.

Comme pour un utilisateur, la classe utilise un objet Quiz vu plus haut.

3.4. Manipulations autour des quiz

Comparaison des réponses:

Une fois que deux utilisateurs ont ajouté leurs réponses, le contrôleur calcule un pourcentage de correspondance. Il est calculé simplement en faisant une moyenne des pourcentages de réponses similaire a chaque questions.


```

$affinity=array();

foreach ($user_array as $value){
    $allFriendAnswers=$friend_array[$value["question_id"]]["reponse"];
    $commonAnswers[$value["question_id"]]=array_intersect($value["reponse"],$allFriendAnswers);

    $nbCommonAnswers=count($commonAnswers[$value["question_id"]]);
    $nbAnswers=max(count($allFriendAnswers),count($value["reponse"]));
    array_push( &array: $affinity, ...vars: $nbCommonAnswers/$nbAnswers);
}
$affinity=intval( var: (array_sum($affinity)/count($affinity))*100);

```

Techniquement, on parcourt l'array **\$user_array** qui contient notamment les réponses de l'utilisateur courant pour chaque question. On stocke les réponses communes aux deux utilisateurs pour une même question grâce à **array_intersect()**.

Puis on calcul le pourcentage de réponses communes de cette façon :

→ Nombre de réponses communes / maximum(nombre de réponses de l'utilisateur, nombre de réponses de l'ami)

On stocke le résultat dans une liste puis on continue, une fois toutes les réponses parcourues on fait la moyenne de celles-ci.

3.5. Problèmes rencontrés

Nous n'avons pas rencontré de difficulté majeures durant le développement du projet hormis la contrainte de temps assez courte qui nous a poussé à augmenter la cadence de travail.

Cependant, la modification d'un quiz par l'utilisateur fut complexe, en effet, nous souhaitions à l'origine que lorsque l'utilisateur modifie un quiz, les réponses des utilisateurs ayant déjà participé à ce quiz soient conservées, malheureusement nous n'avons pas réussi à le mettre en place.

Concrètement, chaque réponse possède un Id que l'on génère à la création, exemple :

- -oui **ID=1**
- -non **ID=2**
- -parfois **ID=3**

Admettons que l'utilisateur veuille modifier les réponses à cette question et qu'il ajoute des réponse et/ou change l'ordre des réponses:

- -non **ID=1? non ID=2**
- -parfois **ID=2? non ID=nouvel ID**
- -oui **ID=3? non ID=1**

Voilà pourquoi, si l'utilisateur souhaite modifier les questions/réponses d'un Quiz, il doit écraser toute les données relatives à celui-ci. L'utilisateur peut cependant modifier le nom et la description du Quiz sans altérer celui-ci.

Nous avons aussi un problème avec la police, en effet nous utilisons la police BIGJOHN qui ne supporte pas les accents, malheureusement nous n'avons pas eu le temps de résoudre ce problème.

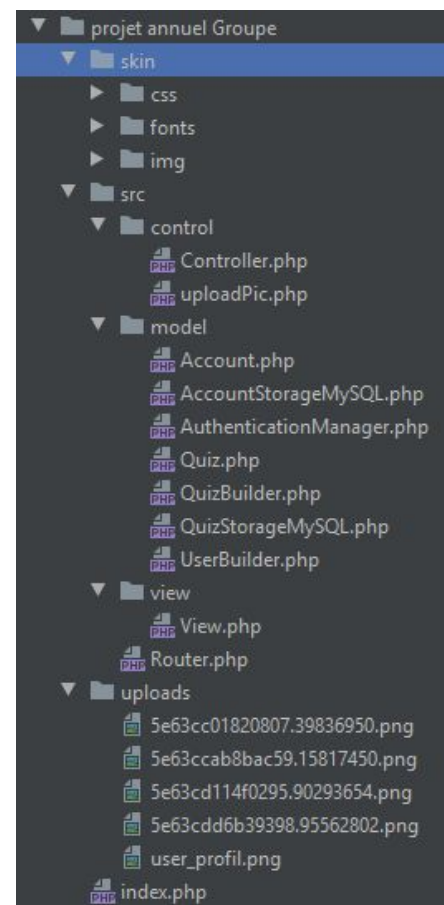
4. L'architecture du projet

4.1. Arborescence du projet (MVCR)

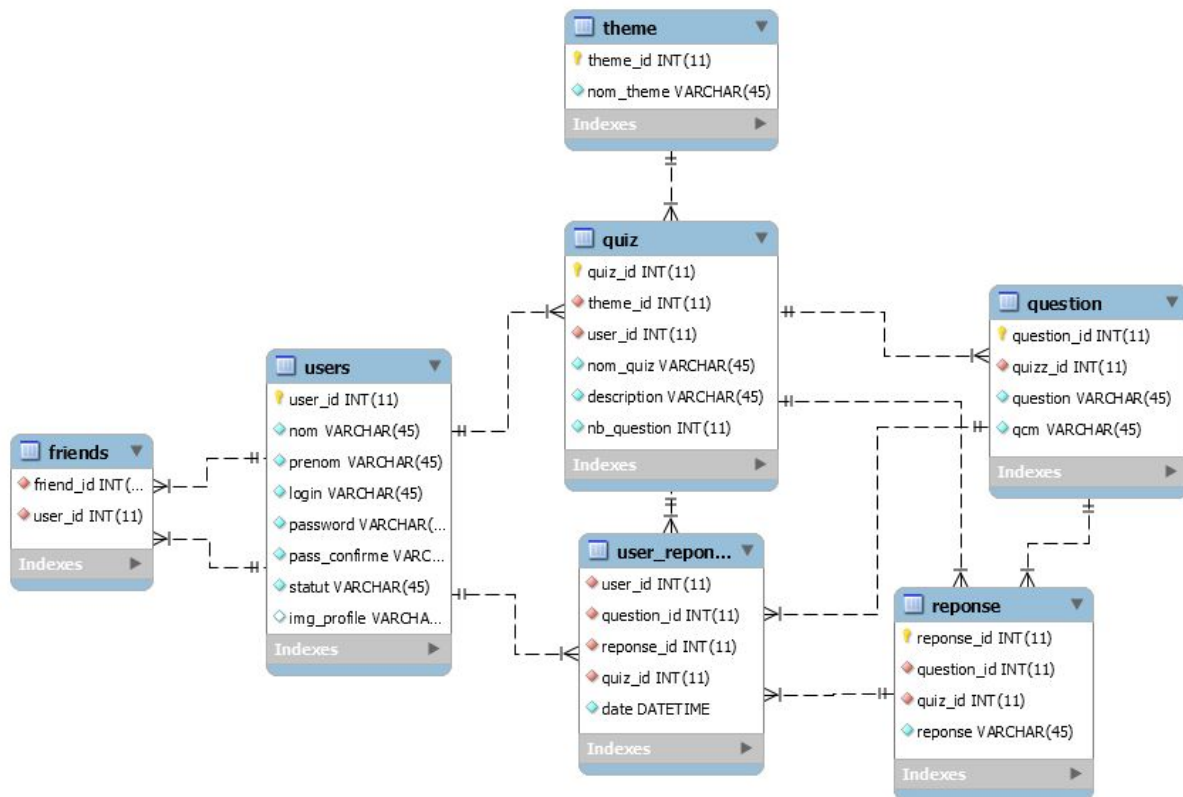
Le projet se compose de 3 répertoires principaux, parmi eux **skin** qui contient les fichiers css, les images du site et les polices.

Ensuite, le répertoire **src** qui contient tout les fichiers php pour la MVC(model, view, control, router).

Enfin, le dossier **uploads**, qui contient les images des photos de profil utilisateurs.



4.2. Diagramme de la base de donnée



Pour modéliser la base de donnée, nous avons travaillé pour faire en sorte de créer des tables cohérentes et simples pour pouvoir extraire les données et retrouver les similarités facilement.

Tout d'abord nous avons une table **users** qui contient toutes les informations sur l'utilisateur, la clé primaire de cette table est `User_id`, elle permet de rendre unique un utilisateur et de faciliter la modification des données, en effet même si un utilisateur change tout les paramètres relatifs à son compte, celle-ci reste inchangée.

La table **users** est connectée à la table **friends** qui est composée de deux attributs qui font références à des `user_id` car un utilisateur peut avoir plusieurs amis. Les `friend_id` représentent les `user_id` des amis de l'utilisateur.

Nous avons une table **réponse** qui contient le contenu de la réponse et dont sa clé primaire est `reponse_id`.

Par symétrie, la table **question** stocke le contenu des question et sa clé primaire est `question_id`.

Les tables **question** et **réponse** sont connectées à la table **quiz** car un **quiz** est un élément central composé de questions et de réponses. Sa clé primaire est `quiz_id` et on lui attribue un thème c'est pour cela que la table **quiz** est reliée à la table **theme** avec comme clé primaire `theme_id`.

La table **user_reponse** fais le lien entre question, réponse et users.Elle permet de stocker les réponses d'un utilisateur.

5. Utilisation du site

Pour avoir une idée visuellement du rendu du site et de son fonctionnement nous allons détailler les différentes vues dans le site. Les fonctionnalités implémentées sont donc mises en avant dans les différentes captures d'écrans, pour montrer les procédés utilisés pour développer notre projet.

Premièrement nous avons l'accueil/le menu du site qui met en avance les utilisations du site et qui propose plusieurs thèmes de quiz populaires auprès des utilisateurs.



Sur la page d'accueil il y a en haut une barre de navigation avec le logo du site qui permet de retourner à l'accueil, le bouton "social" qui renvoie vers une page permettant de voir ses amis, comparer ses résultats et d'ajouter des nouveaux amis. Il y a aussi un bouton "ajouter un quiz" qui permet par le remplissage d'un formulaire de créer son propre quiz, le bouton "page perso" qui permet d'accéder à la modification de ses informations personnelles, ses réponses et ses quiz. Et enfin le bouton "déconnexion" qui permet à un utilisateur de se déconnecter ou de se connecter/s'inscrire s'il ne l'est pas déjà.


Sur la page d'accueil nous pouvons donc voir tous les thèmes de quiz disponibles en cliquant sur le bouton "voir tous".



Quand un utilisateur choisit un thème de quiz, plusieurs quiz sont proposés et il peut donc remplir un formulaire pour répondre au quiz voulu comme montré ci-dessous.

A quiz registration form with a dark blue header bar containing a logo and navigation links: SOCIAL, AJOUTER UN QUIZ, PAGE PERSO, and DECONNEXION. The form has two sections: "AIMES-TU SURFER LA VAGUE ?" with a dropdown menu showing "oui" and "non", and "TON ACTIVITE PREFEREE? (CHOIX MULTIPLES)" with a list box showing "Tennis", "Basketball", and "PingPong". A blue "ENREGISTRER" button is at the bottom.

Après avoir répondu à toutes les questions, on peut retourner sur la “page perso” et comparer les quiz en communs avec l’ami de son choix pour voir la similarité des réponses et savoir si vous connaissez bien votre ami.

SOCIAL AJOUTER UN QUIZ PAGE PERSO DECONNEXION

ACTIVITES

TES ACTIVITES PREFERES

VOS REPONSES

Aimes-tu surfer la vague ?
❌ oui

Ton activite preferee?
✅ Tennis
❌ Basketball
✅ Skate

REPONSES DE VOTRE AMI

Aimes-tu surfer la vague ?
❌ non

Ton activite preferee?
✅ Tennis
✅ Skate

AFFINITE

Vous êtes plutôt différents...

33%

Comparer un autre quiz

6. Conclusion

6.1. Récapitulatif des fonctionnalités principales

Au cours de la réalisation de ce projet tout c'est déroulé sans soucis majeurs, cependant nous avons été légèrement pris par le temps ce pourquoi nous avons dû faire l'impasse sur certaines fonctionnalités ou designs plus aboutis, malgré tout celui-ci a quand même abouti à un résultat convainquant et ergonomique.

Le site est totalement fonctionnel et toutes les fonctionnalités demandées sont opérationnelles.

Nous sommes satisfaits du travail produit.

6.2. Propositions d'améliorations

Le développement du projet Partner Quest est arrivé à un stade qui permet son exploitation. Les principales fonctionnalités sont développées cependant il reste certains points sur lesquels nous pourrions apporter des modifications ou des améliorations.

Nous pourrions par exemple imaginer un système plus complet de gestion d'amis. Chaque utilisateurs pourrait avoir la possibilité d'envoyer des demandes qui donneraient la possibilité à un ami de répondre au même quiz

Un système de messagerie instantanée ou de commentaires améliorerait aussi l'expérience des utilisateurs en leur offrant la possibilité de débattre sur les différents quiz disponibles.

Un système de fil d'actualité permettra aux utilisateurs de visualiser les quiz et les réponses de leurs amis, tandis qu'un système de vote et de partage leur permettraient de trier les meilleurs quiz sur la page d'accueil

Enfin il serait également envisageable d'étendre la parenté des quiz à plusieurs utilisateurs, afin de comparé les réponses de plusieurs utilisateurs simultanément.