Lycée Louis de Foix

Co-voiturage

1. Introduction

Le covoiturage est un moyen de transport qui consiste à partager un véhicule avec d'autres personnes voyageant dans la même direction. La société **BlaBlaAutoa** fait appel à une société de service en ingénierie Informatique **KapGeminik** pour développer son projet.

Ce projet vise à développer une application web qui permet à des utilisateurs de proposer ou de rechercher des trajets de covoiturage. L'application gérera également la réservation de places, les notifications et la gestion des utilisateurs.

2. Objectif Principal

Développer une application web de gestion de covoiturage en PHP avec une architecture MVC (Modèle-Vue-Contrôleur) et une base de données MySQL. L'application doit permettre aux utilisateurs

- de proposer des trajets,
- de rechercher des trajets disponibles et
- de réserver des places.

3. Environnement technique

- Utiliser le langage de programmation php
- Vous pouvez utiliser l'EDI Netbeans
- Configurer un serveur web Apache ou Nginx local ou distant (ex. : XAMPP, WAMP).
- Configurer la base de données MySQL avec par exelple phpmyadmin.
- Possibilité d'utiliser rdocker

4. Fonctionnalités à implémenter

4.1. Gestion des utilisateurs

- **Inscription** : Les utilisateurs doivent pouvoir s'inscrire avec un nom, un prénom, un email, un mot de passe, et éventuellement un numéro de téléphone.
- **Connexion** : Les utilisateurs doivent pouvoir se connecter avec leur email et mot de passe.
- **Gestion de profil** : Chaque utilisateur doit pouvoir modifier ses informations personnelles (nom, email, mot de
- passe, etc.).
- **Déconnexion** : Les utilisateurs doivent pouvoir se déconnecter.

4.2. Proposition de trajet (conducteur)

- Les conducteurs doivent pouvoir proposer un trajet en fournissant les informations suivantes :
  - Point de départ et d'arrivée
  - Date et heure de départ
  - Nombre de places disponibles
  - Prix par place
- Ils doivent pouvoir gérer leurs trajets (modifier, supprimer).

4.3. Recherche de trajet (passager)

- Les utilisateurs doivent pouvoir rechercher des trajets en fonction :
  - Du lieu de départ et d'arrivée
  - De la date et de l'heure souhaitées
- Affichage des trajets disponibles avec les informations du conducteur, prix et nombre de places restantes.

4.4. Réservation d'un trajet

- Les passagers doivent pouvoir réserver une ou plusieurs places dans un trajet proposé par un conducteur.
- Le conducteur reçoit une notification lorsqu'une réservation est effectuée.
- Gestion des statuts de réservation : (en attente, confirmée, annulée).

4.5. Historique des trajets

- Chaque utilisateur doit pouvoir accéder à l'historique des trajets auxquels il a participé (en tant que conducteur ou passager).

4.6. Gestion des avis et notations

- Après chaque trajet, les utilisateurs doivent évaluer le conducteur et laisser un commentaire.
- Les conducteurs doivent aussi pouvoir évaluer les passagers.

4.7. Administration

- Une interface d'administration pour gérer les utilisateurs, les trajets, et les réservations.
- Modération des avis laissés par les utilisateurs.

5. Architecture logicielle

5.1. MVC

1. **Modèle (M)** :
    - Représente la base de données et les classes qui interagissent avec la base (ex. : `User`,... etc.).
    - Chaque modèle correspond à une table dans la base de données et gère la logique métier.
2. **Vue (V)** :
    - Les vues sont les interfaces utilisateur (HTML, CSS, JavaScript) affichant les données fournies par le contrôleur.
    - Utiliser des fichiers PHP pour générer dynamiquement les pages web.
3. **Contrôleur (C)** :
    - Le contrôleur reçoit les requêtes utilisateur, interagit avec les modèles pour récupérer les données, et retourne la vue correspondante.
    - Exemples de contrôleurs : `UserController`,`....`.

5.2. Base de données MySQL

Définir le MEA puis le modèle relationnel.

5.3. Étapes de Développement

1. **Implémentation du Modèle :**
    - Créer les modèles correspondant aux tables de la base de données.
    - Mettre en place des méthodes pour les opérations CRUD (Create, Read, Update, Delete).
2. **Implémentation des Vues :**

    Créer des templates PHP pour :

    - l'inscription,
    - la connexion,
    - la proposition et
    - la recherche de trajets,
    - la réservation, etc.
3. **Implémentation des Contrôleurs :**

    Développer les contrôleurs pour gérer les différentes fonctionnalités :

    - connexion,
    - proposition de trajet,
    - réservation
4. **Test et débogage :**
    - Tester l'application pour s'assurer du bon fonctionnement des fonctionnalités.
    - Corriger les bugs éventuels.

6. Cahier des charges technique

L'intérêt serait dans un découplage fort entre le serveur web et le serveur de bases de données. L'application web sera hébergée par des serveurs qui seront acquis par la société **BlaBlaAutoa**, et seront situés dans ses locaux.
La maintenance sera prise en charge par la SSII **KapGeminik**.
Dans un premier temps, elle disposera en plus de son serveur web, d'un serveur de base de données.
La solution sera développée avec les outils et méthodes couramment utilisés par le prestataire KapGeminik, à savoir :

- développement en PHP, MVC
- un SGBDR MariaDB oy MysQL pour la gestion des données.

7. Cahier des charges organisationnel

La base de données doit être conçue, validée par un enseignant, et mise en place avant la programmation des modules. Des occurrences en nombre suffisant et pertinentes devront être insérées.

Vous pouvez travailler en équipe de deux ou trois personnes au maximum. La répartition des tâches devra être explicitement définie et présentée. Vous pouvez éventuellement travailler sur la découverte d'une nouvelle technologie en équipe, puis réaliser chacun un module différent l'utilisant.

La réalisation du projet nécessite de suivre les étapes suivantes :

1. Conception du schéma global de données sous forme d'un diagramme de classe UML
2. Écriture/adaptation des classes métiers
3. Répartition des différentes fonctionnalités
4. Écriture des composants de l'application web en respectant le pattern MVC

Remarque importante :
Chaque étudiant doit prendre en charge les fonctionnalités qu'il développe du début à la fin : création et peuplement de la base de données, écriture des composants modèles, vues, et contrôleurs associés à chaque fonctionnalité.

8. Productions attendues ou livrables

- le schéma de la base de données
- les captures d'écran du trello réalisées à chaque séance d'AP
- les captures d'écran des interfaces réalisées, accompagnées d'un bilan qui tracera
  - les différentes phases de réalisation de chaque écran ainsi que
  - les difficultés rencontrées au cours de leur réalisation.
- le code source de l'application, pertinemment et commenté.

9. Feuille de route du développement

Voici la feuille de route que vous devrez suivre:

Séance	Résultats attendus	Groupe / Individuel
n°1 09/09	Répartition du travail Diagramme de classe Copies d'écran de Trello	G G G
2 16/09	Diagramme de cas d'utilisation Développements Copies d'écran de Trello (à montrer chaque semaine...)	G I G
3, 4 23/09	Développements Documentation du code Tests des méthodes Jeu d'essai Copies d'écran de Trello (à montrer chaque semaine...)	I I I I I
4 30/09		
5 07/10	Remise des documents	