



TOUCHE PAS AU KLAXON

Application de covoiturage inter-sites

Développeur : Godji

Date : Remi

github : <https://github.com/Remi-13-exe/covoiturage>

1 CONTEXTE & MISSION

Brief de mission

Vous êtes développeur au sein d'une grande entreprise disposant de plusieurs implantations géographiques.

De nombreux trajets sont réalisés quotidiennement entre les différents sites de l'entreprise (mobilité inter-sites).

Cependant, plusieurs véhicules effectuent souvent le même trajet avec un faible taux d'occupation — souvent un seul conducteur par voiture.

Objectif :

Mettre en place une **application web de covoiturage interne** permettant aux employés de publier leurs trajets et d'en consulter d'autres, afin de favoriser le partage de véhicule.

L'application doit être **déployée sur l'intranet** de l'entreprise et permettre :

- La **visualisation des trajets planifiés**, classés par date de départ croissante.
- La **connexion des employés** pour proposer un trajet, en modifier ou supprimer les leurs.
- La **consultation de détails** sur un trajet (conducteur, contact, places restantes).
- Une **administration complète** (gestion des trajets, utilisateurs, et agences).

Contraintes techniques

- **Langage principal :** PHP
- **Architecture :** MVC
- **Base de données :** MySQL / MariaDB
- **Front-end :** Bootstrap + Sass (palette de couleurs via variables Bootstrap)
- **Qualité du code :**
 - Documentation avec **DocBlock**
 - Vérification via **PHPStan**
 - Tests unitaires avec **PHPUnit** (au moins pour les écritures en base)
- **Déploiement :** Intranet (WAMP, XAMPP, ou équivalent)

Charte graphique imposée

Une palette de couleurs spécifique est imposée pour ce site, et d'autres projets futurs du groupe utiliseront des variantes.

C'est pourquoi **les variables Bootstrap** ont été utilisées afin de faciliter la réutilisation du code et du thème.

2 ARCHITECTURE DU SITE

Structure des pages

1. Header

- **Visiteur non connecté :**
 - Gauche : nom de l'application
 - Droite : bouton de connexion
- **Utilisateur connecté :**
 - Gauche : nom de l'application

- Droite : bouton “Créer un trajet”, prénom/nom, bouton déconnexion

- **Administrateur :**

- Gauche : nom (lien vers dashboard)
- Droite : menu horizontal (gestion utilisateurs, agences, trajets) + bouton déconnexion

2. Footer

- Nom de l'application + Copyright

3. Page d'accueil

Affiche les **trajets disponibles (avec places restantes)** triés par date de départ croissante.

Chaque ligne indique :

- Agence de départ / arrivée
- Dates de départ et d'arrivée
- Nombre de places disponibles

4. Utilisateur connecté

Affiche les mêmes informations que la page d'accueil + fonctionnalités supplémentaires :

- **Bouton Détails** → ouvre une fenêtre modale contenant :
 - Identité du conducteur
 - Téléphone et email
 - Nombre total de places
- **Boutons Modifier / Supprimer** si l'utilisateur est l'auteur du trajet

5. Création de trajet

- L'utilisateur saisit départ, arrivée, dates, et nombre de places.
- Ses infos personnelles (nom, email, téléphone) sont pré-remplies et non modifiables.
- Contrôles de cohérence :
 - Départ ≠ arrivée
 - Arrivée postérieure au départ

6. Tableau de bord administrateur

Fonctionnalités :

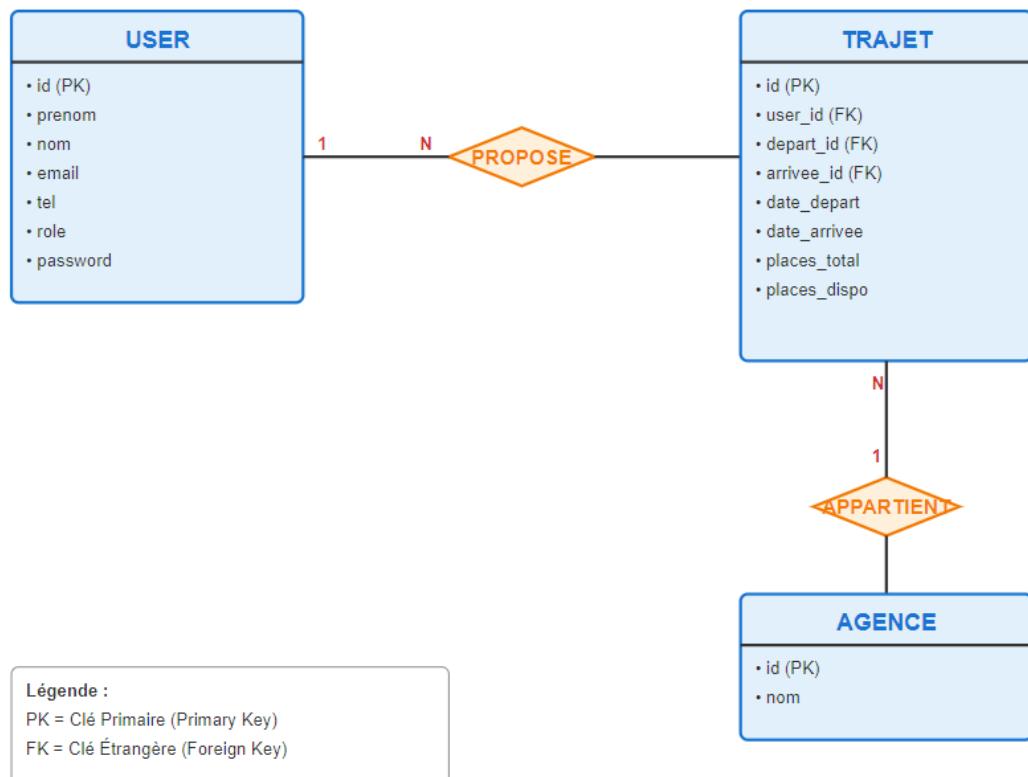
- Lister les utilisateurs
- Lister / créer / modifier / supprimer les agences
- Lister tous les trajets
- Supprimer un trajet

3 MODÈLE CONCEPTUEL DE DONNÉES (MCD)

User	1	N	Trajet	N	1	Agence
id			id	<th></th> <th>id</th>		id
prenom			user_id			nom
nom			depart_id			
email			arrivee_id			
tel			date_depart			
role			date_arrivee			
password			places_total			
			places_dispo			

Relations :

- Un utilisateur propose plusieurs trajets.
- Chaque trajet a un point de départ et d'arrivée (liés à une agence).
- Une agence peut être point de départ ou d'arrivée de plusieurs trajets.



4 MODÈLE LOGIQUE DE DONNÉES (MLD)

-- Table des utilisateurs

```

CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    prenom VARCHAR(50) NOT NULL,
    nom VARCHAR(50) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    tel VARCHAR(20),
    
```

```

role ENUM('user', 'admin') DEFAULT 'user',
password VARCHAR(255) NOT NULL
);

-- Table des agences
CREATE TABLE agences (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(100) NOT NULL
);

-- Table des trajets
CREATE TABLE trajets (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    depart_id INT NOT NULL,
    arrivee_id INT NOT NULL,
    date_depart DATETIME NOT NULL,
    date_arrivee DATETIME NOT NULL,
    places_total INT NOT NULL,
    places_dispo INT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE,
    FOREIGN KEY (depart_id) REFERENCES agences(id),
    FOREIGN KEY (arrivee_id) REFERENCES agences(id)
);

```

5 INSTALLATION & LANCEMENT

Pré-requis

- WAMP / XAMPP avec PHP ≥ 8.0 et MySQL
- Git

Étapes

Cloner le projet

```
git clone https://github.com/Remi-13-exe/covoiturage
```

1.

Créer la base

```
mysql -u root -p < database/create_db.sql  
mysql -u root -p < database/seed_db.sql
```

2.

Configurer la connexion à la base

Modifier les identifiants dans `config.php`

```
define('DB_HOST', 'localhost');  
define('DB_NAME', 'covoitusage');  
define('DB_USER', 'root');  
define('DB_PASS', '');
```

3.

4. Lancer le serveur local

Ouvrir <http://localhost/covoitusage/>

5. Connexion test :

- o Voir comptes ci-dessous

6 COMPTES DE TEST

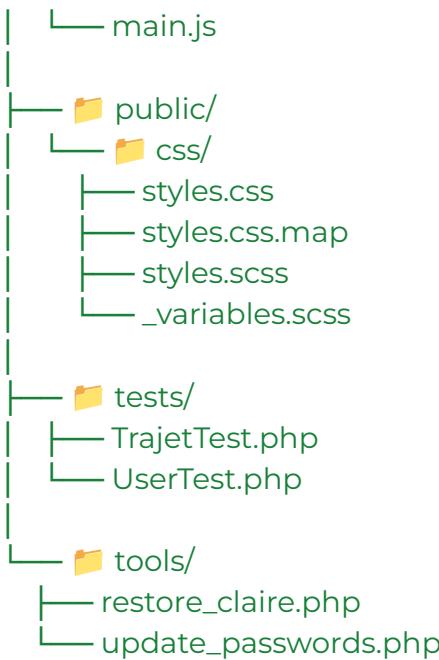
Type	Email	Mot de passe
Admin	admin@test.com	admin123
Utilisateur	alexandre.martin@email.fr	user123

7 CONTENU DU DÉPÔT

```
└── .htaccess  
└── composer.json  
└── composer.lock  
└── config.php  
└── create_user.php  
└── helpers.php  
└── index.php
```

```
├── package-lock.json
├── package.json
├── phpstan.neon
├── README.md
└── test_db.php

├── app/
│   ├── Controllers/
│   │   ├── AdminController.php
│   │   ├── TrajetController.php
│   │   └── UserController.php
│
│   ├── Models/
│   │   ├── Agence.php
│   │   ├── Trajet.php
│   │   └── User.php
│
│   └── Views/
│       ├── accueil.php
│       ├── admin_dashboard.php
│       ├── footer.php
│       ├── header.php
│       ├── login.php
│       ├── trajet_edit.php
│       └── trajet_form.php
│
└── assets/
    ├── MCD.png
    ├── MLD.txt
    └── jeu-d-essais/
        ├── agences.csv
        └── users.csv
    └── visuels/
        ├── accueil.png
        ├── details.png
        ├── header_admin.png
        ├── message_erreur.png
        └── visiteur.png
    └── database/
        ├── create_db.sql
        └── seed_db.sql
    └── js/
```



8 TESTS UNITAIRES (PHPUnit)

Des tests ont été implémentés pour vérifier :

- L'ajout d'un trajet (`TrajetTest.php`)
- L'ajout et suppression d'une agence (`test_agences.php`)
- L'enregistrement d'un utilisateur (`UserTest.php`)

Chaque test valide les écritures en base et la cohérence des données.

9 QUALITÉ & RÉUTILISABILITÉ

Le code a été structuré selon une **architecture MVC claire** :

- Les **Controllers** gèrent la logique métier,
- Les **Models** interagissent avec la base via PDO,

- Les **Views** utilisent Bootstrap pour un rendu uniforme.

Le projet a été documenté en **DocBlock**, analysé avec **PHPStan**, et conçu pour être **facilement réutilisable** dans d'autres projets internes (achats groupés, billetterie, etc.).

CAPTURE D'ECRAN DU TEST PHPSTAN :

```
PS C:\wamp64\www\covoiturage> vendor/bin/phpstan analyse --level=5
>>
Note: Using configuration file C:\wamp64\www\covoiturage\phpstan.neon.
17/17 [=====] 100%

[OK] No errors

.PS C:\wamp64\www\covoiturage> []
```