

DOSSIER DE PROJET : TOUCHE PAS AU KLAXON

Application de Covoiturage Intra-Entreprise

Étudiant : Tony Morieux

Contact : tmorieux@gmail.com

Date : Décembre 2025

Version : 1.0.0

1. CODE SOURCE & STRUCTURE

Dépôt GitHub : https://github.com/Toon-mo/Morieux_Tony_devoir_MVC_Touche_Pas_Au_Klaxon.git

L'application respecte l'architecture MVC et les normes PSR-4 avec la structure suivante :

MVC_TOUCHE_PAS_AU_KLAXON/

- |— config/ # Configuration (Base de données, Sécurité)
- |— public/ # Dossier accessible via le Web
 - | |— assets/ # Ressources statiques
 - | | |— jeu-d-essais/ # Fichiers .txt pour le seed
 - | | |— visuels/ # Documents et images du projet
 - | |— #Documents PDF
 - | |— scss/ # Fichiers sources SASS
 - | |— index.php # Point d'entrée unique (Front Controller)
- |— src/ # Cœur de l'application (Namespaces PSR-4)
 - | |— Controllers/ # Contrôleurs (Logique métier)
 - | | |— Models/ # Modèles (Accès aux données)
- |— tests/ # Tests unitaires PHPUnit
- |— vendor/ # Dépendances gérées par Composer
- |— views/ # Vues (Templates HTML/PHP)
- |— .env # Fichier de configuration locale (Ignoré par Git)
- |— .env.example # Modèle de configuration
- |— .gitignore # Fichiers à ignorer par Git
- |— .htaccess # Redirection des requêtes vers public/

- |— `composer.json` # Définition des dépendances et de l'autoload
 - |— `composer.lock` # Verrouillage des versions des dépendances
 - |— `database_schema.sql` # Script de création de la BDD
 - |— `phpunit.xml` # Configuration des tests unitaires
 - |— `README.md` # Documentation du projet
 - |— `seed.php` # Script d'alimentation de la base de données
-

2. INSTALLATION RAPIDE

Prérequis : PHP 8.0+, MySQL, Composer, Apache (mod_rewrite).

1. Cloner le projet : `git clone [URL_REPO]`
 2. Dépendances : Lancer `composer install` puis `composer dump-autoload`.
 3. Base de données :
 - Créer une BDD `touche_pas_au_klaxon`.
 - Importer le script SQL fourni (`database_schema.sql`).
 4. Configuration : Dupliquer `.env.example` en `.env` et configurer les accès BDD.
 5. Alimentation (Seed) : Lancer `php seed.php` ou accéder à `http://localhost/projet/seed.php`.
-

3. IDENTIFIANTS DE DÉMONSTRATION

Le script de seed génère les comptes suivants (Mot de passe : `password123`) :

Rôle	Email	Fonctionnalités clés
Administrateur	<code>alexandre.martin@email.fr</code>	Gestion Agences/Utilisateurs, Modération Trajets
Utilisateur	<code>sophie.dubois@email.fr</code>	Création/Modif/Suppression de ses trajets, Consultation

4. ARCHITECTURE TECHNIQUE

- Pattern : MVC (Modèle-Vue-Contrôleur) strict.

- Backend : PHP 8 Orienté Objet, PDO (Singleton), Composer.
 - Frontend : Bootstrap 5.3, SASS (variables personnalisées), JavaScript vanilla.
 - Base de données : MySQL/MariaDB (Moteur InnoDB, encodage utf8mb4).
 - Sécurité : Protections CSRF, XSS, Injections SQL, Hachage des mots de passe.
-

5. MODÉLISATION DES DONNÉES (MLD)

- AGENCE (<u>Id_Agence</u>, ville)
 - UTILISATEUR (<u>Id_Utilisateur</u>, nom, prenom, email, telephone, mot_de_passe, admin, #Id_Agence)
 - TRAJET (<u>Id_Trajet</u>, date_heure_depart, date_heure_arrivee, nb_places_total, nb_places_dispo, #Id_Conducteur, #Id_Agence_Depart, #Id_Agence_Arrivee)
-

6. FONCTIONNALITÉS IMPLÉMENTÉES

Partie Publique (Visiteur)

- Liste des trajets futurs avec places disponibles.
- Tri chronologique et indicateurs visuels (badges).
- Invitation à la connexion pour voir les détails.

Espace Utilisateur (Connecté)

- Dashboard : "Mon Espace" avec gestion des trajets.
- Détails : Modale avec coordonnées du conducteur (Email/Tél).
- CRUD Trajets :
 - Création avec contrôles de cohérence (Dates, Lieux, Places 1-8).
 - Modification et Suppression (uniquement ses propres trajets).
- Sécurité : Vérification stricte de l'auteur (isOwner).

Espace Administrateur

- Hub Admin : Tableau de bord centralisé.
- Gestion Agences : CRUD complet (Ajout/Modif/Suppr des villes).
- Gestion Utilisateurs : Liste, Ajout, Modification rôles, Suppression.

- **Modération Trajets : Vue globale et suppression de n'importe quel trajet.**
-

7. QUALITÉ & SÉCURITÉ

- **Sécurité :**
 - **Validation CSRF sur tous les formulaires (POST).**
 - **Échappement des sorties (htmlspecialchars).**
 - **Requêtes préparées PDO systématiques.**
 - **Protection des fichiers sensibles (.env, seed.php) via .htaccess.**
 - **Qualité Code :**
 - **Documentation complète (DocBlocks).**
 - **Structure PSR-4 et PSR-12.**
 - **Typage strict des méthodes.**
 - **Validation métier robuste ((PHP côté serveur).**
-

8. CONCLUSION

Le projet remplit l'intégralité du cahier des charges : une application fonctionnelle, sécurisée, avec une distinction claire des rôles (Utilisateur/Admin) et une architecture maintenable. Les contraintes techniques (SASS, PHPUnit, MVC) sont respectées.