📄 Documentation Technique – Projet EcoRide

1. Réflexion technologique initiale

Le projet EcoRide a été développé avec des technologies simples, robustes et accessibles :

- PHP 8 : utilisé côté serveur pour sa compatibilité avec de nombreux hébergements.

- MySQL (ou MariaDB) : pour la gestion de la base de données relationnelle.

- Bootstrap 5 : framework CSS facilitant le responsive design (desktop et mobile).

- JavaScript : utilisé pour les interactions dynamiques avec fetch (AJAX).

- XAMPP : serveur local regroupant Apache, PHP et MySQL.

- Docker & Docker Compose : pour conteneuriser et faciliter le déploiement de l'application.

2. Configuration de l'environnement de travail

- XAMPP : pour le développement local (Apache, PHP, MySQL).

- Visual Studio Code : éditeur avec extensions utiles (PHP Intelephense, PHP Server, etc.).

- phpMyAdmin : interface graphique pour administrer la base de données.

- Trello : gestion de projet selon la méthode Kanban.

- Git & GitHub : versionnement avec branches main, feature, dev.

- Docker Compose : utilisé pour lancer simultanément PHP, MySQL et phpMyAdmin.

3. Dynamisation de l'interface utilisateur (AJAX)

Une recherche dynamique de trajets est implémentée côté front-end avec JavaScript (fetch) :

- Lorsqu'un utilisateur remplit les champs de recherche, un appel AJAX est déclenché vers un script PHP (rechercher.php).

- Ce script retourne une réponse JSON, sans rechargement de la page, ce qui améliore l'expérience utilisateur.

📂 Fichiers concernés :

- assets/js/script.js : logique de fetch AJAX + rendu HTML.

- controllers/ajax_recherche_trajets.php : réception des paramètres + réponse JSON.

- views/index.php : structure du formulaire et affichage des résultats.

4. Modèle Conceptuel de Données (MCD)(voir mcd.pdf)

Entités principales :

- Utilisateur : id, nom, prenom, email, mot_de_passe, role, credits, photo

- Trajet : id, user_id, vehicule_id, depart, adresse_depart, destination, adresse_arrivee, date, duree_minutes, places, prix, statut, preferences, eco, fumeur

- Véhicule : id, user_id, marque, modele, energie, immatriculation, date_immatriculation

- Réservation : id, user_id, trajet_id, date_reservation

- Avis : id, utilisateur_id, trajet_id, note, commentaire, valide, created_at

- Confirmation : id, id_trajet, id_passager, valide, commentaire, statut, note, traite_par, date_validation

5. Diagramme de cas d'utilisation (voir diagramme uml.pdf)

Acteurs :

- Utilisateur (chauffeur et/ou passager)

- Employé

- Administrateur

Cas d'usage :

- S'inscrire / Se connecter

- Publier un trajet

- Rechercher / Réserver un trajet

- Annuler une réservation

- Confirmer ou signaler un trajet

- Laisser un avis

- Gérer les utilisateurs (admin)

- Valider les signalements (employé)

- Accéder aux statistiques (admin)

6. Diagramme de séquence – Réservation d'un trajet

1. L'utilisateur clique sur « Réserver »

2. Le système vérifie : son identité, ses crédits disponibles, la disponibilité du trajet.

3. La réservation est enregistrée.

4. Deux crédits sont débités automatiquement.

5. Une confirmation est affichée à l'utilisateur.

7. Déploiement de l'application

📌 Local (XAMPP) :

- Copier le projet dans htdocs

- Créer la base de données 'ecoride'

- Importer sql/ecoride_clean.sql via phpMyAdmin

- Accéder via : http://localhost/ecoride

📌 Docker Compose :

Fichier docker-compose.yml :

docker-compose up --build

Accès :

- http://localhost:8000 (application)

- http://localhost:8080 (phpMyAdmin, root/root, serveur : db)

8. Script SQL

Le fichier sql/ecoride_clean.sql contient :

- La création de toutes les tables (users, trajets, reservations, avis, confirmations, etc.)

- Les clés primaires et étrangères

- Des jeux de données de test (utilisateurs, trajets, véhicules, etc.)

9. Programmation Orientée Objet (POO)

Classe Trajet (models/Trajet.php) :

- Propriétés : id, depart, destination, prix, etc.

- Méthodes : __construct($data), resume()

Exemple :

```
foreach ($raw_trajets as $row) {

$trajets[] = new Trajet($row);

}
```

10. Maquettes & Intégration graphique

Les maquettes respectent un processus UX/UI avec :

- Wireframes (wireframe.pdf) : schéma structurel

- Mockups (mockups.pdf) : version finale avec couleurs et interactions

📍 Intégration visible dans :

- Recherche filtrée (views/index.php)

- Liste des trajets (cartes Bootstrap)

- Réservations (views/mes_reservations.php)

- Détail trajet (views/details.php)

11. Script JavaScript (AJAX)

Le fichier assets/js/script.js :

- Gère la soumission du formulaire de recherche

- Récupère les filtres (prix, note, durée, etc.)

- Envoie une requête AJAX à rechercher.php via fetch

- Reçoit une réponse JSON avec les trajets

- Affiche dynamiquement les résultats

Bonus :

- Icônes de chargement

- Affichage d'erreur

- Message si aucun trajet trouvé