Cahier des charges — Projet EcoRide

1. Présentation du projet

Nom du projet : EcoRide
Nature : Application web de covoiturage éco-responsable

Contexte : Face à l'urgence écologique et au coût croissant des transports, EcoRide vise à promouvoir le covoiturage en facilitant la mise en relation entre conducteurs et passagers.

Objectif principal : Créer une application web intuitive permettant la recherche, la réservation et la gestion de trajets en toute simplicité.

---

2. Objectifs et besoins

2.1 Objectifs fonctionnels

- Permettre aux utilisateurs de **chercher un trajet** selon une destination et une date.
- Mettre en relation conducteurs et passagers.
- Gérer les profils utilisateurs (conducteur / passager).
- Offrir un système de **réservation et de paiement par crédits internes**.
- Assurer une interface simple, responsive et accessible.

2.2 Objectifs non fonctionnels

- Accessibilité : site responsive (desktop, tablette, mobile).
- Performance : requêtes rapides (< 3 sec).
- Sécurité : gestion des sessions, données personnelles protégées.

---

3. Public cible

- Étudiants et jeunes actifs recherchant des trajets économiques.
- Salariés souhaitant réduire leur empreinte carbone en covoiturant.
  Personnes non motorisées ayant besoin de trajets ponctuels.

---

4. Périmètre fonctionnel

L'application proposera dans un premier temps :

1. **Page d’accueil** (présentation du service, recherche rapide).
2. **Moteur de recherche de trajets** (ville départ / arrivée, date).
3. **Page résultats** : liste des trajets disponibles avec détails.
4. **Fiches conducteurs** : profil, véhicule, avis.
5. **Système de réservation** avec gestion des crédits.
6. **Espace utilisateur** : gestion du compte, trajets, crédits.
7. **Système d’administration** (gestion utilisateurs / trajets).

---

5. Contraintes

- **Délais :** réalisation dans le cadre de l’ECF (durée définie par l’échéance).

- **Technologie imposée :**

  - **Front-end :**
    - HTML5, CSS3 obligatoires.
    - Respect des normes **W3C**.
    - Accessibilité minimale (alt sur images, structure sémantique).

  - **Back-end :**
    - PHP (procédural ou POO), sans framework lourd.
    - Connexion BDD sécurisée via **PDO** et requêtes préparées.
    - Sécurisation des mots de passe avec **password_hash()** / **password_verify()**.
    - Gestion des sessions et rôles utilisateurs (authentification).
    - Architecture MVC simplifiée ou séparation claire front / back.

  - **Base de données :**
    - MySQL relationnel obligatoire.
    - Normalisation jusqu’à la **3ème forme normale (3FN)**.
    - Script SQL de création et d’initialisation de la base livré.
    - Sauvegarde et restauration possibles.
    - (Optionnel) MongoDB pour logs/statistiques.

  - **Sécurité :**
    - Protection contre injection SQL, XSS, CSRF.
    - Validation côté serveur des formulaires.
    - Gestion sécurisée des sessions et cookies.

  - **Déploiement :**
    - Hébergement sur **Heroku / Fly.io ..**
    - Déploiement via Git (push → buildpack PHP/MySQL).
    - Base de données exportée/importée sur serveur.
    - URL publique accessible.

  - **Gestion de projet :**
    - Méthode Agile (Scrum / Kanban).

- Suivi avec **Trello** (Backlog → En cours → Terminé).
- Versioning avec **GitHub**.
- Documentation (cahier des charges, UML, MCD)..

Ressources logicielles

- **Environnement de développement :**
  - VS Code (éditeur de code, extensions PHP/MySQL/HTML/CSS).
  - XAMPP ou WAMP (serveur local Apache/PHP/MySQL).

- **Gestion de projet :**
  - Trello (organisation des user stories et tâches).
  - GitHub (gestion de version, suivi du code).

- **Conception & design :**
  - Figma (maquettes UI/UX).
  - draw.io (UML, MCD, diagrammes techniques).

- **Base de données :**
  - phpMyAdmin (gestion de MySQL en local et en production).

- **Déploiement :**
  - Heroku (hébergement cloud).
  - Fly.io (alternative performante et scalable)

---

6. Méthodologie

- **Gestion de projet Agile (Scrum)** : itérations courtes avec Trello pour suivre les user stories (Backlog → En cours → Terminé).
- **Maquettage** : Figma (UI) + draw.io (UML / MCD).
- **Versioning** : GitHub (branches dev / main).

---

7. Choix techniques

7.1 Front-end

- **HTML5 / CSS3 / Figma** : création de pages responsives, accessibles et rapides à développer.
- **Justification** : standard web, large compatibilité, Figma = gain de temps et cohérence graphique.

7.2 Back-end

- **PHP (API REST)** avec PDO pour la connexion à MySQL.
- **Justification** : PHP est demandé dans l'ECF, simple à déployer, PDO sécurise les requêtes SQL.

7.3 Base de données

- **MySQL** : base relationnelle pour gérer utilisateurs, trajets, réservations.
- **MongoDB (optionnel)** : gestion des logs et statistiques.
- **Justification** : MySQL pour cohérence forte (réservations, transactions), Mongo pour flexibilité.

7.4 Déploiement

- **Heroku** (hébergement cloud, intégration Git).
- **Alternative : Fly.io** (performances légères, scalable).
  Justification : solutions cloud gratuites adaptées à un projet étudiant.

7.5 Outils

- **VS Code** (développement).
- **phpMyAdmin** (gestion MySQL).
- **GitHub** (versioning).
- **Trello** (gestion Agile).
- **Figma & draw.io** (UI et UML/MCD).

---

8. Livrables

- Cahier des charges.
- Maquettes (Figma).
- Diagrammes UML + MCD (draw.io).
- Script SQL pour génération de la base.
- Code source (front + back + API).

- Documentation d'installation & d'utilisation.
- Application déployée sur Heroku/Fly.io.

---

9. Conclusion

EcoRide est une application web de covoiturage éco-responsable qui vise à promouvoir la mobilité durable. Grâce à une architecture simple mais évolutive, elle répondra aux besoins essentiels des utilisateurs tout en laissant la porte ouverte à des évolutions futures.