

NOM	KAHN
Prénom	Adrien
Date de naissance	29/01/1998

Copie à rendre

TP – Développeur Web et Web Mobile

Documents à compléter et à rendre

Lien du git : <https://github.com/Mirtyl-T/ecoride>

Lien de l'outil de gestion de projet :
<https://trello.com/invite/b/68d03245aff53a39f88e99c3/ATTI9152c6d57be53ab6c03c7e925fbe9360425BC8FD/kaban-ecoride>

Lien du déploiement : <https://ecoride-akahn-178db09ac932.herokuapp.com/>

Login et mot de passe administrateur :

admin@ecoride.com

MotDePasse123

SANS CES ELEMENTS, VOTRE COPIE SERA REJETEE

Partie 1 : Analyse des besoins

1. Effectuez un résumé du projet en français d'une longueur d'environ 20 lignes soit 200 à 250 mots

EcoRide est une plateforme de covoiturage écologique développée dans le cadre de la formation "Développeur Web et Web Mobile". Cette startup française vise à réduire l'impact environnemental des déplacements en encourageant le partage de trajets automobiles.

Le projet consiste à développer une application web complète permettant aux utilisateurs de proposer ou rechercher des covoiturages. La plateforme met l'accent sur l'aspect écologique en privilégiant les véhicules électriques, considérés comme des "voyages écologiques".

Les fonctionnalités principales incluent : une page d'accueil avec barre de recherche, un système de visualisation des covoiturages avec filtres (prix, durée, aspect écologique, note du conducteur), la gestion des comptes utilisateurs (passagers et chauffeurs), un système de crédits pour les paiements, et des espaces dédiés aux employés et administrateurs pour la modération.

L'architecture technique combine une base de données relationnelle et NoSQL, avec un développement full-stack incluant front-end (HTML5, CSS, JavaScript) et back-end (PHP/PDO). Le projet suit une méthodologie agile avec gestion Kanban et utilise Git avec des branches dédiées au développement et à la production.

Le déploiement doit être effectué sur des plateformes cloud, accompagné d'une documentation complète incluant maquettes, charte graphique, manuel d'utilisation et

documentation technique. Cette approche professionnelle prépare aux épreuves du titre professionnel.

2. Exprimez le cahier des charges, l'expression du besoin ou les spécifications fonctionnelles du projet

CAHIER DES CHARGES - PROJET ECORIDE

1. CONTEXTE ET OBJECTIFS

Porteur du projet : José, directeur technique de la startup EcoRide

Objectif principal : Développer une plateforme web de covoiturage écologique pour réduire l'impact environnemental des déplacements automobiles

Public cible : Voyageurs soucieux de l'environnement recherchant une solution économique

2. SPÉCIFICATIONS FONCTIONNELLES

2.1 Gestion des utilisateurs (US 7, 8)

- **Visiteurs :** Navigation libre, recherche de trajets
- **Utilisateurs :** Compte avec pseudo, email, mot de passe sécurisé, 20 crédits initiaux
- **Profils :** Passager, Chauffeur, ou les deux
- **Employés :** Modération des avis et gestion des litiges
- **Administrateurs :** Gestion globale, analytics, suspension de comptes

2.2 Interface utilisateur (US 1, 2)

- **Page d'accueil :** Présentation entreprise, barre de recherche, mentions légales
- **Menu navigation :** Accueil, covoiturages, connexion, contact
- **Charte graphique :** Thématique écologique obligatoire

2.3 Système de covoiturage (US 3, 4, 5, 6)

- **Recherche :** Par ville de départ/arrivée et date
- **Affichage trajets :** Pseudo/photo/note chauffeur, places disponibles, prix, horaires, aspect écologique
- **Filtres :** Écologique (véhicule électrique), prix maximum, durée, note minimale
- **Détails trajet :** Avis conducteur, modèle véhicule, préférences
- **Participation :** Validation double, gestion crédits, mise à jour places

2.4 Gestion chauffeur (US 8, 9, 11)

- Informations véhicule : Immatriculation, modèle, marque, couleur, nombre de places
- Préférences : Fumeur/non-fumeur, animaux autorisés, préférences personnalisées
- Création trajet : Adresses départ/arrivée, prix (commission 2 crédits), sélection véhicule
- Gestion trajet : Démarrage, arrêt, validation participants

2.5 Système économique

- Crédits : Monnaie virtuelle, 20 crédits à l'inscription
- Commission : 2 crédits prélevés par la plateforme par trajet
- Paiement : Validation double confirmation avant débit

2.6 Historique et évaluations (US 10, 11, 12)

- Historique : Trajets passés pour chauffeurs et passagers
- Annulations : Remboursement crédits, notification email
- Évaluations : Avis et notes post-trajet, validation employés
- Litiges : Signalement incidents, intervention employés

2.7 Back-office (US 12, 13)

- Espace employé : Validation avis, gestion litiges
- Espace administrateur : Création comptes employés, analytics (covoiturations/jour, revenus), suspension comptes

3. CONTRAINTES TECHNIQUES

- Bases de données : Relationnelle (MySQL/PostgreSQL) + NoSQL (MongoDB)
- Architecture : Front-end (HTML5, CSS, JavaScript) + Back-end (PHP/PDO)
- Sécurité : Mots de passe sécurisés, gestion sessions, validation données
- Déploiement : Obligatoire sur plateforme cloud (Heroku, Azure, Vercel)

4. LIVRABLES ATTENDUS

- Code source GitHub avec bonnes pratiques Git
- Application déployée fonctionnelle
- Documentation technique complète
- Maquettes wireframes et mockups (desktop + mobile)
- Manuel utilisateur avec identifiants de test
- Charte graphique et gestion de projet Kanban

Partie 2 : Spécifications technique

1. **Spécifiez les technologies que vous avez utilisé en justifiant les conditions d'utilisation et pourquoi le choix de ses éléments**

Réflexions Technologiques Initiales

Stack

Technologique

Choisie Backend :

Symfony (PHP)

Justifications du

choix :

1. Maturité et Robustesse
 - Framework PHP mature avec une architecture MVC solide
 - Ecosystem riche avec de nombreux bundles
 - Performance optimisée pour les applications web
2. Productivité
 - ORM Doctrine intégré
 - Système de routing flexible
 - Validation et sérialisation intégrées
3. Sécurité
 - Composant Security robuste
 - Protection CSRF intégrée
 - Gestion des sessions sécurisées

- Chiffrement et hashage des mots de passe

4. Maintenabilité

- Architecture modulaire
- Injection de dépendances
- Tests unitaires et fonctionnels facilités
- Documentation extensive

Base de Données :

Approche Hybride

MongoDB +

Doctrine ODM :

- Flexibilité pour les données de review
- Scalabilité horizontale

Justification de l'approche hybride :

- MongoDB pour les données flexibles
- Potentiel MySQL/PostgreSQL pour les données relationnelles strictes

Frontend : Twig + SCSS + JavaScript

Twig (Templating) :

- Intégration native avec Symfony
- Syntaxe claire et sécurisée
- Héritage de templates efficace

SCSS (Styling) :

- Organisation modulaire des styles
- Compatibilité avec Bootstrap

Bootstrap + JavaScript :

- Framework CSS responsive
- Composants UI prêts à l'emploi
- JavaScript vanilla pour les interactions

Outils de Build : Webpack Encore

Avantages :

- Intégration native Symfony
- Compilation automatique des assets
- Hot reload en développement
- Optimisation pour la production

2. Comment avez-vous mis en place votre environnement de travail ? Justifiez vos choix. (README.md)

MISE EN PLACE DE L'ENVIRONNEMENT DE TRAVAIL

1. CHOIX TECHNOLOGIQUES JUSTIFIÉS

Backend - Symfony 6.x

Justification :

- Framework PHP mature : Symfony offre une architecture MVC robuste avec des composants réutilisables
- Écosystème riche : Doctrine ORM pour la base relationnelle, bundles MongoDB pour NoSQL
- Sécurité intégrée : Gestion des sessions, CSRF, validation des données
- Performance : Cache optimisé, lazy loading
- Répondre aux exigences : Compatible avec les spécifications PHP/PDO du cahier des charges

Frontend - Bootstrap + Webpack Encore

Justification :

- Bootstrap : Framework CSS responsive, accélère le développement d'interfaces mobiles
- Webpack Encore : Bundling moderne des assets, compilation SCSS, optimisation automatique
- Compatibilité : Respect des contraintes HTML5/CSS/JavaScript du projet

Bases de données - Approche hybride

Justification :

- Doctrine ORM : Gestion relationnelle pour les données structurées (utilisateurs, trajets)
- MongoDB ODM : Base NoSQL pour données flexibles (avis, préférences personnalisées)
- Double contrainte : Répond à l'exigence obligatoire d'utiliser les deux types de BD

2. ENVIRONNEMENT DE DÉVELOPPEMENT

Prérequis techniques

PHP ≥ 8.1 → Performances et fonctionnalités modernes

Composer → Gestionnaire de dépendances PHP

Node.js → Compilation assets frontend

Git → Versionning avec bonnes pratiques

Architecture de déploiement

- Docker : Conteneurisation avec **compose.yaml** pour portabilité

- CI/CD : Actions GitHub pour intégration continue
- Serveur local : Symfony CLI ou serveur PHP intégré

3. WORKFLOW DE DÉVELOPPEMENT

Gestion des dépendances

bash

```
composer install      # Backend PHP
```

```
npm install          # Frontend JavaScript
```

```
composer require mongodb # Extension NoSQL
```

Compilation des assets

bash

```
npm run dev          # Développement avec watch
```

```
npm run build         # Production optimisée
```

Base de données

bash

```
doctrine:database:create      # Création BDD relationnelle
```

```
doctrine:migrations:migrate  # Migrations schema
```

```
doctrine:mongodb:schema:create # Schema MongoDB
```

```
doctrine:fixtures:load        # Données de test
```

4. JUSTIFICATIONS STRATÉGIQUES

Pourquoi Symfony ?

- Productivité : Générateurs automatiques (controllers, entités)
- Maintenabilité : Architecture claire, séparation des responsabilités
- Évolutivité : Composants modulaires, extensibilité
- Documentation : Ecosystem bien documenté, communauté active

Pourquoi cette stack ?

- Cohérence : Stack PHP moderne compatible avec les compétences DWWMM
- Performance : Webpack pour optimisation frontend, Doctrine pour requêtes optimisées
- Sécurité : Guards Symfony, validation automatique, protection CSRF

- Déploiement : Compatible cloud (Heroku, Azure) via Docker

5. AVANTAGES DE CET ENVIRONNEMENT

Respect du cahier des charges : Toutes les contraintes techniques respectées

Développement rapide : Outils modernes, génération automatique

Qualité du code : Standards PSR, tests automatisés

Collaboration : Git workflow, environnement reproductible

Déployabilité : Configuration Docker, variables d'environnement

Cette configuration offre un environnement professionnel moderne tout en respectant les contraintes pédagogiques du projet DWWM, facilitant le développement d'une application web complète et déployable.

3. Énumérez les mécanismes de sécurité que vous avez mis en place, aussi bien sur vos formulaires que sur les composants front-end ainsi que back-end.

Sécurité

-

Composant

Security

robuste •

Protection

CSRF


intégrée

- Gestion des sessions sécurisées
- Chiffrement et hashage des mots de passe

4. Décrivez une veille technologique que vous avez effectuée, sur les vulnérabilités de sécurité.

Dans le cadre de mes projets web, j'ai effectué une veille sur les bonnes pratiques de gestion des sessions sécurisées. J'ai consulté les recommandations OWASP Session Management et les guides de sécurité PHP/Symfony. Les points clés identifiés incluent l'utilisation de tokens CSRF pour éviter les attaques cross-site, la régénération automatique des ID de session après authentification, et la configuration des cookies avec les flags `HttpOnly` et `Secure`. J'ai également étudié les techniques de fixation de session et les moyens de les prévenir. Cette veille m'a conduit à implémenter des timeouts de session appropriés, à sécuriser le stockage des données de session côté serveur, et à utiliser HTTPS obligatoirement pour les cookies sensibles. Les ressources principales ont été la documentation

Symfony Security, les bulletins CERT-FR, et quelques articles techniques sur les vulnérabilités de session hijacking



1. Décrivez une situation de travail ayant nécessité une recherche durant le projet à partir de site anglophone. N'oubliez pas de citer la source

Recherche sur la documentation Bootstrap pour l'intégration responsive

Lors du développement de l'interface utilisateur de mon projet EcoRide, j'ai rencontré des difficultés pour implémenter correctement le système de grille responsive de Bootstrap. Les éléments ne s'affichaient pas correctement sur mobile et tablette. J'ai dû consulter la documentation officielle anglophone de Bootstrap sur

<https://getbootstrap.com/docs/5.3/getting-started/introduction/> pour comprendre la syntaxe des classes de colonnes et les breakpoints responsive.

La section "Grid system" m'a permis de comprendre l'utilisation des classes `col-sm-`, `col-md-`, `col-lg-` et l'importance des conteneurs `.container` et `.container-fluid`. J'ai également étudié la partie sur les "Utilities" pour les espacements (`m-`, `p-`) et l'alignement. Cette recherche m'a conduit à restructurer mon HTML avec les bonnes classes Bootstrap, notamment pour créer un layout adaptatif pour les cartes de trajets et le formulaire d'inscription. La documentation m'a aussi aidé à implémenter correctement les composants navbar et modal avec leurs classes JavaScript associées.

Source : Bootstrap Documentation. (2023). *Introduction - Bootstrap v5.3*.

<https://getbootstrap.com/docs/5.3/getting-started/introduction/>

2. Mentionnez l'extrait du site anglophone qui vous a aidé dans la question précédente en effectuant une traduction en français.

Extrait du site Bootstrap qui a aidé pour la configuration de l'environnement :

"Navbars require a wrapping `.navbar` with `.navbar-expand{-sm|-md|-lg|-xl|-xxl}` for responsive collapsing and color scheme classes. Navbars and their contents are fluid by default. Use our spacing and flex utility classes for controlling spacing and alignment within navbars. Navbars are responsive by default, but you can easily modify them to change that."

EcoRide - Covoiturage éco-responsable pour une mobilité verte

Traduction française :

"Les barres de navigation nécessitent un conteneur `.navbar` avec `.navbar-expand{-sm|-md|-lg|-xl|-xxl}` pour le repliement adaptatif et les classes de jeux de couleurs. Les barres de navigation et leur contenu sont fluides par défaut. Utilisez nos classes utilitaires d'espacement et de flexbox pour contrôler l'espacement et l'alignement dans les barres de navigation. Les barres de navigation sont adaptatives par défaut, mais vous pouvez facilement les modifier pour changer ce comportement."

Cette documentation Bootstrap a été essentielle pour justifier l'utilisation de Bootstrap dans le stack technique d'EcoRide, notamment pour la création d'interfaces utilisateur responsives et la gestion efficace de la navigation mobile - des éléments cruciaux pour respecter les spécifications du projet.

<https://getbootstrap.com/docs/5.3/components/navbar/>

Partie 4 : Informations complémentaire

Autres ressources

Documentation technique utilisée

- **Symfony Documentation** : Guide officiel pour l'architecture MVC et l'ORM Doctrine
 - <https://symfony.com/doc/current/index.html>
- **MongoDB PHP Extension** : Documentation pour l'intégration NoSQL
 - <https://www.mongodb.com/docs/php-library/master/>
- **Doctrine ODM** : Mapping objet-document pour MongoDB
 - <https://www.doctrine-project.org/projects/doctrine-mongodb-odm/en/2.4/index.html>
- **Webpack Encore** : Configuration et compilation des assets
 - <https://symfony.com/doc/current/frontend.html>

Outils de développement

- **GitHub** : Versionning et collaboration - <https://github.com/Mirtyl-T/ecoride>
- **Trello** : Gestion de projet Kanban - <https://trello.com/invite/b/68d03245aff53a39f88e99c3/ATTI9152c6d57be53ab6c03c7e925fbe9360425BC8FD/kaban-ecoride>
- **Heroku** : Plateforme de déploiement cloud - <https://ecoride-akahn-178db09ac932.herokuapp.com/>

- **Composer** : Gestionnaire de dépendances PHP
- **npm/Yarn** : Gestionnaire de dépendances JavaScript

Ressources d'apprentissage

- **OWASP** : Bonnes pratiques de sécurité web
 - <https://owasp.org/www-project-top-ten/>
- **MDN Web Docs** : Documentation JavaScript et CSS
- **PHP Standards** : PSR pour les conventions de code PHP

Librairies et frameworks utilisés

- **Symfony 6.x** : Framework PHP principal
- **Bootstrap 5.3** : Framework CSS responsive
- **Doctrine ORM/ODM** : Mapping base de données relationnelle et NoSQL
- **Twig** : Moteur de templates
- **PHPUnit** : Tests unitaires

Informations complémentaires

Méthodologie de développement

Le projet EcoRide a été développé selon une approche agile avec une gestion Kanban via Trello. L'organisation du travail s'est articulée autour de user stories clairement définies, permettant un développement itératif et incrémental.

Architecture technique détaillée

L'application suit une architecture hybride combinant :

- **Base relationnelle** : Gestion des utilisateurs, trajets et transactions via Doctrine ORM
- **Base NoSQL MongoDB** : Stockage des avis, préférences personnalisées et données flexibles
- **Frontend responsive** : Bootstrap pour l'interface utilisateur adaptative
- **Sécurité renforcée** : Implémentation des standards Symfony Security

Déploiement et CI/CD

Le projet est configuré pour un déploiement automatisé sur Heroku avec :

- Variables d'environnement sécurisées
- Base de données cloud configurée
- Compilation automatique des assets
- Configuration Docker pour la portabilité

Défis techniques rencontrés et solutions

1. **Intégration hybride des bases de données** : Mise en place d'une architecture permettant l'utilisation simultanée de MySQL et MongoDB
2. **Sécurisation des transactions** : Implémentation d'un système de double validation pour les paiements par crédits
3. **Interface responsive** : Adaptation de l'interface pour tous les types d'appareils avec Bootstrap

Perspectives d'évolution

- Intégration d'une API de géolocalisation pour le calcul automatique des distances
- Système de notification en temps réel
- Application mobile native
- API RESTful pour l'intégration avec des services tiers

Respect du cahier des charges

Le projet répond intégralement aux spécifications demandées :

- Double base de données (relationnelle + NoSQL)
- Interface responsive et accessible
- Système de gestion des utilisateurs complet
- Fonctionnalités de covoiturage avec filtres avancés
- Back-office administrateur et employé
- Déploiement cloud opérationnel
- Documentation technique complète

Identifiants de test

Administrateur :

- Email : admin@ecoride.com
- Mot de passe : MotDePasse123

Utilisateur test :

- Comptes générés via fixtures Doctrine
- 20 crédits initiaux par défaut

Maintenance et support

Le code source est maintenu sur GitHub avec des bonnes pratiques Git :

- Branches distinctes pour développement et production
- Commits atomiques et messages descriptifs
- Documentation inline et README détaillé

Cette approche professionnelle garantit la maintenabilité et l'évolutivité de la plateforme EcoRide.

1.

