Documentation technique

Réflexions initiale technologique

Le langage PHP a été utilisé pour gérer la partie backend de l'application. Ce choix s'explique par la facilité de déploiement de PHP sur la plupart des serveurs web, sa courbe d'apprentissage relativement accessible, ainsi que son intégration naturelle avec les bases de données MySQL. PHP a permis d'implémenter les traitements liés aux utilisateurs, aux trajets, aux réservations, ainsi que la logique de gestion des sessions, des crédits et des règles métiers spécifiques (validation, annulation, etc.).

Pour le stockage des données, c'est une base de données MySQL qui a été choisie. MySQL est une solution relationnelle fiable, bien documentée, gratuite, et parfaitement adaptée aux applications web structurées. Elle permet de gérer efficacement les relations entre utilisateurs, trajets et réservations, tout en garantissant l'intégrité des données.

Du côté client, l'interface est construite en HTML, CSS et JavaScript. HTML a servi à structurer les pages et les formulaires, tandis que CSS a été utilisé pour styliser les éléments de manière responsive. Le choix d'un design responsive permet d'offrir une expérience fluide, quel que soit l'appareil utilisé (ordinateur, smartphone, tablette). Aucune bibliothèque CSS externe n'a été nécessaire, ce qui a permis de conserver une maîtrise complète du rendu visuel.

Enfin, JavaScript a été employé pour dynamiser certaines parties de l'interface. Des appels fetch() ont été mis en place pour interagir avec le serveur sans recharger entièrement la page, notamment pour la mise à jour du profil utilisateur ou la participation à un trajet. Ce choix permet une meilleure réactivité de l'application et améliore l'expérience utilisateur.

Configuration de l'environnement de travail

Pour organiser efficacement le développement du projet Ecoride, j'ai mis en place un environnement de travail simple mais structuré, adapté à mes besoins et à mes connaissances au moment de la conception. Mon objectif était de garantir une bonne lisibilité du code, une facilité de test, et une mise en route rapide du projet.

Le projet est hébergé localement à l'aide du serveur Apache via XAMPP. Ce choix s'est imposé car XAMPP permet de disposer d'un environnement complet (serveur web, PHP, MySQL) prêt à l'emploi, sans configuration complexe. Cela m'a permis de me concentrer sur le développement fonctionnel sans me soucier de l'installation séparée des différents services. La base de données a été gérée via phpMyAdmin, qui facilite la visualisation des tables, des relations, et les tests de requêtes SQL en phase de développement.

Le projet est structuré de manière modulaire, avec une séparation claire des responsabilités. Le dossier *class/* contient les classes représentant la logique métier, comme Reservation ou Travel, qui regroupent les méthodes de manipulation des données. Le dossier *controllers/* sert de point d'entrée pour chaque page : il initialise la session, inclut les dépendances nécessaires, et détermine dynamiquement quel contenu HTML charger via un système de gabarit (layout.php). Ce fonctionnement permet une structure flexible et cohérente sans reposer sur un modèle MVC strict.

Le dossier *back/* regroupe les traitements côté serveur, en particulier ceux liés aux actions des utilisateurs (création de trajet, validation, annulation, etc.). Les interfaces sont organisées dans le dossier *templates/,* divisé en *pages/* pour les écrans complets, et

components/ pour les éléments réutilisables comme les en-têtes et les pieds de page ainsi que pour les listes ou les popups, qui ne sont pas toujours partagés entre plusieurs pages mais permettent de structurer finement l'interface en évitant que les fichiers de page deviennent trop volumineux. Le dossier *css/* contient les feuilles de style personnalisées, *script/* les fichiers JavaScript liés aux interactions dynamiques (comme les appels fetch()), et *icons/* qui contient les icones de l'application ou *photos/* qui regroupe les photos des utilisateurs.

Pour documenter le projet, j'ai créé un fichier README.md à la racine du dossier. Ce fichier contient les instructions complètes pour cloner le projet, le placer dans un environnement local (comme XAMPP), configurer la base de données, importer les fichiers SQL (structure et données de test), et démarrer l'application. Il précise également les prérequis techniques (PHP 8+, MySQL, navigateur moderne, etc.), décrit les comptes de test disponibles, et sert de point d'entrée pour toute personne souhaitant comprendre, installer ou tester l'application Ecoride.

Enfin, j'ai utilisé un éditeur de code comme VS Code, avec des extensions utiles (coloration syntaxique PHP, validation HTML/CSS, auto-formatage), afin de travailler plus confortablement et avec cohérence. Ces choix techniques ont été faits dans une logique de simplicité, d'accessibilité et de bonne pratique en développement web.

Modèle conceptuel de données (MCD)

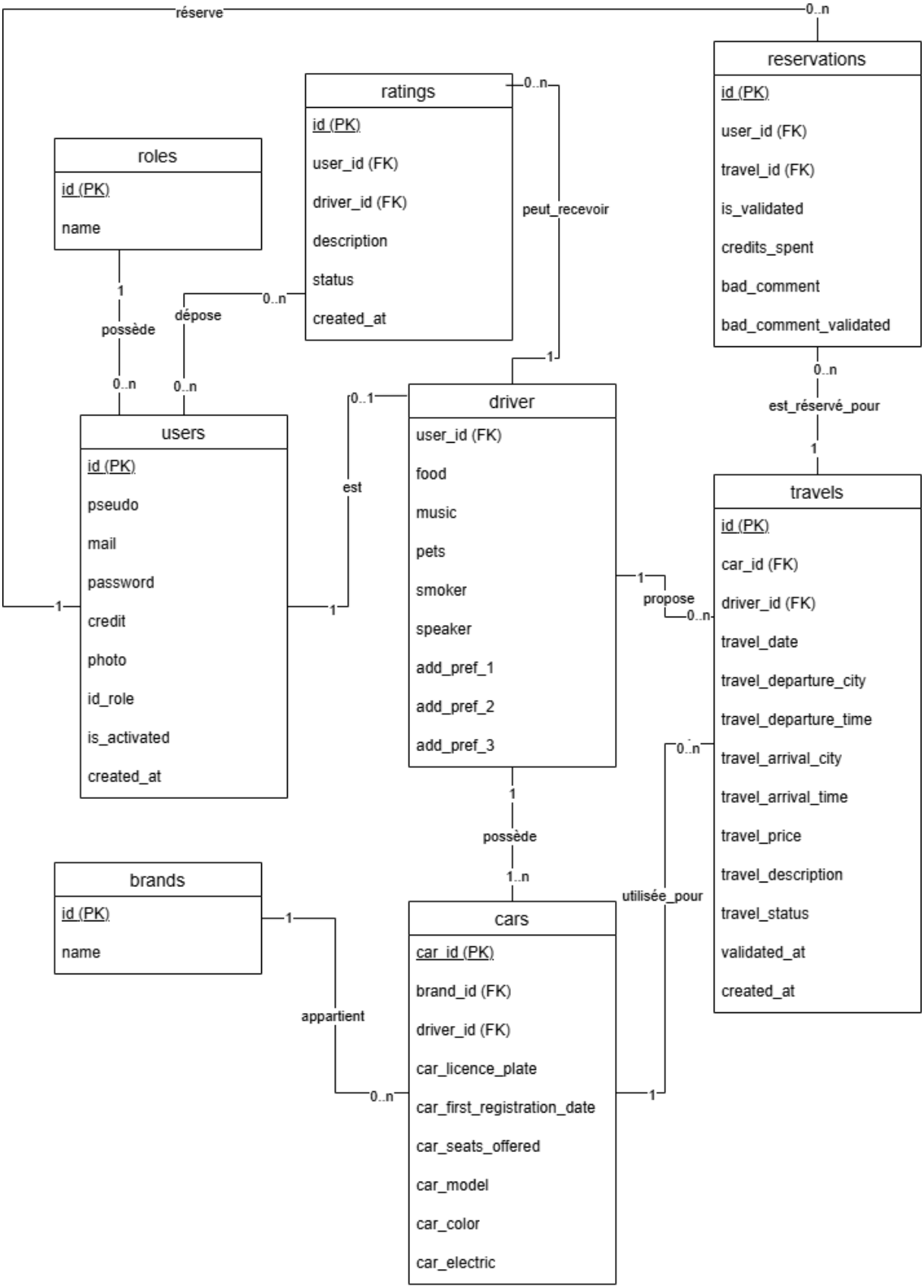


Diagramme d’utilisation

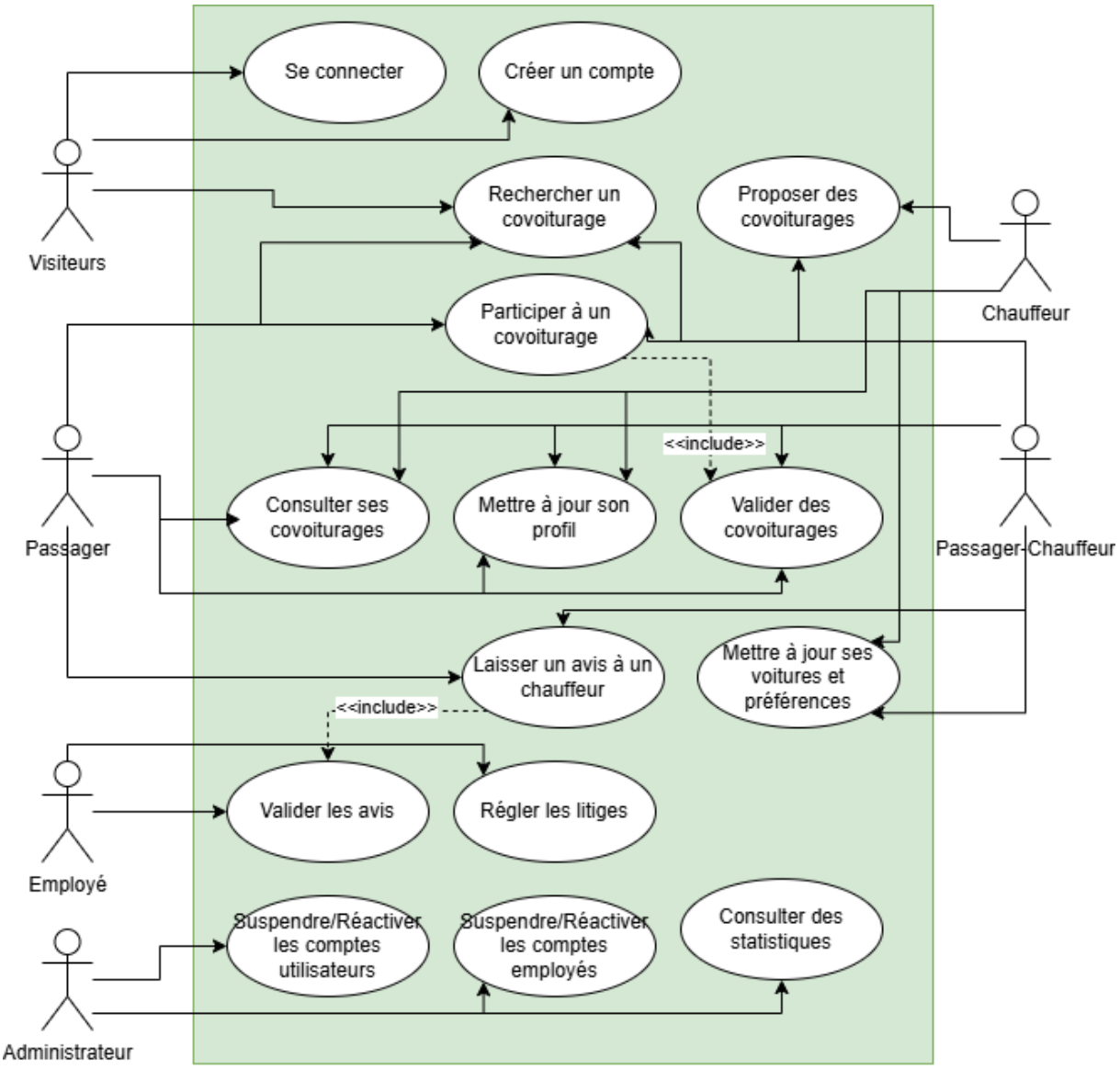
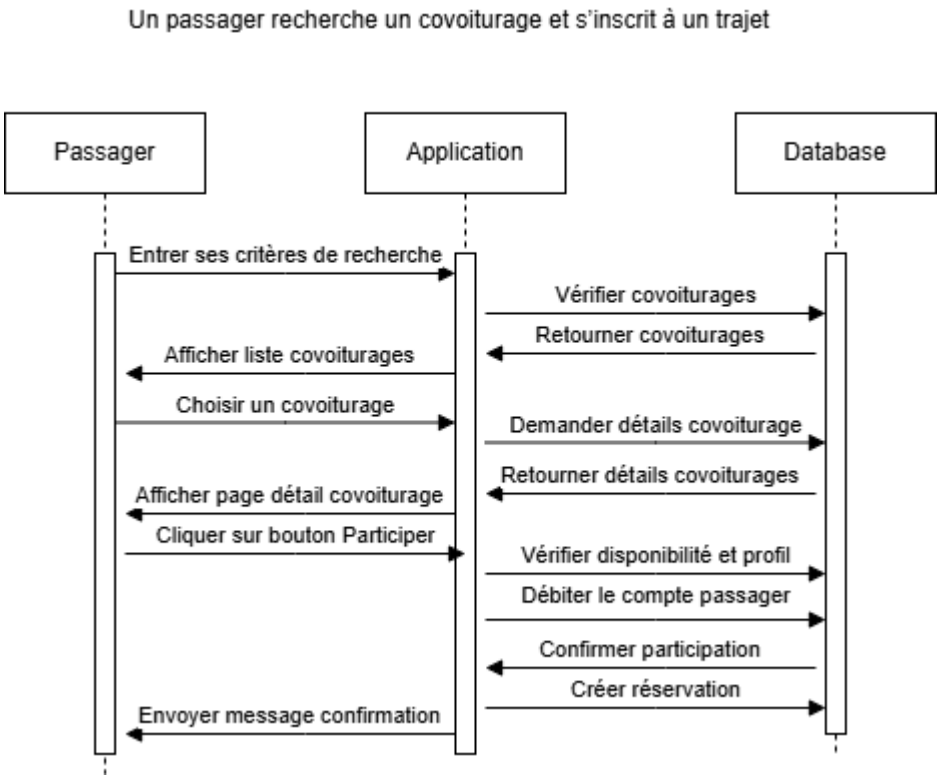


Diagramme de séquence



Déploiement de l’application

Pour déployer mon application Ecoride, j’ai d’abord travaillé en local, puis je l’ai mise en ligne sur la plateforme AlwaysData. Le but de cette étape était de rendre l’application accessible sur Internet, tout en permettant de gérer facilement la transition entre l’environnement de développement et l’environnement de production.

Gestion des environnements (local/production)

Afin que mon application fonctionne correctement en local et en production, j’ai mis en place une gestion conditionnelle des paramètres de configuration selon l’environnement.

a. Connexion à la base de données

J’ai mis en place une détection automatique de l’environnement via la variable $_SERVER['HTTP_HOST'], afin d’utiliser des paramètres adaptés selon que l’application tourne en local ou en production.

Cela m’a permis de conserver un seul fichier de configuration (database.php) tout en gérant deux bases différentes.

```php
$host = $_SERVER['HTTP_HOST'];

if (in_array(needle: $host, haystack: ['localhost', '127.0.0.1', '::1'])) {
    // Local connection (XAMPP)
    $dbHost = 'localhost';
    $dbName = 'ecoride';
    $dbPort = 3308;
    $dbUser = 'root';
    $dbPass = '';
} else {
    // AlwaysData connection (production)
    $dbHost = 'mysql-ecoride-sge.alwaysdata.net';
    $dbName = 'ecoride-sge_ecoride';
    $dbPort = 3306; // par défaut
    $dbUser = '411431';
    $dbPass = 'Germain14!';
}

try {
    $pdo = new PDO(
        dsn: "mysql:host=$dbHost;port=$dbPort;dbname=$dbName;charset=utf8mb4",
        username: $dbUser,
        password: $dbPass
    );
    $pdo->setAttribute(attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    die("Erreur de connexion à la base de données : " . $e->getMessage());
}
```

Fichier database.php

b. Configuration de BASE_URL

Pour que les fichiers statiques (icônes, scripts…) soient bien chargés selon l'environnement, j'ai également défini la constante BASE_URL de manière dynamique

Cela m'a permis d'éviter les erreurs de chargement, notamment pour les icônes et liens internes.

Déploiement sur AlwaysData

J'ai choisi AlwaysData comme hébergeur pour plusieurs raisons :

- Offre gratuite suffisante pour un projet étudiant
- Simplicité d'utilisation
- Support PHP et bases de données MySQL
- URL en alwaysdata.net utilisable directement

J'ai suivi les étapes suivantes :

1. J'ai créé un compte sur alwaysdata.com
2. J'ai ajouté une application web PHP dans le panneau d'administration
3. J'ai choisi un sous-domaine : ecoride-sge.alwaysdata.net
4. J'ai créé une base de données MySQL dans l'interface

Pour créer ma base de données MySQL, j'ai suivi les étapes suivantes :

1. Dans la section Bases de données/MySQL, j'ai ajouté une base de données nommée « ecoride-sge_ecoride »
2. Depuis mon environnement local, j'ai exporté la structure de mes tables sous forme de fichier .sql
3. Je l'ai importée dans la base de production via phpMyAdmin
4. J'ai ensuite configuré les paramètres de connexion à cette base dans mon fichier database.php (comme expliqué plus haut).

Transfert des fichiers avec FileZilla

Pour envoyer mes fichiers sur le serveur, j'ai utilisé **FileZilla** :

1. Je me suis connecté avec les identifiants FTP fournis par AlwaysData.
2. J'ai transféré les fichiers dans le dossier /www/.
3. J'ai vérifié l'arborescence : index.php, css/, back/, etc.
4. J'ai accédé au site via https://ecoride-sge.alwaysdata.net pour vérifier que tout s'affichait correctement.

Tests après déploiement

Une fois l'application en ligne, j'ai réalisé plusieurs tests pour m'assurer du bon fonctionnement :

- Connexion / inscription
- Recherche de covoiturage
- Participation à un trajet

- Création d’un covoiturage
- Chargement des images et icônes (vérification de BASE_URL)