# Sommaire

[1. Contexte du projet](#_heading=h.1d6mma2ci7xl)

[1.1. Présentation du projet](#_heading=h.wek20lqal41w)

[1.2. Date de rendu du projet](#_heading=h.hsbf2s9v0gpe)

[2. Besoins fonctionnels](#_heading=h.72adhlswac0n)

[3. Ressources nécessaires à la réalisation du projet](#_heading=h.lr47i5krzqwp)

[3.1. Ressources matérielles](#_heading=h.moe5ly27tmhr)

[3.2. Ressources logicielles](#_heading=h.cvyk6p742phf)

[4. Gestion du projet](#_heading=h.wj6e27o1xw4s)

[5. Conception du projet](#_heading=h.4jjvlmxvhcof)

[5.1. Le front-end](#_heading=h.b3pf1ig00kap)

[5.1.1. Wireframes](#_heading=h.xbdoysf41ddh)

[5.1.2. Maquettes](#_heading=h.taj1xh3jo3sr)

[5.1.3. Arborescences](#_heading=h.7cpah6l43vkf)

[5.2. Le back-end](#_heading=h.ri4n2nlk4x5y)

[5.2.1. Diagramme de cas d’utilisation](#_heading=h.4hfnlk8rv9if)

[5.2.2. Diagramme d’activités](#_heading=h.d1ay6n1m7wn9)

[5.2.3. Modèles Conceptuel de Données (MCD)](#_heading=h.mhbyfyrbmn9)

[5.2.4. Modèle Logique de Données (MLD)](#_heading=h.rr2kq3n12ss)

[5.2.5. Modèle Physique de Données (MPD)](#_heading=h.5jowy0y259aj)

[6. Technologies utilisées](#_heading=h.6j2nbpnxpppw)

[6.1. Langages de développement Web](#_heading=h.uhwlwqptkq2s)

[6.2. Base de données](#_heading=h.q8i476rr9oys)

[7. Sécurité](#_heading=h.15p9a6rgigad)

[7.1. Login et protection des pages administrateurs](#_heading=h.q04sq9cnrw82)

[7.2. Hachages des mots de passe avec Bcrypt](#_heading=h.ct1wz6q0jyd1)

[7.3. Protection contre les attaques XSS (Cross-Site Scripting)](#_heading=h.cimahn54kest)

[7.4. Protection contre les injections SQL](#_heading=h.cg22ajiezi4v)

# 

# Contexte du projet

## 1.1. Présentation du projet

Votre agence web a été sélectionnée par le comité d’organisation des jeux olympiques de Los Angeles 2028 pour développer une application web permettant aux organisateurs, aux médias et aux spectateurs de consulter des informations sur les sports, les calendriers des épreuves et les résultats des JO 2028.

Votre équipe et vous-même avez pour mission de proposer une solution qui répondra à la demande du client.

## 1.2. Date de rendu du projet

Le projet doit être rendu au plus tard le 18/12/2025.

# Besoins fonctionnels

Le site web devra avoir une partie accessible au public et une partie privée permettant de gérer les données.

Les données seront stockées dans une base de données relationnelle pour faciliter la gestion et la mise à jour des informations. Ces données peuvent être gérées directement via le site web à travers un espace administrateur.

# Ressources nécessaires à la réalisation du projet

**REPRENDRE RÉPONSES MISSION 1**

## 3.1. Ressources matérielles

* PC
* Connexion internet
* Serveur local
* Périphérique de travail

## 3.2. Ressources logicielles

* Visual studio code
* Php
* Mysql
* Mamp
* Mocodo
* Java-script
* Html-CSS

# Gestion du projet

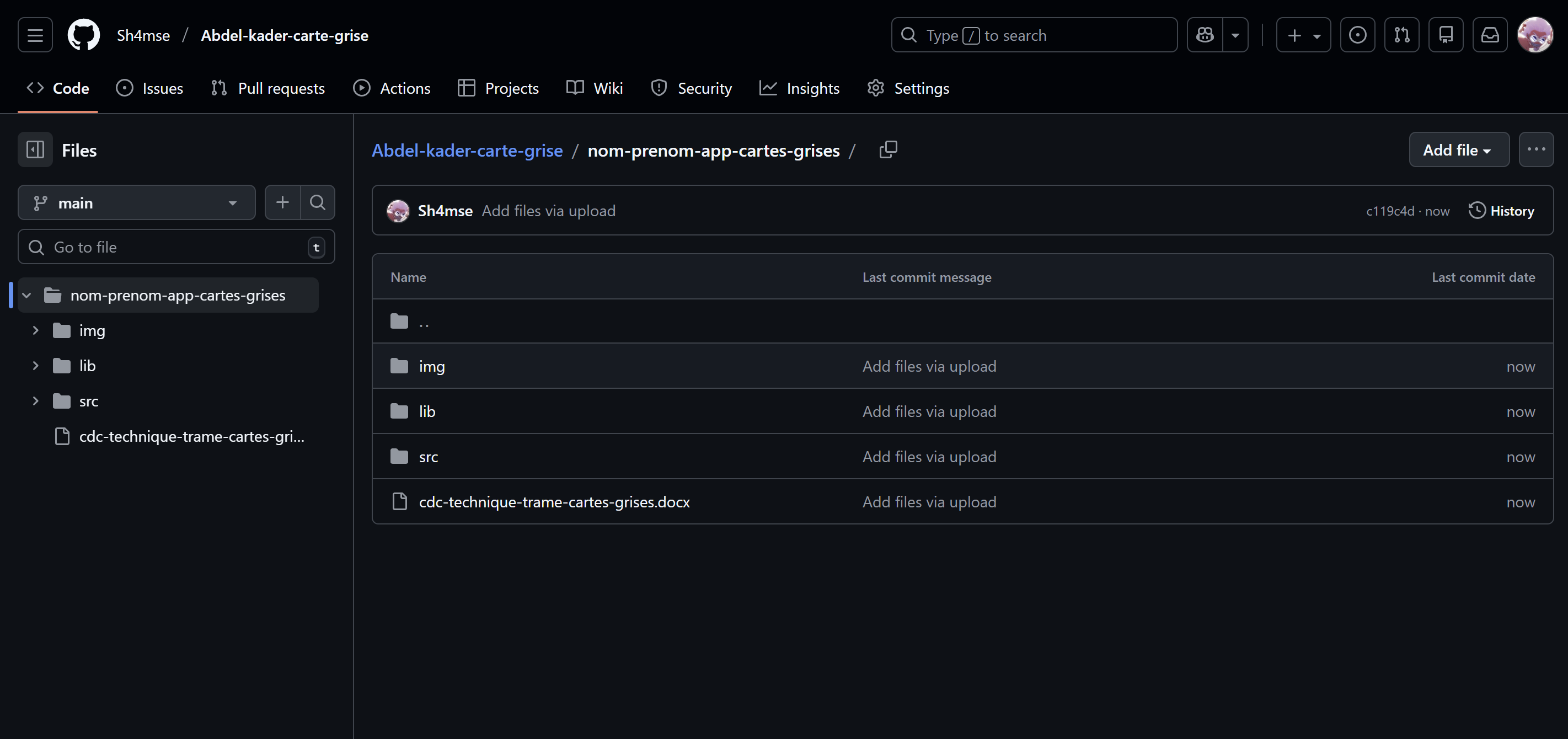
Pour réaliser le projet, nous utiliserons la méthode Agile Kanban. Nous utiliserons également l’outil de gestion de projet en ligne Trello.

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

**INSÉREZ UNE CAPTURE D'ÉCRAN DE VOTRE TRELLO**

Nous travaillons également sur GitHub, plateforme de développement collaboratif.



# Conception du projet

## 5.1. Le front-end

**REPRENDRE RÉPONSES MISSION 4**

### Une image contenant texte, capture d’écran, ligne, diagramme Le contenu généré par l’IA peut être incorrect.Une image contenant texte, capture d’écran, nombre, Police Le contenu généré par l’IA peut être incorrect.5.1.1. Wireframes

Une image contenant texte, capture d’écran, Police, diagramme

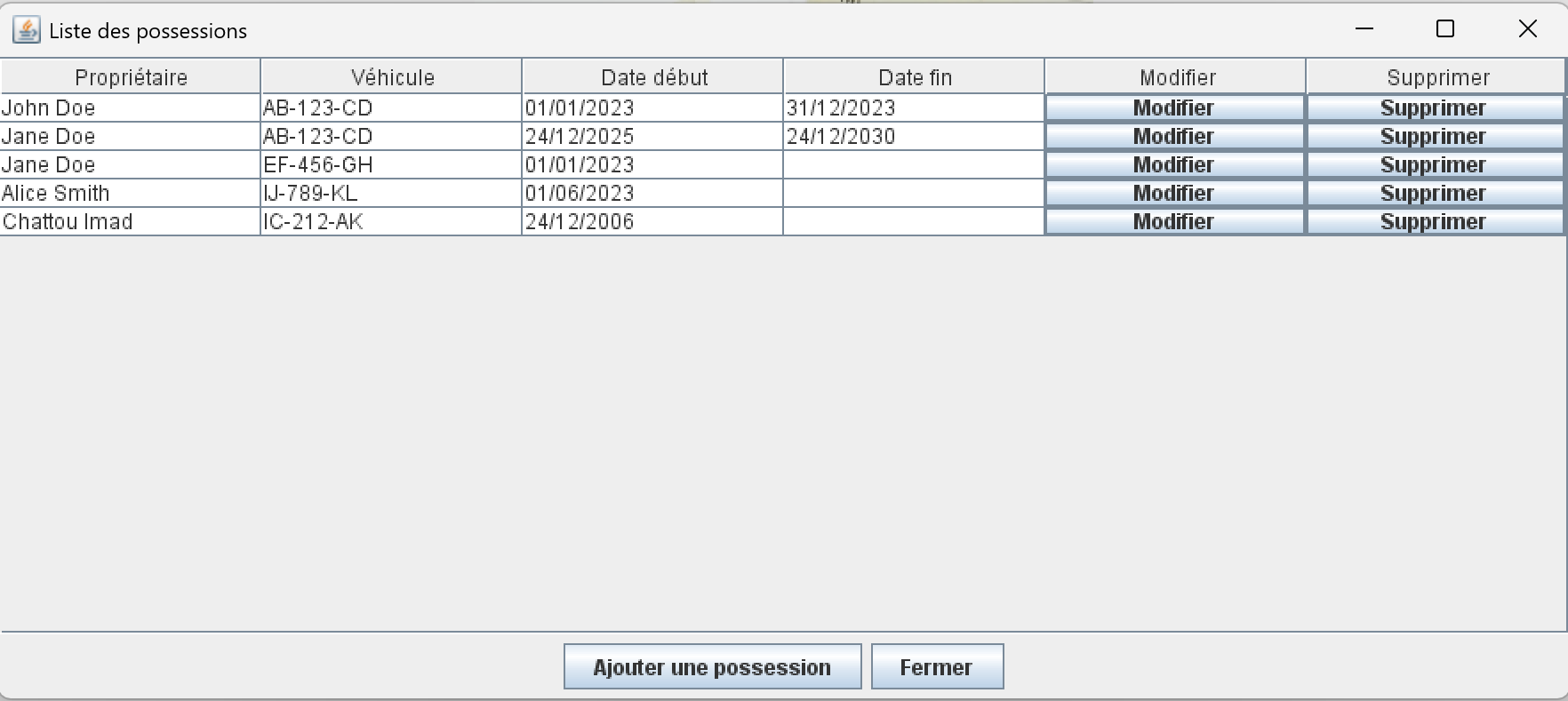
Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, Police, nombre

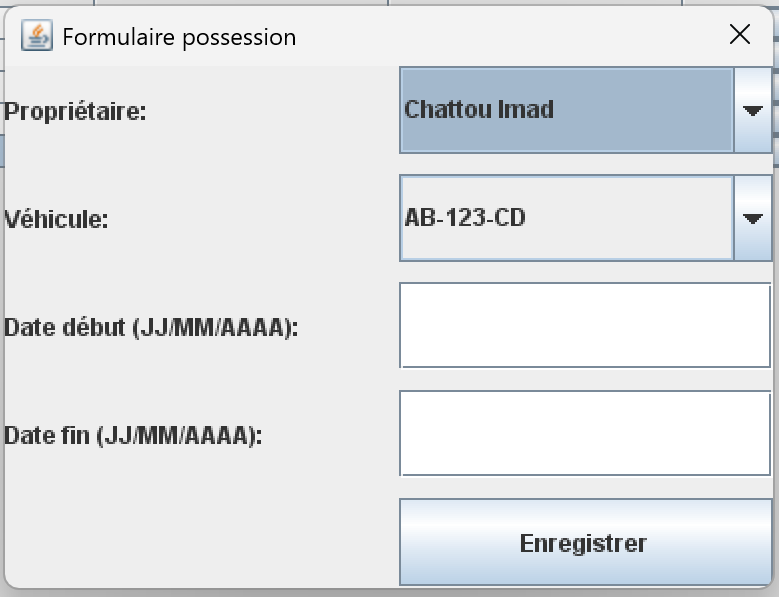
Le contenu généré par l’IA peut être incorrect.

### 5.1.2. Maquettes

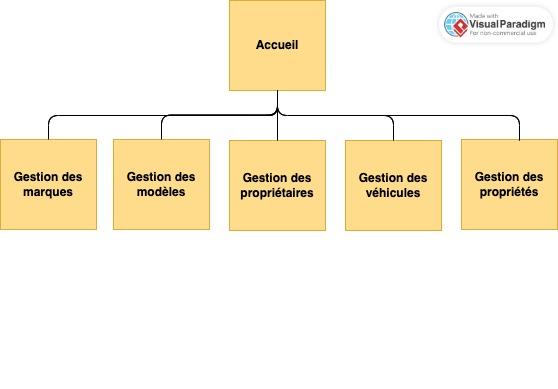
Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.





### 5.1.3. Arborescences



## 5.2. Le back-end

**REPRENDRE RÉPONSES MISSIONS 2 ET 3**

### Une image contenant texte, capture d’écran, diagramme, Police Le contenu généré par l’IA peut être incorrect.5.2.1. Diagramme de cas d’utilisation

### 5.2.2. Diagramme d’activités

### 5.2.3. Modèles Conceptuel de Données (MCD)

### 

### 5.2.4. Modèle Logique de Données (MLD)

• MARQUE (id\_marque, nom\_marque)

Clé primaire :id\_marque

• MODELE (id\_modele, nom\_modele, id\_marque)

Clé primaire :id\_modele

Clé étrangère :id\_marque en référence à id\_marque de MARQUE

• VEHICULE (id\_vehicule, matricule, annee\_sortie, poids, puissance\_chevaux, puissance\_fiscale, id\_modele)

Clé primaire : id\_vehicule

Clé étrangère :id\_modele en référence à id\_modele de MODELE

• PROPRIETAIRE (id\_proprietaire, nom, prenom, adresse, cp, ville)

Clé primaire :id\_proprietaire

• POSSEDER (id\_proprietaire, id\_vehicule, date\_debut\_propriete, date\_fin\_propriete)

Clé primaire :id\_proprietaire, id\_vehicule

Clés étrangères :id\_proprietaire en référence à id\_proprietaire de PROPRIETAIRE id\_vehicule en référence à id\_vehicule de VEHICULE

### 5.2.5. Modèle Physique de Données (MPD)

### 

# Technologies utilisées

**REPRENDRE RÉPONSES MISSION 1**

## 6.1. Langages de développement Web

Une image contenant cercle, symbole, logo, Police

Le contenu généré par l’IA peut être incorrect.

## Une image contenant texte, Graphique, graphisme, Police Le contenu généré par l’IA peut être incorrect.6.2. Base de données

Une image contenant Graphique, Police, conception

Le contenu généré par l’IA peut être incorrect.

# Sécurité

**DÉFINISSEZ ET EXPLIQUEZ BRIÈVEMENT VOTRE SOLUTION.**

**AJOUTEZ SI VOUS LE SOUHAITEZ DES COURS EXTRAITS DE CODE.**

## 7.1. LOGIN ADMIN (Titre conservé)

**Définition de la solution :**

L'accès à l'interface d'administration est protégé par une étape d'identification obligatoire. Il s'agit de s'assurer que l'utilisateur possède les droits requis avant de lui ouvrir les fonctionnalités de gestion.

* **Processus :** Saisie des identifiants via un formulaire, hachage et comparaison via BCrypt, puis initialisation d'une session utilisateur en cas de succès.

**PAGES ADMIN**

Il s'agit des espaces de travail réservés exclusivement aux gestionnaires du système. Ces interfaces permettent de piloter l'application (Tableau de bord, annuaire des utilisateurs, journaux d'événements et paramètres système).

**PROTECTION**

Mise en place de barrières de sécurité pour interdire l'usage des outils critiques par des personnes non autorisées.

**Extraits de code**

Java

|  |
| --- |
| // VALIDATION MANUELLE AVANT SUPPRESSION  int reponse = JOptionPane.showConfirmDialog(ProprietaireView.this,  "Voulez-vous vraiment supprimer ce dossier ?",  "Alerte de sécurité",  JOptionPane.YES\_NO\_OPTION);  // CAPTURE DES ANOMALIES D'AFFICHAGE  try {  new ProprietaireView(proprietaireController).showWindow();  } catch (Exception erreurAccès) {  showError("Erreur d'interface Propriétaire", erreurAccès);  } |

**MESURES DE SÉCURITÉ :**

* Double vérification auprès de l'usager avant toute suppression.
* Interception des erreurs pour éviter le plantage de l'application.
* Isolation des méthodes de modification de données.

## 7.2. Hachages des mots de passe avec Bcrypt (Titre conservé)

**Définition de la solution :**

Bcrypt est une méthode de chiffrement unidirectionnelle spécifiquement créée pour le stockage des mots de passe. Au lieu d'enregistrer le texte en clair, on génère une empreinte numérique unique et irréversible. Sa force réside dans sa lenteur calculée, ce qui rend les attaques par dictionnaire ou force brute extrêmement coûteuses en temps pour un pirate.

**ATOUTS DE BCRYPT :**

* **Sel (Salt) automatique :** Protection native contre l'usage de tables de hachage pré-calculées (Rainbow Tables).
* **Évolutivité :** Possibilité de renforcer la sécurité en ajustant le coût de calcul selon la puissance des processeurs actuels.

**Extraits de code**

Java

|  |
| --- |
| // DÉTECTION DES ENREGISTREMENTS IDENTIQUES  public static boolean estDejaExistant(String numMatricule, Integer idAExclure) {  String requêteSql = "SELECT COUNT(\*) FROM VEHICULE WHERE matricule = ?";    if (idAExclure != null) {  requêteSql += " AND id\_vehicule != ?";  }  try (Connection laConnexion = DBConnection.getConnection();  PreparedStatement instruction = laConnexion.prepareStatement(requêteSql)) {  instruction.setString(1, numMatricule);  if (idAExclure != null) {  instruction.setInt(2, idAExclure);  }  try (ResultSet resultat = instruction.executeQuery()) {  if (resultat.next()) {  return resultat.getInt(1) > 0;  }  }  } catch (SQLException erreur) {  System.err.println("Log Erreur (Unicité) : " + erreur.getMessage());  }  return false;  } |

## 7.3. Protection contre les attaques XSS (Cross-Site Scripting) (Titre conservé)

**Définition de la solution :**

L'attaque XSS consiste à injecter des scripts malveillants dans une application. Si ces données ne sont pas nettoyées, le navigateur des autres utilisateurs exécutera ce code, permettant le vol de cookies ou la redirection vers des sites frauduleux.

**FORMES D'ATTAQUES :**

1. **XSS Persistant :** Le script est sauvegardé directement en base de données.
2. **XSS Non-persistant :** Le code transite temporairement par un paramètre d'URL ou un champ de recherche.

**Extraits de code**

Java

|  |
| --- |
| // NETTOYAGE DES ENTRÉES (Exemple Vehicule.java)  public static boolean verifPresence(String codePlate, Integer idIgnore) {  String sqlCmd = "SELECT COUNT(\*) FROM VEHICULE WHERE matricule = ?";  // L'usage de setString() ci-dessous neutralise les tentatives d'injection  // ... (Logique de connexion identique)  }  // RENDU SÉCURISÉ DANS L'INTERFACE (ProprietaireView.java)  // Ajout des lignes dans le composant JTable  vueTableau.addRow(new Object[]{  p.getIdProprietaire(),  p.getNom(), // La JTable traite le texte comme du contenu brut  p.getPrenom(), // Aucun script ne peut s'exécuter dans cette cellule  p.getAdresse(),  "Action : Supprimer"  }); |

**STRATÉGIES APPLIQUÉES :**

* **Neutralisation SQL :** Utilisation systématique de PreparedStatement.
* **Échappement automatique :** Utilisation de composants graphiques Java Swing qui ne traitent pas le HTML/JS.

## 7.4. Protection contre les injections SQL (Titre conservé)

**Définition de la solution :**

Une injection SQL survient lorsqu'un pirate parvient à glisser des commandes SQL à l'intérieur d'un formulaire. Sans protection, le serveur exécute ces ordres "parasites" comme s'ils faisaient partie du code légitime, risquant ainsi la fuite totale de la base de données ou sa destruction.

**Extraits de code**

Java

|  |
| --- |
| // USAGE DES REQUÊTES PARAMÉTRÉES  public static boolean enregistrerVehicule(String plaque, int anSortie, double masse,  int cvVapeur, int cvFiscaux, int refModele) {    // On utilise des points d'interrogation comme marqueurs de place (Placeholders)  String commande = "INSERT INTO VEHICULE (matricule, annee\_sortie, poids, " +  "puissance\_chevaux, puissance\_fiscale, id\_modele) " +  "VALUES (?, ?, ?, ?, ?, ?)";  try (Connection db = DBConnection.getConnection();  PreparedStatement pStmt = db.prepareStatement(commande)) {  // Attribution des valeurs avec typage rigoureux  pStmt.setString(1, plaque);  pStmt.setInt(2, anSortie);  pStmt.setDouble(3, masse);  pStmt.setInt(4, cvVapeur);  pStmt.setInt(5, cvFiscaux);  pStmt.setInt(6, refModele);  return pStmt.executeUpdate() > 0;  } catch (SQLException e) {  System.err.println("Echec de l'insertion : " + e.getMessage());  return false;  }  } |

**POINTS CLÉS DU CODE :**

* **Typage fort :** En utilisant setInt() ou setString(), le système vérifie que la donnée correspond au format attendu.
* **Transactions :** Utilisation de commit et rollback pour garantir que la base de données ne reste jamais dans un état instable en cas de bug.