Projet Jee - Agence de location automobile

Description du Projet

Ce projet est une application web de gestion de location de véhicules, développée en utilisant **Spring Boot** avec les composants **Spring Data JPA**, **Spring MVC**, et **Thymeleaf** pour le rendu des vues. L'application permet de gérer des entités telles que :

- **Client** (utilisateurs enregistrés)
- **Contrat de Location** (liaison entre les clients et les véhicules)
- **Véhicule** (les voitures disponibles pour la location)
- **Paiement** (gestion des paiements des locations)
- **Type de Véhicule** (catégories de véhicules)

L'objectif principal est d'offrir un système de gestion complet respectant l'architecture MVC, tout en intégrant une logique métier efficace pour relier les entités et faciliter leur manipulation.

Fonctionnalités Principales

1. Gestion des Clients

- Inscription et connexion des utilisateurs.
- CRUD complet sur les clients (à travers l'interface administrateur).
- Gestion de l'état de connexion des utilisateurs.

2. Gestion des Véhicules

- Ajout et recherche de véhicules en base de données.
- Classification des véhicules par type (catégorie, modèle).
- Affichage dynamique des véhicules dans une vue utilisateur.

3. Contrats de Location

- Création d'un contrat de location entre un client et un véhicule.
- Gestion des contrats via l'interface administrateur.
- Liaison entre les clients et les contrats pour une relation **OneToMany**.

5. Authentification et Autorisation

- Mise en place d'un système d'authentification avec **Spring Security**.

- Différenciation des rôles (**ADMIN** et **CLIENT**) pour accéder à des fonctionnalités distinctes.

6. Interface Utilisateur Dynamique

- Pages web réactives et ergonomiques utilisant **Thymeleaf** et **Tailwind CSS.**.
- Formulaires interactifs pour filtrer les véhicules par catégorie ou modèle.
- Accès conditionné à certaines pages en fonction du rôle utilisateur.

Architecture du Projet

L'architecture suit les principes **MVC** (Model-View-Controller) :

- **`src/main/java/com/agencelocation/`** :
  - **`config/`** : Contient les configurations de **Spring Security** et autres services.
  - **`controller/`** : Regroupe les contrôleurs pour chaque entité (ex : `ClientController`, `VehiculeController`, etc.).
  - **`model/`** : Contient les classes représentant les entités (ex : `Client`, `Vehicule`, etc.).
  - **`repository/`** : Définit les interfaces pour l'accès aux données.
  - **`GestionlocationApplication.java`** : Point d'entrée principal de l'application Spring Boot.
- **`resources/`** :
  - **`templates/`** : Contient les vues **Thymeleaf** (ex : `home.html`, `location.html`, `login.html`…).
  - **`application.properties`** : Fichier de configuration de la base de données.
- **`pom.xml`** : Fichier de configuration des dépendances **Maven**.

Technologies Utilisées

- **Spring Boot** : Framework pour le développement rapide d'applications Java.
- **Spring Data JPA** : Gestion des opérations CRUD avec une base de données relationnelle.
- **Spring Security** : Gestion de l'authentification et de l'autorisation.
- **Thymeleaf** : Moteur de template pour la création de vues dynamiques.
- **Tailwind CSS** : Framework CSS pour un design moderne et responsive.
- **H2 Database** : Base de données relationnelle pour stocker les entités.
- **Maven** : Gestionnaire de dépendances du projet.

Installation et Exécution

Prérequis

- **Java 17 (le projet ne fonctionne pas avec des versions antérieures)**
- **Maven (Synchronisez Maven dans l'IDE avant de lancer le projet, loader le script Maven).**
- **Connexion Internet : Requise pour le bon fonctionnement de Tailwind CSS.**
- **IDE** (IntelliJ, Eclipse, etc.)

Instructions

1. Clonez le repository :
   git clone https://github.com/Redabelcadi11/gestionlocation
2. Ouvrez le projet dans votre IDE.
3. Installez les dépendances Maven :
   mvn clean install
4. Configurez la base de données dans `application.properties`.
5. Lancez l'application Spring Boot.
6. Accédez à l'application sur : http://localhost:8080

Exemple d'Utilisation

1. **Accueil** : L'utilisateur arrive sur la page d'accueil.
2. L'utilisateur clique sur le bouton voir les véhicules disponibles.
3. Il est alors redirigé vers vers la page /vehicules qui montre les véhicules disponibles.
4. **Gestion des Véhicules** : L'utilisateur clique sur le véhicule qu'il souhaite.
5. L'utilisateur clique sur "Louer ce vehicule".
6. **Contrats** : Il est redirigé vers une page pour faire un contrat de location, où l'utilisateur sélectionne la date souhaitée.

Contributeurs

- **BELCADI ABBASSI Mohammed Reda** : Développement des entités, de la base de données, Conception des vues avec Thymeleaf, utilisation de Tailwind CSS.
- **OZUST Jordi**: Implémentation des contrôleurs, de la logique métier, de la configuration de sécurité.

Auto-évaluation

Fonctionnalités : 4.25/5

L'application couvre toutes les fonctionnalités demandées, notamment l'ajout, la modification, la recherche et la liaison des entités en base de données. Seule la suppression d'une entité est manquante.

Technique : 5/5

Architecture MVC respectée avec utilisation des bonnes pratiques HTTP et Spring, et chaque vue manipule des données transmises par son contrôleur

Qualité : 5/5

L'interface utilisateur est soignée, moderne et responsive grâce à **Tailwind CSS**. Le code source est propre, bien structuré et versionné sur un dépôt **GitHub** avec des commits réguliers pour chaque membre de l'équipe.

Soutenance : 5/5

La soutenance ayant été annulée, la note maximale est attribuée pour cette partie.

Note finale : 19.25/20

Le projet est complet, bien structuré, et respecte les exigences imposées.
