Projet : Application Web Spring MVC pour la Gestion des Locations d'Immeubles

Auteur : M LO Makhmadane

I. Introduction

Ce projet a pour objectif de développer une **application web Spring Boot** destinée à la **gestion des locations d'immeubles**. Cette application permettra :

- Aux **propriétaires** de gérer leurs immeubles et unités de location,
- Aux **locataires** de consulter les offres et suivre leurs contrats de location,
- À un **administrateur** de superviser l'ensemble des utilisateurs et opérations de location.

II. Objectifs Fonctionnels

1. **Gestion des Immeubles**
    - Ajouter, modifier, supprimer des immeubles.
    - Gérer les détails : adresse, description, nombre d'unités, équipements, etc.
2. **Gestion des Unités de Location (Appartements)**
    - Ajouter, modifier, supprimer des unités.
    - Spécifier : nombre de pièces, surface, loyer, statut (disponible/occupé), etc.
3. **Gestion des Locataires**
    - Inscription et authentification (login) des locataires.
    - Consultation des offres filtrables (par prix, localisation, nombre de pièces).
    - Envoi et suivi des **demandes de location**.
4. **Gestion des Locations et Paiements**
    - Création et visualisation des **contrats de location**.
    - Suivi des **paiements mensuels de loyers**.
    - **Génération de reçus PDF** et gestion des **relances automatiques**.
5. **Administration (pour le gestionnaire)**
    - Gestion des utilisateurs (propriétaires, locataires).
    - Suivi des activités, statistiques, rapports de location et paiement.
    - Configuration des paramètres système.

III. Technologies Utilisées

- **Backend :**
    - Spring Boot, Spring MVC
    - Spring Data JPA (Hibernate)
    - Spring Security (authentification)
    - Lombok (pour réduire le boilerplate)
- **Frontend :**
    - Thymeleaf (moteur de template)
    - Bootstrap (interface responsive)
- **Base de Données :**
    - MySQL ou PostgreSQL
- **Outils :**
    - Maven, IntelliJ IDEA / Eclipse
    - JUnit et Spring Test (MockMVC)
    - Flyway ou Liquibase (pour la gestion des migrations, optionnel)

IV. Fonctionnalités Détaillées

1. Gestion des Immeubles

- **Ajout d'un immeuble** avec nom, adresse, description, équipements.
- **Modification et suppression** d'immeubles existants.
- Affichage de la liste des immeubles avec pagination.

2. Gestion des Unités

- Ajout de nouvelles unités dans un immeuble donné.
- Attributs : numéro, surface, pièces, loyer, statut.
- Modification / suppression d'une unité.

3. Gestion des Locataires

- **Formulaire d'inscription** et **connexion sécurisée**.
- **Espace personnel** pour voir son profil, ses demandes, contrats.
- Visualisation des unités disponibles avec filtres.
- Envoi de **demande de location**.

4. Gestion des Contrats & Paiements

- Création de **contrats de location** (date de début/fin, montant, immeuble, locataire).
- Visualisation des **paiements mensuels** avec statut (payé/en retard).
- Génération de **reçus PDF**.
- **Relances automatiques par mail** (bonus).

5. Administration

- Liste des utilisateurs avec rôles.
- Tableau de bord d'activités.
- **Rapports PDF/Excel** sur :
    - Nombre de contrats en cours
    - Revenus générés
    - Loyers impayés, etc.

V. Architecture du Projet

1. Modèle de Données (JPA)

- **Entités :**
    - Immeuble, UniteLocation, Locataire, Contrat, Paiement, Utilisateur
- **Relations :**
    - Immeuble ⟶ plusieurs UniteLocation
    - Locataire ⟶ plusieurs Contrat
    - Contrat ⟶ 1 UniteLocation + 1 Locataire
    - Paiement ⟶ 1 Contrat

2. Architecture Logicielle (MVC)

- **Controller :** gère les routes et requêtes HTTP
- **Service :** contient la logique métier
- **Repository (DAO) :** accède à la base via Spring Data JPA
- **View :** pages web avec Thymeleaf