

Spécifications Techniques - Application de Gestion d'Hôtels "Super-hotel.fr"

1. Framework Backend

a. Spring Boot : Version 3.4.1

Modules utilisés :

- Spring Web : Gestion des requêtes HTTP et exposition de l'API REST.
 - Spring Data JPA : Interaction avec la base de données via JPA/Hibernate.
 - Spring Security : Gestion de l'authentification et des autorisations avec JWT.
-

2. Langage de Programmation

- Java : Version 20
-

3. Base de Données

- PostgreSQL : Stockage des villes, hôtels, utilisateurs, réservations, et gestionnaires.
- Connexion JDBC : Configuration via `application.properties`.
- Schéma de base de données :
 - `City` (id, name)
 - `Hotel` (id, name, address, phone, stars, description, lowest_price, city_id)
 - `Room` (id, hotel_id, room_type, price, available)
 - `User` (id, name, email, password, role)
 - `Booking` (id, user_id, hotel_id, checkin_date, checkout_date, guests)

4. Bibliothèques Complémentaires

- Lombok : Réduction du code boilerplate avec des annotations comme `@Data`, `@Getter`, `@Setter`.
- JWT (JSON Web Token) : Sécurisation des endpoints API avec authentification.


5. Gestion de la Sécurité

- Spring Security + JWT :
 - Authentification des utilisateurs (visiteurs, gestionnaires, superviseurs).
 - Gestion des rôles (`ROLE_VISITOR`, `ROLE_MANAGER`, `ROLE_SUPERVISOR`).
 - Restriction d'accès aux endpoints sensibles (ex : ajout/modification d'hôtels, gestion des villes).

6. Gestion des Dépendances


- Maven (`pom.xml`) : Gestion et résolution des dépendances.


7. Structure du Projet

 `src/main/java/fr/fms`

 `entities` : Contient les entités Java (`City`, `Hotel`, `Room`, `User`, `Booking`).

 `repositories` : Interfaces JPA pour l'accès aux données.

 `services (business)` : Contient la logique métier.

 `controllers` : Exposition des routes API et gestion des requêtes utilisateur.

 `security` : Gestion des tokens JWT et configuration Spring Security.

8. API REST - Endpoints

- GET /api/cities → Liste des villes disponibles.
 - GET /api/hotels?city={cityId} → Liste des hôtels par ville.
 - GET /api/hotels/{hotelId} → Détails d'un hôtel.
 - POST /api/bookings → Réservation d'une chambre.
 - POST /api/auth/login → Authentification d'un utilisateur.
 - POST /api/admin/cities → (Admin) Ajout d'une ville.
 - PUT /api/admin/hotels/{id} → (Manager/Admin) Modification d'un hôtel.
 - DELETE /api/admin/hotels/{id} → (Admin) Suppression d'un hôtel.
-

9. Frontend

- Framework : Angular 15
 - Consommation de l'API REST fournie par le backend avec `HttpClient`.
 - Services Angular : Gestion des hôtels, villes, réservations et utilisateurs.
 - Gestion de l'authentification avec JWT (stockage du token dans `localStorage`).
 - Interface responsive avec Bootstrap ou TailwindCSS.
 - Pagination pour la liste des hôtels.
-

10. Évolutions Futures (Sprint Suivant)

- Paiement en ligne sécurisé.
 - Géolocalisation des hôtels via Google Maps API.
 - Gestion des images multiples pour les hôtels.
 - Notifications en temps réel via WebSockets pour les réservations.
 - Ajout des avis et commentaires pour les hôtels.
-