**CAHIER DE CHARGES**

SOMMAIRE :

1. **FONCTIONNALITES DES UTILISATEURS**
2. **FONCTIONNALITES GLOBALES DE RESEAU SOCIAL IMMOBILIER**
3. **SUGGESTIONS D'AMELIORATION**
4. **TECHNOLOGIE UTILISEE :**
5. **ETAPES GLOBALES DE LA CONCEPTION DE NOTRE RESEAU SOCIAL IMMOBILIER**

Voici un bilan de la définition des fonctionnalités basé sur les éléments que vous avez décrits, ainsi que ceux qui émergent du diagramme UML :

**I- FONCTIONNALITES DES UTILISATEURS**

**1. Annonceurs (hérite de de la classe acquereur)**

* **Inscription avec vérification d'identité**
* **Tableau de bord personnalisé**
* **Accès aux statistiques des annonces** (vues, contacts, favoris)
* **Gestion de plusieurs propriétés**
* **Création, modification (description, photo, prix, statut prise ou non et suppression d'annonces)**
* **Upload et gestion de photos/vidéos (lorsque le client veut en savoir plus sur une annonce, il peut vois plus de photos et vidéos du site) ;**
* **Gestion des demandes de contact (répondre aux messages demandent d’achat et messages)**
* **Historique des conversations (consulter les anciens messages et notifications)**
* **Publication d'annonces (options payantes mais gratuit le premier mois de la création de compte)**
* **Calendrier de disponibilité pour les visites**
* **Système de notification pour nouveaux contacts**
* **Indicateurs de performance des annonces (consulter les vues d’une annonce) ;**
* **Outils d'aide à la rédaction d'annonces (page sur laquelle on précise les différents champs à remplir pour faire une annonce) ;**
* **Validation d'action sur les annonces** (ex: validation de vente, réservation)
* **Gestion de son compte (création, suppression, nom de compte, description compte, photo profil)**
* **Rédaction de documents juridiques** (contrats, promesses de vente)

**2. Visiteurs**

* **Navigation sans inscription (juste voir les annonces sans parcourir c’est-à-dire sans voir les diverses photos et vidéo de l’annonce)**
* **Authentification pour bénéficier des fonctionnalités avancées (bref ne peut que parcourir, liker, commenter, réserver, etc… s’il n’est pas inscrit/authentifier)**
* **Recherche avancée de biens immobiliers** (moteur de recherche)
* **Filtres personnalisés** (permet un trie en fonction des prix, surface, localisation, description, statut)

**3. Acquéreur (hérite de la classe visiteur)**

* **Consulter une annonce** (parcourir l’annonce et voir les diverses photos et vidéo du bien)
* **Sauvegarde des recherches favorites**
* **Alertes pour nouvelles annonces**
* **Favoris et comparaison de biens (méthodes de classement selon le critère souhaité)**
* **Contact direct avec les annonceurs**
* **Historique des biens consultés**
* **Planification de visites via la plateforme**
* **Évaluation des annonces et des annonceurs (réserver un bien, acheter un bien, consulter l’annonceur, liker, commenter, noter)**
* **Critiquer la plateforme (faire part aux administrateurs via messagerie des différents défauts de la plateforme en vue d’une amélioration lors de la mise à jour)**

**3. Administrateurs**

* **Tableau de bord administratif avec statistiques**
* **Gestion des utilisateurs** (pages contenant tous les utilisateurs et pouvant consulter leur site et effectuer toutes les opérations : création, modification, suppression de compte)
* **Attribution de rôles et permissions**
* **Gestion des annonces** (modération, validation, suppression des annonces)
* **Gestion financière** (suivi des transactions, plans tarifaires)
* **Configuration de la plateforme** (paramètres généraux, filtres)
* **Gestion de contenu** (pages statiques, FAQ, articles)
* **Outils de communication** (envoi d'emails, notifications, partie messagerie)
* **Système de modération** (signalements, détection de contenu suspect)
* **Rapports et analyses** (rapports personnalisables, statistiques comportementales)
* **Maintenance technique (consulter les critiques des utilisateurs)**
* **Support utilisateur** (gestion des tickets)

**II – FONCTIONNALITES GLOBALES DU RESEAU SOCIAL IMMOBILIER**

* **Profils utilisateurs avancés** pour tous types d'utilisateurs (paramétrer sa photo de profil, son nom de compte, sa description)
* **Connexion et réseau** (ajouter/suivre d'autres utilisateurs)
* **Flux d'actualités** pour publications des annonces (notification pour like, achat et promesse de vente)
* **Système de messagerie interne** amélioré (commentaire sur l’annonce)
* **Événements et webinaires** partagés sur la plateforme
* **Système d'évaluation** après interactions
* **Forums de discussion** pour poser des questions et partager des conseils (échanges entre utilisateurs, et administrateurs) ;

**III - SUGGESTIONS D'AMELIORATION**

**(Bien évidemment, on ne va rien faire de tout ça mais on va juste les citer oralement qu’avec plus d’expérience, on l’aurait fait)**

1. **Fonctionnalités Utilisateurs** :
   * **Personnalisation des Profils** : Permettre aux utilisateurs de personnaliser leur profil avec des thèmes ou des options de mise en page.
   * **Intégration de Réseaux Sociaux** : Faciliter le partage d'annonces sur des plateformes externes comme Facebook ou Twitter, pour accroître la visibilité.
2. **Expérience Utilisateur** :
   * **Tutoriels et Aide à la Navigation** : Offrir des guides interactifs pour aider les utilisateurs à s’orienter facilement dans la plateforme.
   * **Intelligence Artificielle** : Utiliser des recommandations basées sur le comportement des utilisateurs pour proposer des annonces pertinentes.
3. **Système de Messagerie** :
   * **Réponses Automatisées** : Implémenter des réponses préenregistrées pour les questions fréquentes.
   * **Support en Temps Réel** : Offrir un chat en direct pour répondre instantanément aux questions des utilisateurs.
4. **Fonctionnalités Sociales** :
   * **Interaction entre Utilisateurs** : Ajouter des outils pour que les utilisateurs puissent poser des questions directement sur les annonces, favorisant l'interaction.
   * **Évaluations et Témoignages** : Mettre en place un système où les utilisateurs peuvent laisser des recommandations pour les annonceurs fiables.
5. **Sécurité et Confidentialité** :
   * S’assurer de la mise en conformité avec les réglementations sur la protection des données (ex : RGPD).
   * **Options de Confidentialité** : Permettre aux utilisateurs de régler leurs paramètres de confidentialité sur les informations visibles par d’autres.
6. **Rapports et Analyses** :
   * Offrir des tableaux de bord personnalisables pour différents types d’utilisateurs afin qu’ils puissent visualiser les données qui les intéressent.

**IV - TECHNOLOGIE UTILISEE :**

Voici un bilan des langages de programmation et des technologies que nous pourrons utiliser pour réaliser ton projet de plateforme immobilière :

**1. Frontend (Interface Utilisateur)**

* **HTML/CSS** :
  + Utilisé pour la structure et le style de la page web.
  + Assure une présentation responsive et intuitive.
* **JavaScript** :
  + Langage de programmation principal pour l'interactivité sur le frontend.
* **React.js** :
  + Bibliothèque JavaScript pour construire des interfaces utilisateur dynamiques.
  + Permet de créer des composants réutilisables, ce qui est essentiel pour une application complexe.

**2. Backend (Serveur et Logique Métier)**

* **Java** :
  + Langage de programmation principal pour le développement backend.
  + Utilisé en conjonction avec Spring Boot pour créer des applications web.
* **Spring Boot** :
  + Framework Java pour développer des applications backend de manière rapide et efficace.
  + Fournit des outils pour la gestion des dépendances, la sécurité, et l'API REST.

**3. Base de Données**

* **SQLite** :
  + Système de gestion de base de données relationnelle pour stocker des données structurées.
  + Utile pour gérer les utilisateurs, les annonces et les interactions.
* **JDBC** :

**4. Environnement de Développement**

* **IDE** :
  + **Spring tool Suite 4** pour le développement Java et Spring Boot.
  + **Visual Studio Code** pour le développement frontend (HTML, CSS, JavaScript, React.js).
* **Serveur Local** :
  + **XAMPP/WAMP** ou autre mais nous ne sommes pas sûr.

**5. Outils et Technologies Complémentaires**

* **Git** :
  + Système de contrôle de version pour gérer le code source et collaborer avec d'autres développeurs.
* **API REST** :
  + Utilisation de RESTful APIs pour la communication entre le frontend et le backend.
* **Outils de Test** :
  + **JUnit** pour les tests unitaires côté backend.
  + **Jest** pour les tests côté frontend dans React.

**V - ETAPES GLOBALES DE LA CONCEPTION DE NOTRE RESEAU SOCIAL IMMOBILIER**

# 🛤️ Plan d'action très détaillé pour ton projet Réseau Social Immobilier

## 1. **Mise en place de l'environnement**

* **Backend** :
  + Installe **Spring Tool Suite** (ou IntelliJ si tu veux).
  + Crée un projet **Spring Boot** avec les dépendances :
    - Spring Web
    - Spring Data JPA
    - JDBC API
    - Lombok (facultatif, pour simplifier getters/setters)
* **Frontend** :
  + Installe **Node.js** + **Visual Studio Code**.
  + Lance npx create-react-app immobilier-social-network.
* **Base de données** :
  + Utilise **SQLite** (comme demandé).
  + Installe **DB Browser for SQLite** pour gérer ta BDD facilement.

## 2. **Modélisation backend avec Spring Boot**

C’est la traduction de ton UML en **Java**.

* Crée les **entités JPA** pour :
  + Utilisateur
  + Acquéreur (hérite de Utilisateur)
  + Annonceur (hérite d’Acquéreur)
  + Annonce
  + Message
  + Notification
  + Paiement
  + Contrat

(chaque classe doit suivre ton UML → attributs + relations @OneToMany, @ManyToOne, etc.)

## 3. **Création des Repositories**

* Un JpaRepository pour chaque entité :
  + Exemple : UserRepository, AnnonceRepository, etc.
* Ces interfaces permettront de **faire facilement les opérations CRUD**.

## 4. **Développement des Services**

* Logique métier (services Java) :
  + Inscription utilisateur
  + Connexion utilisateur
  + Création/édition/suppression d’annonces
  + Envoi/réception de messages
  + Gestion des paiements et contrats
  + Système de notifications
* Exemple de service :
* public interface UserService {
* User registerUser(User user);
* User loginUser(String email, String password);
* }

## 5. **Création des Contrôleurs REST**

* Un contrôleur REST par fonctionnalité :
  + /api/users
  + /api/annonces
  + /api/messages
  + /api/payments
  + /api/contracts
* Chaque endpoint renverra du **JSON**.
* Exemple :
* @PostMapping("/register")
* public ResponseEntity<User> register(@RequestBody User user) {
* return ResponseEntity.ok(userService.registerUser(user));
* }

## 6. **Configuration de la sécurité**

* À court terme : sécuriser avec un JWT serait propre, mais tu peux commencer par :
  + Contrôler les accès via @PreAuthorize ou @Secured.
  + Faire de simples contrôles par rôle (ADMIN, ACQUEREUR, ANNONCEUR).

## 7. **Création du Frontend React.js**

* Structure ton projet React :
  + /components
    - LoginForm, RegisterForm, PropertyList, PropertyDetails, MessageSection
  + /pages
    - HomePage, ProfilePage, AnnoncePage
  + /services
    - api.js pour tous les appels API (axios)
* Utilise **Axios** pour communiquer avec l'API backend.

## 8. **Connexion Frontend <-> Backend**

* Configurer **Axios** pour pointer vers http://localhost:8080/api.
* Exemples d’appels :
  + POST /api/users/register
  + GET /api/annonces
  + POST /api/messages

## 9. **Gestion de l'état (React Context ou Redux)**

* Utilise **Context API** pour gérer :
  + L’état de l’utilisateur connecté
  + Les annonces affichées
  + Les messages reçus/envoyés

## 10. **Gestion du paiement (plus tard)**

* Utiliser **Stripe** pour intégrer un paiement simple.
* Backend ➔ Créer une session Stripe.
* Frontend ➔ Intégrer **@stripe/react-stripe-js** pour payer.

## 11. **Test et Validation**

* **Backend** :
  + Tester les services et contrôleurs avec **JUnit** et **MockMvc**.
* **Frontend** :
  + Tester les composants React avec **Jest** et **React Testing Library**.

## 12. **Déploiement**

* **Backend** :
  + Déployer Spring Boot sur **Railway**, **Render**, ou un petit VPS.
* **Frontend** :
  + Déployer React sur **Vercel** ou **Netlify**.
* (Attention : penser à la base SQLite → envisager plus tard MySQL en production.)

## 13. **Maintenance et Améliorations**

* Corriger les bugs remontés par les utilisateurs.
* Ajouter des fonctionnalités supplémentaires :
  + Système de chat en temps réel (Socket.IO, Websockets)
  + Gestion avancée des rôles
  + Notifications push

# 📜 Synthèse Visuelle de ton Chemin

1. Préparer l'environnement

2. Créer backend Spring Boot : entités -> repo -> services -> contrôleurs

3. Créer frontend React : composants -> pages -> appels API

4. Connecter React ⇄ API

5. Sécuriser (simple d'abord)

6. Ajouter paiement plus tard

7. Tester (backend + frontend)

8. Déployer

9. Écouter les utilisateurs ➔ améliorer

# 🎯 En résumé : **ta priorité immédiate**

1. Formation "Créer une API avec Spring Boot" ✅
2. Mettre en place ton projet Spring Boot en suivant ce plan.
3. Commencer React juste après avoir ton API qui tourne correctement.