

Documentation Technique - Plateforme de Gestion d'inscription

1. Introduction

Ce document détaille l'architecture technique, le modèle de données et les fonctionnalités clés de la plateforme de gestion d'inscription. L'objectif est de fournir une référence complète pour le développement, la maintenance et la compréhension du système.

2. Architecture Globale

La plateforme est conçue selon une architecture modulaire et sécurisée, s'articulant autour de deux composants principaux :

- **Frontend (Application Web Client)** : Développé avec **Angular**, il offre une interface utilisateur réactive et personnalisée pour chaque rôle. L'interaction en temps réel est gérée via **WebSockets**.
- **Backend (API RESTful)** : Construit avec **Spring Boot 3 et JDK 21**, il expose une **API RESTful** sécurisée (JWT + Spring Security) pour la logique métier, la persistance des données et la communication en temps réel (STOMP WebSockets). Il intègre également des fonctionnalités de génération de documents.

La **base de données relationnelle PostgreSQL** assure la persistance des données, avec un accent sur l'optimisation des requêtes.

3. Détail des Entités

Le modèle de données est le fondement de l'application, assurant l'intégrité et la cohérence des informations. Les entités sont conçues pour refléter fidèlement les concepts métier de la clinique.

3.1 Entités de Base

InfoPersonnel : nom, prénom, email, téléphone, adresse, genre, dateNaissance.

3.2 Gestion Candidats

Candidat : hérite de InfoPersonnel, lié à DossierMedical, RendezVous, Factures, Prescriptions.

3.3 Gestion des Rendez-vous

RendezVous : dateRDV, heure, motif, statutRDV, commentaire.

3.4 Gestion Dossier

Dossier : statut, date Soumission, commentaire

3.5 Gestion Contact d'urgence

Contact d'urgence : nom complet, lien de parenté, téléphone, email

3.6 Gestion Parcours

Parcours : établissement, spécialisation, niveau, commentaire, date début, date fin

3.7 Gestion Piece justificative

Piece justificative : type, fichier, fichier content type, date upload, valide commentaire, spécialisation, niveau, commentaire, date début, date fin

3.8 Gestion des Utilisateurs

- **Utilisateur** : hérite de InfoPersonnel, password, actif, role, login/logout
- **Role**: roleType (ADMIN, USER).

4. Enums Utilisés

Pour garantir la cohérence des données et faciliter la validation :

- **Niveau étude** : (BACCALAUREAT, LICENCE, MASTER, DOCTORAT, AUTRE).
- **Sexe** : (M, F).
- **Statut dossier** : (EN_ATTENTE, INCOMPLET, VALIDE_AUTO, EN_COURS_VERIF, VALIDE_MANUEL, REFUSE).
- **Statut RDV** : (EN_ATTENTE, CONFIRMER, REFUSE).
- **Type pièce** : (CNI_RECTO, CNI_VERSO, ACTENAISANCE, DIPLOME, PASSEPORT).

5. Diagramme de Cas d'Utilisation

Objectif

Le diagramme de cas d'utilisation met en évidence les interactions entre les **acteurs** (Candidat, Agent/Administrateur) et le système de gestion des inscriptions.

Acteurs

- **Candidat** : utilisateur principal qui soumet une demande d'inscription.
- **Agent (Administrateur)** : utilisateur interne chargé de vérifier et valider les dossiers.

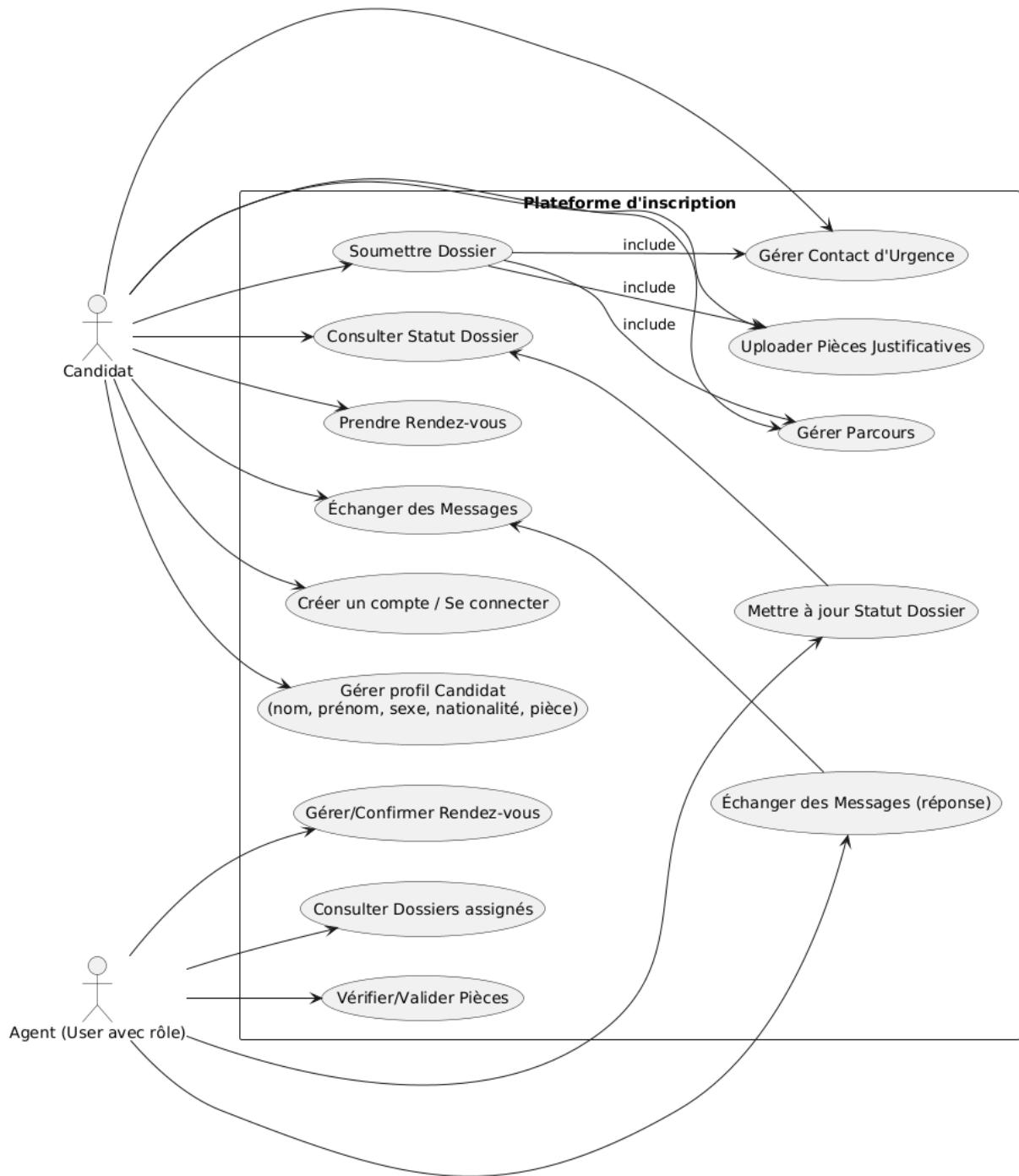
Cas d'utilisation du Candidat

1. **Créer un compte / Se connecter** : s'authentifier sur la plateforme.
2. **Gérer son profil** : renseigner nom, prénom, sexe, nationalité, type de pièce d'identité, etc.
3. **Gérer le contact d'urgence** : ajouter une personne à contacter en cas de besoin.
4. **Gérer son parcours académique** : ajouter les établissements fréquentés et le niveau d'étude atteint.
5. **Uploader les pièces justificatives** : dépôt sécurisé des diplômes, CNI, acte de naissance, photo, etc.
6. **Soumettre le dossier** : finaliser l'inscription (le dossier passe au statut *EN_ATTENTE*).
7. **Consulter le statut du dossier** : suivre l'avancement (validé automatiquement, en cours de vérification, validé manuellement, refusé).
8. **Prendre rendez-vous** : demander un entretien ou une rencontre administrative.
9. **Échanger des messages** : communiquer avec l'administration via la messagerie interne.

Cas d'utilisation de l'Agent

1. **Consulter les dossiers assignés** : accéder aux dossiers candidats.
2. **Vérifier/Valider les pièces** : contrôler la conformité des justificatifs.
3. **Mettre à jour le statut du dossier** : accepter, rejeter ou demander des compléments.
4. **Gérer/Confirmer les rendez-vous** : accepter ou refuser les demandes de RDV.
5. **Répondre aux messages** : interagir avec le candidat via la messagerie.

 Le diagramme montre donc une vision **fonctionnelle** : *qui fait quoi* dans le système.



6. Diagramme de Classes

Objectif

Le diagramme de classes modélise la **structure interne** du système : entités persistantes, attributs et relations.

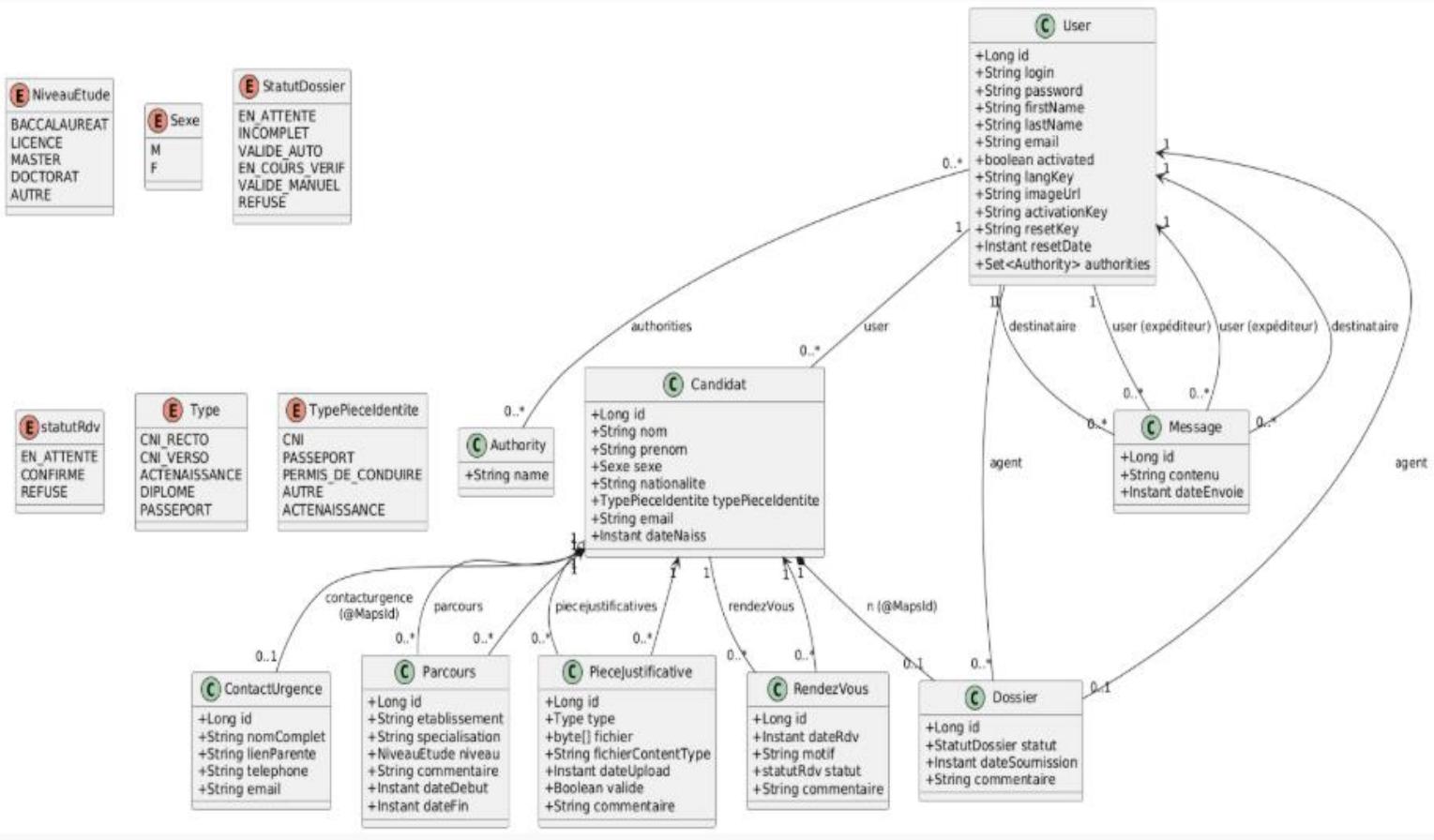
Classes principales

- **User** : représente tout utilisateur (candidat, agent). Attributs : login, email, password, activated.
- **Authority** : gère les rôles (ADMIN, USER). Relation *ManyToMany* avec User.
- **Candidat** : informations personnelles (nom, prénom, sexe, nationalité, type de pièce, email, date de naissance). Lié à un User.
- **ContactUrgence** : personne à contacter (nom complet, lien de parenté, téléphone, email). Relation *OneToOne* avec Candidat.
- **Dossier** : regroupe l'inscription d'un candidat. Attributs : statut (enum StatutDossier), dateSoumission, commentaire. Relation *OneToOne* avec Candidat et *ManyToOne* avec User (agent).
- **Parcours** : représente les études antérieures (établissement, spécialisation, niveau d'étude, dates). Relation *ManyToOne* avec Candidat.
- **PieceJustificative** : fichier uploadé (type, contenu, date upload, valide ou non). Relation *ManyToOne* avec Candidat.
- **RendezVous** : demande de RDV (date, motif, statut). Relation *ManyToOne* avec Candidat.
- **Message** : communication interne (contenu, date envoi). Relation *ManyToOne* avec User (expéditeur et destinataire).

Enums associées

- **Sexe** : M, F.
- **NiveauEtude** : Baccalauréat, Licence, Master, Doctorat, Autre.
- **TypePiecelIdentite** : CNI, Passeport, Permis, Acte de naissance, Autre.
- **Type** : CNI recto-verso, diplôme, acte de naissance, passeport.
- **StatutDossier** : En attente, Incomplet, Validé automatique, En cours de vérification, Validé manuel, Refusé.
- **StatutRdv** : En attente, Confirmé, Refusé.

 Ce diagramme représente la **base de données logique** et la **structure objet** du système.



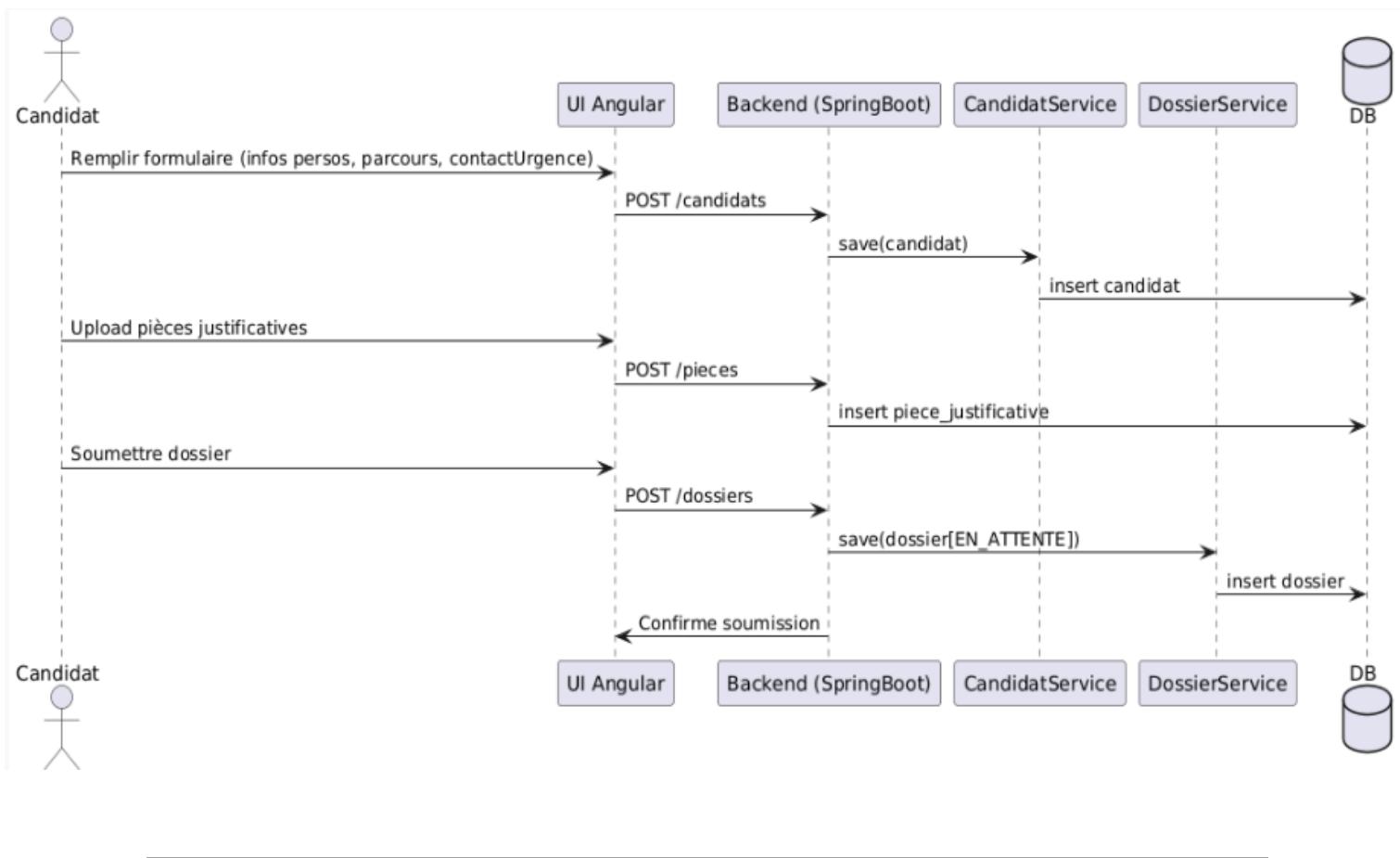
3. Diagrammes de Séquence

Objectif

Ces diagrammes décrivent les **scénarios dynamiques** : le flux d'interactions entre acteur ↔ UI ↔ Backend ↔ DB.

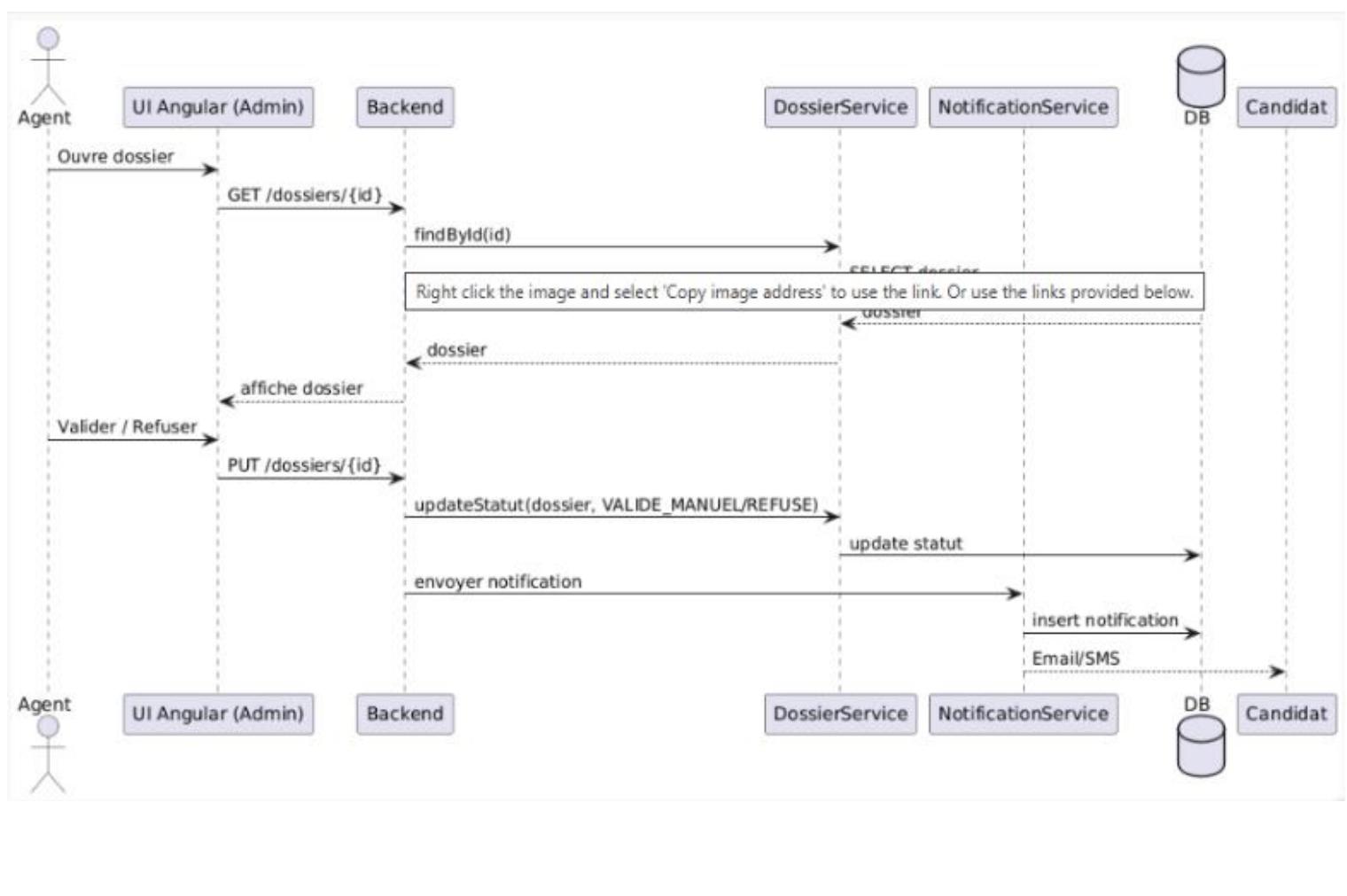
3.1 Soumission d'un dossier (par le candidat)

1. Le candidat crée son profil (informations personnelles, contact, parcours).
2. Il upload ses pièces justificatives.
3. Il soumet le dossier → statut = *EN_ATTENTE*.
4. Le système enregistre en base et renvoie une confirmation.



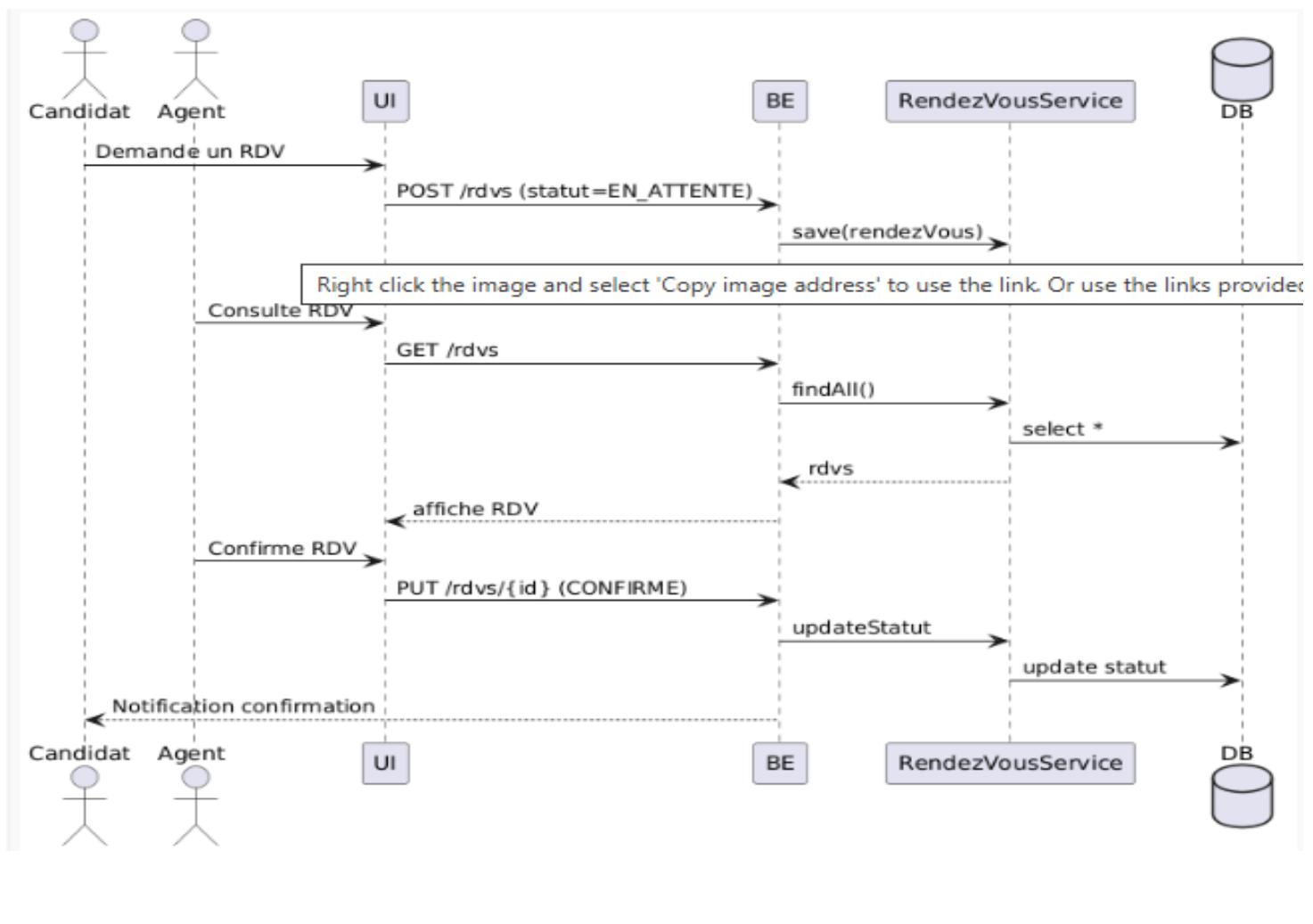
3.2 Validation du dossier (par un agent)

1. L'agent ouvre un dossier via l'interface admin.
2. Le backend récupère le dossier depuis la DB.
3. L'agent choisit d'accepter ou refuser → mise à jour du champ statut.
4. Le système envoie une notification (email/SMS) au candidat.



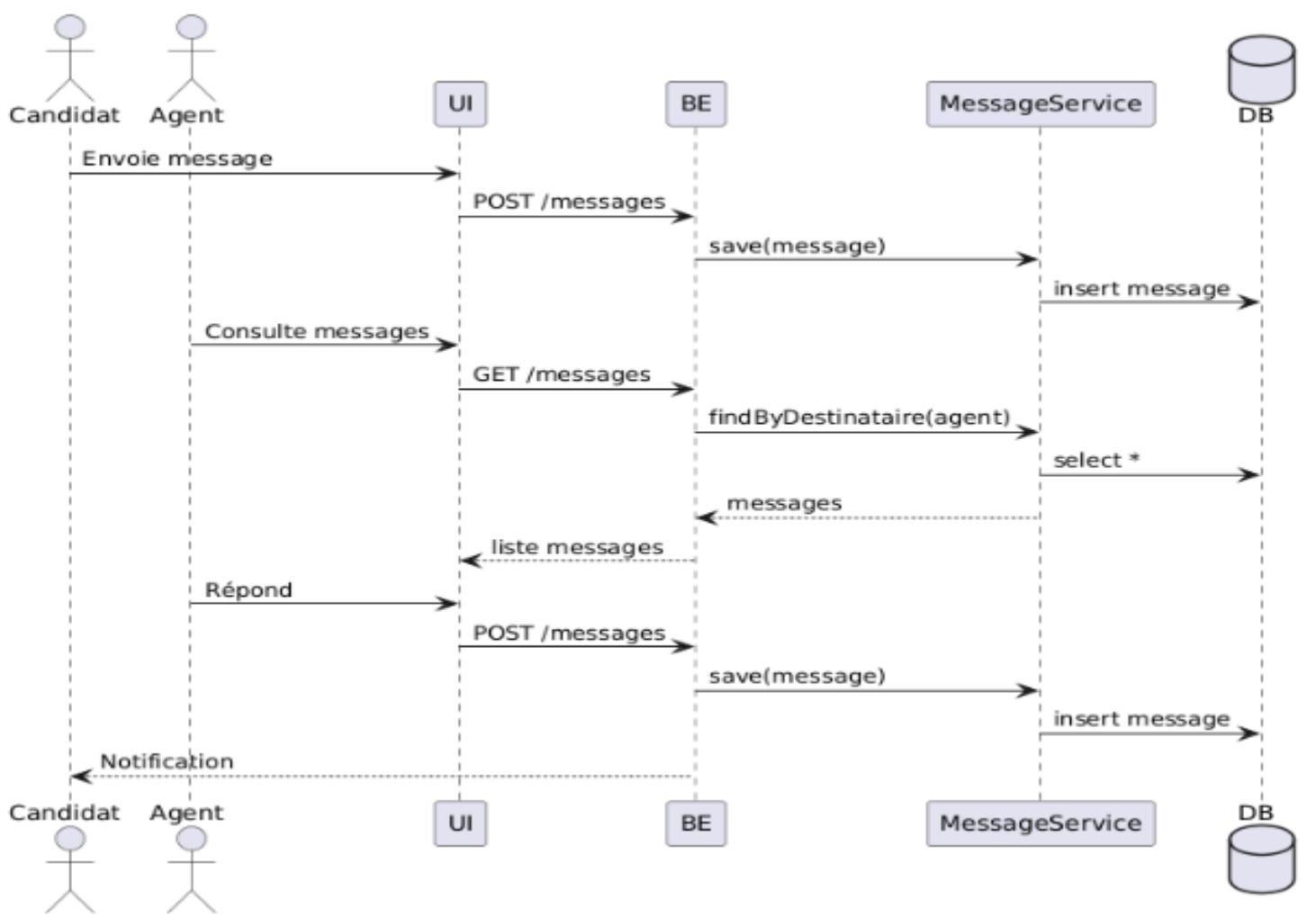
3.3 Prise et confirmation d'un rendez-vous

1. Le candidat demande un rendez-vous → statut = EN_ATTENTE.
2. Le backend stocke la demande.
3. L'agent consulte les RDV en attente.
4. L'agent confirme/refuse → mise à jour du statutRdv.
5. Le candidat reçoit une notification de confirmation/refus.



3.4 Messagerie interne (Candidat ↔ Agent)

1. Le candidat envoie un message (POST /messages).
2. Le backend l'enregistre en DB.
3. L'agent consulte sa boîte de réception.
4. L'agent répond → nouveau message inséré en DB.
5. Le candidat est notifié de la réponse.



✓ Ces séquences mettent en évidence les **processus métier** et les échanges **asynchrones/synchrones** entre acteurs et système.

Conclusion

- Le **diagramme de cas d'utilisation** illustre la **vision fonctionnelle** du système (qui interagit et comment).
- Le **diagramme de classes** détaille la **modélisation objet et relationnelle** (structure persistante).
- Les **diagrammes de séquence** décrivent les **flux d'exécution réels** pour les principaux processus (inscription, validation, RDV, messagerie).

 En combinant ces trois types de diagrammes, nous obtenons une **documentation UML complète** qui couvre :

- le **quoi** (cas d'utilisation),
- le **comment stocker/modéliser** (classes),
- le **comment ça s'exécute** (séquences).