

Systeme de Gestion de Rendez-vous Médical

Présente par :
Badri youssouf douale



Date: fevrier 2026



Encadré par :
Mr Nasry



SOMMAIRE ↗

* INTRODUCTION

* OBJECTIFS

* ANALYSE

* CONCEPTION

* TECHNOLOGIES

* ARCHITECTURE

* TESTE/SWAGGER

INTRODUCTION



Contexte Général

La digitalisation des services de santé est devenue une nécessité. Les patients attendent des services accessibles en ligne 24/7, tandis que le personnel médical a besoin d'outils efficaces pour gérer leur temps.

Problématique

Les cliniques font face à plusieurs défis

Gestion manuelle

La prise de rendez-vous par téléphone et sur papier prend trop de temps. Chaque demande peut durer 15 à 20 minutes, ce qui ralentit le travail et limite le nombre de patients servis.



Erreurs humaines

Les méthodes manuelles entraînent beaucoup d'erreurs. On observe des doubles réservations, des oublis ou des mauvaises saisies, avec un taux d'erreur proche de 30 %.



Manque de traçabilité

Sans système numérique, il est difficile de suivre l'historique des rendez-vous. Les informations se perdent facilement et la continuité des soins devient compliquée.



Pas de statistiques

L'absence de données centralisées empêche toute analyse. La clinique ne peut pas mesurer son activité ni optimiser ses ressources, ce qui limite son efficacité.



OBJECTIFS

Objectif global

Digitaliser entièrement la gestion des rendez-vous médicaux via une API REST sécurisée et documentée.



ANALYSE



L'analyse vise à identifier les acteurs, leurs besoins, les contraintes et les flux d'information du système de gestion des rendez-vous médicaux. Elle constitue la base de la conception technique et fonctionnelle.

Acteurs et Besoins fonctionnels

Patient

Inscription et authentification

Création de compte avec email, mot de passe et informations personnelles (nom, prénom, téléphone, date de naissance).

Prise de rendez-vous

Choix d'un docteur disponible, sélection de la date et de l'heure, saisie des symptômes.

Consultation des rendez-vous

affichage sous forme de calendrier ou liste, avec statut (en attente, accepté, refusé, annulé).

Annulation

possibilité d'annuler un rendez-vous selon les règles définies (ex. pas d'annulation à moins de 24h).

Acteurs et Besoins fonctionnels

Docteur

authentification

accès sécurisé au système.

Gestion des rendez-vous

consultation des demandes assignées
par la secrétaire.

Décision médicale

acceptation ou refus d'un rendez-
vous en fonction de sa disponibilité
et des symptômes.

Statut

mise à jour de sa disponibilité (libre/
occupé).

Calendrier

visualisation des rendez-vous par
jour, semaine ou mois.

Acteurs et Besoins fonctionnels

Secrétaire

authentification

accès sécurisé au système.

Gestion des demandes

visualisation des rendez-vous en attente de traitement.

Assignation des docteurs

choix du docteur approprié en fonction des symptômes et de la spécialité.

Planification urgente

création directe de rendez-vous pour les cas prioritaires.

Modification/annulation

ajustement des rendez-vous existants.

Statistiques

génération de rapports sur l'activité (nombre de rendez-vous, taux d'annulation, disponibilité des docteurs).



API clinique.

Acteurs et Besoins fonctionnels

Administrateur

Gestion des comptes

création et suppression des comptes
docteurs et secrétaires.

Supervision

contrôle global du système et suivi
des activités.

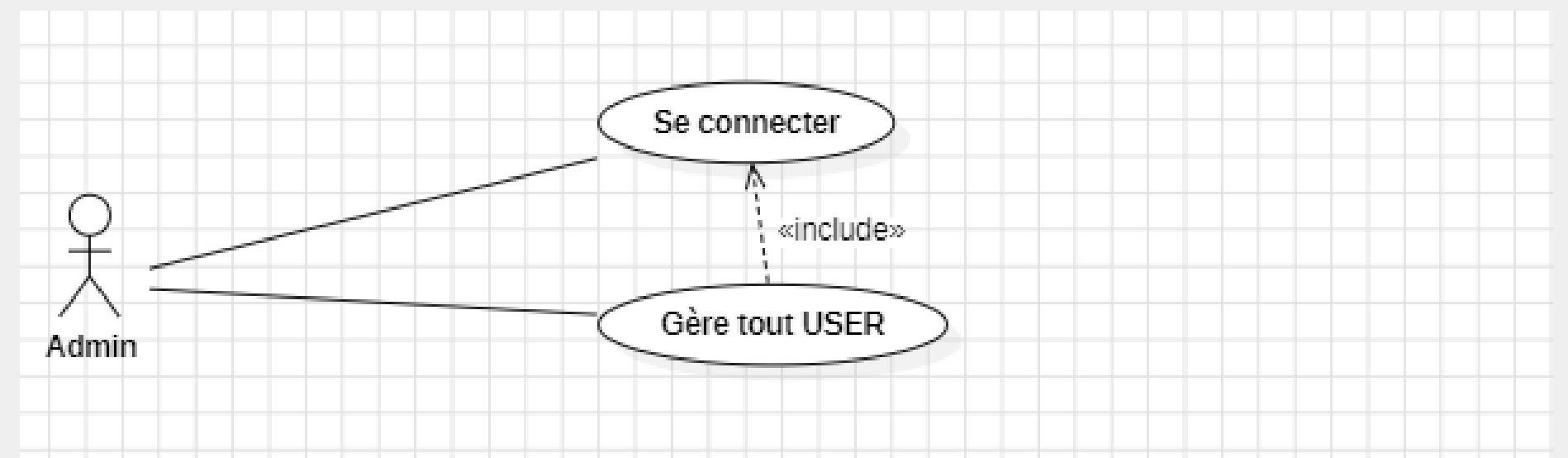
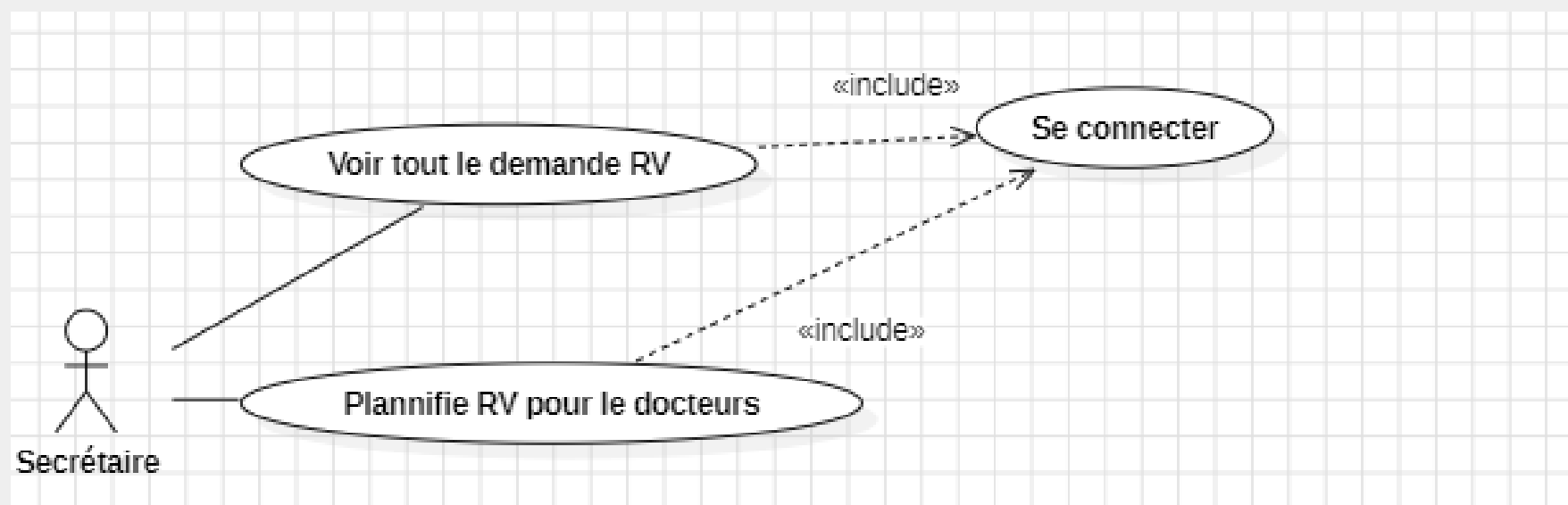
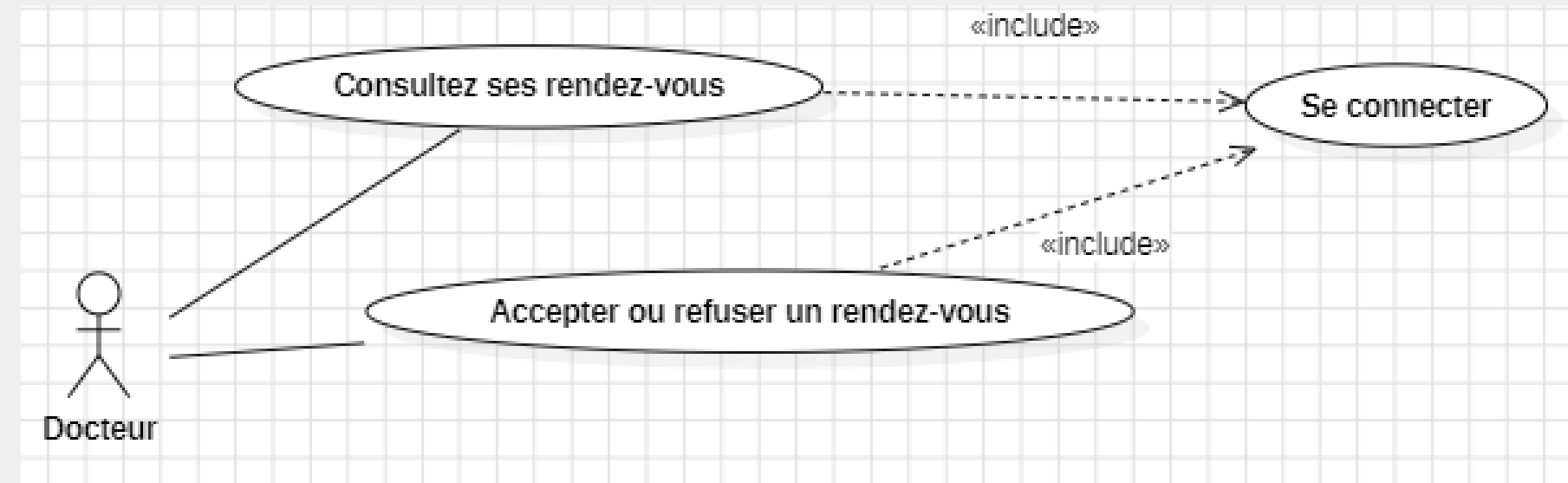
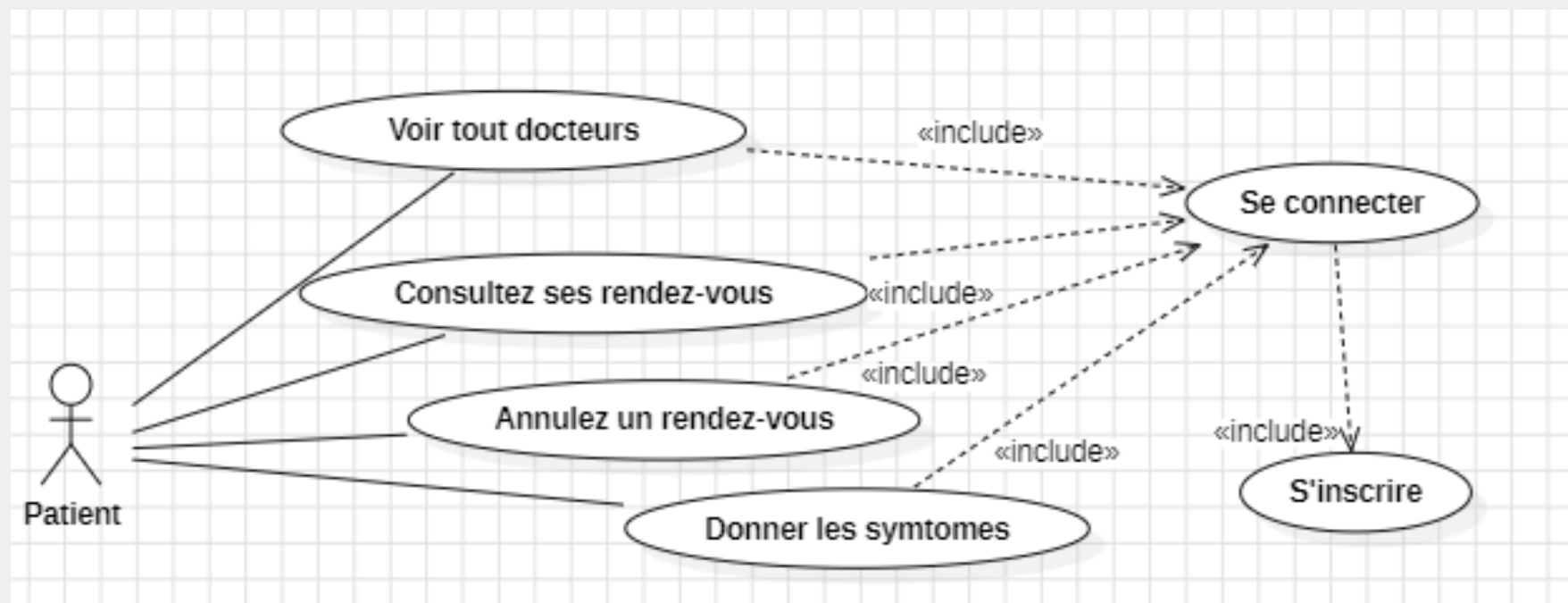
Besoins non fonctionnels (améliorations)

Modification du profil

Filtrage des docteurs par spécialité

Notifications (email ou SMS)

Diagramme de Cas d'Utilisation



CONCEPTION



La conception traduit l'analyse en modèles visuels et techniques. Elle permet de définir les interactions entre les acteurs, la structure des données et les flux de traitement.

Diagramme de Classes

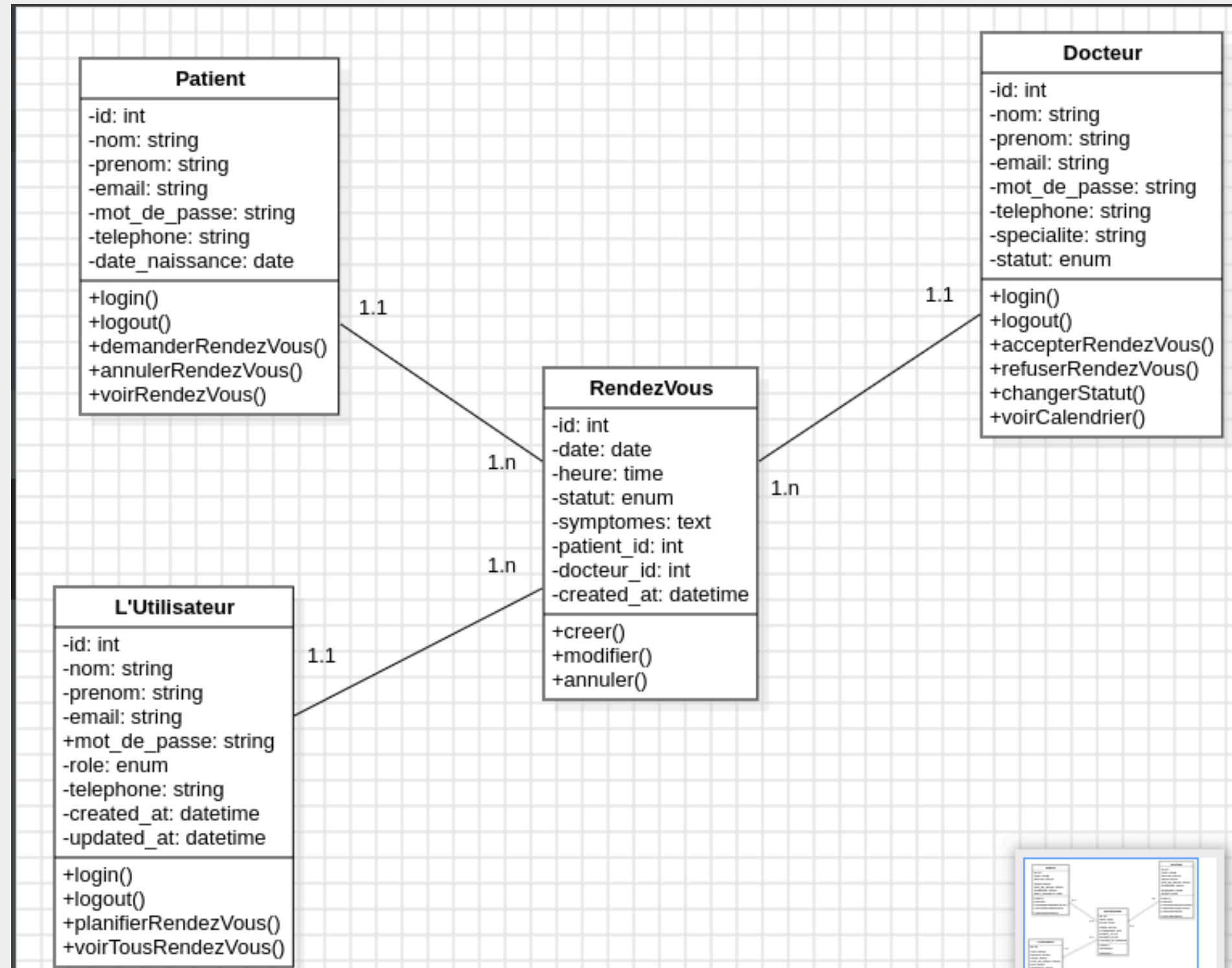
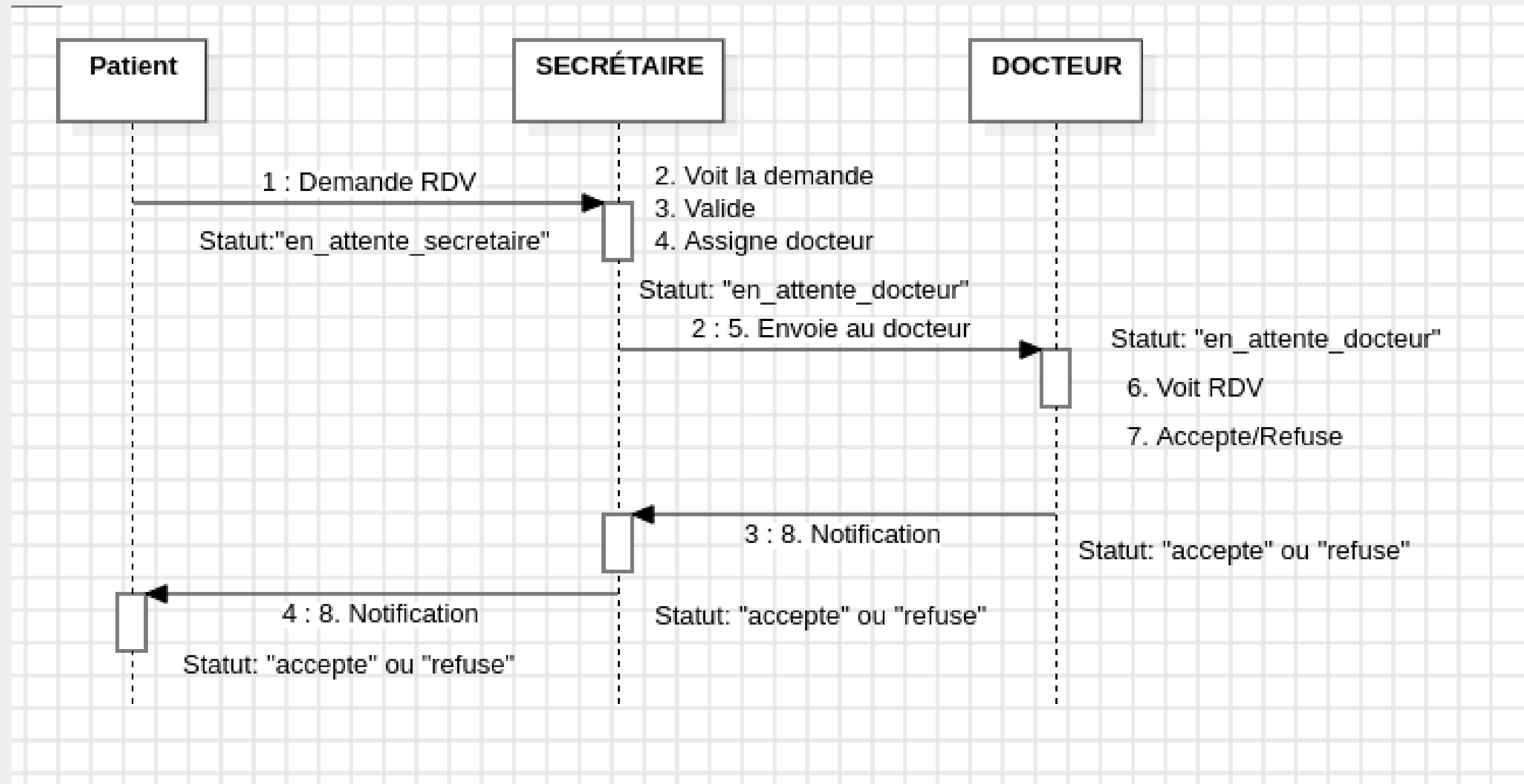


Diagramme de Séquence D'un rendez-vous



TECHNOLOGIES UTILISÉES



- Node.js
- Express.js
- MySQL
- Sequelize
- JSON Web Token
- Swagger/OpenAPI
- Redis
- bcrypt

ARCHITECTURE DU SYSTÈME

L'architecture du système repose sur le modèle MVC (Modèle-Vue-Contrôleur), une API RESTful, une base de données relationnelle, un cache en mémoire, et une documentation interactive. Elle garantit modularité, sécurité, performance et évolutivité

Architecture logique (MVC)

Routes : définissent les endpoints REST

Middleware : gèrent l'authentification JWT, la validation des rôles (RBAC), et la sécurité

Base de données : stocke les données persistantes (MySQL) et les sessions (Redis)

Contrôleurs : contiennent la logique métier pour chaque rôle (patient, docteur, secrétaire, admin)

Modèles Sequelize : représentent les entités (Patient, Docteur, RendezVous...) et leurs relations

SCÉNARIOS D'UTILISATION AVEC SWAGGER

SCÉNARIO 1 : Configuration Initiale

L'administrateur crée les comptes nécessaires pour démarrer la clinique : deux docteurs (un cardiologue et une pédiatre) et une secrétaire. À la fin, l'équipe est prête et le système devient opérationnel.



SCÉNARIO 2 : Parcours Patient Complet

Un patient s'inscrit, demande un rendez-vous et la secrétaire assigne le docteur approprié. Le docteur accepte la demande et le patient reçoit la confirmation. Ce scénario montre le cycle complet d'un rendez-vous, de la demande à la validation finale.



Conclusion

L'API Clinique constitue une solution complète, moderne et sécurisée pour la gestion des rendez-vous médicaux. Elle s'appuie sur la documentation Swagger/OpenAPI 3.0, ce qui la rend facile à utiliser grâce à une interface de test intégrée et bien documentée avec plus de 30 endpoints décrits. La sécurité est assurée par l'utilisation de JWT, du hachage bcrypt et du contrôle d'accès basé sur les rôles (RBAC). Son architecture MVC garantit une bonne scalabilité et une organisation claire, tandis que l'adoption du standard OpenAPI lui confère un caractère professionnel et conforme aux meilleures pratiques. Enfin, les six scénarios détaillés démontrent que toutes les fonctionnalités sont opérationnelles via Swagger UI, permettant une adoption rapide et efficace par les développeurs et les testeurs.