Software Requirements Specification (SRS)

Project: Hospital Appointment Management System
Author: Breton Gaspard, Coste Thomas, Ceza Mathis, Cazac Florian
Course: Software Engineering

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to describe the functional and non-functional requirements of the Hospital Appointment Booking System (HABS).
It provides a clear reference for developers, testers, and stakeholders to ensure that the system meets the hospital's needs for scheduling, managing, and monitoring medical appointments.

1.2 Scope

HABS is a web-based system that allows patients to book, modify, or cancel medical appointments online.
Doctors can manage their schedules, confirm or reject requests, and administrators can manage users, specialties, and time slots.
The system aims to simplify the hospital's appointment management process and reduce administrative workload.

1.3 Definitions, Acronyms, and Abbreviations

HABS : Hospital Appointment Booking System

API : Application Programming Interface

MFA : Multi-Factor Authentication

1.4 References

ISO 9241-210: Human-Centered Design Principles

GDPR: General Data Protection Regulation

HL7: Health Level 7 Data Interchange Standard

2. General Description

2.1 Product Perspective

HABS will integrate with the hospital’s existing systems (such as EHR or patient databases).
It acts as a centralized platform that connects patients, doctors, and administrators through a secure and user-friendly web interface.

2.2 Product Features

Online appointment booking and management

Doctor schedule visualization and control

Email/SMS notifications

Appointment history and tracking

Administrative control panel for managing users, specialties, and time slots

2.3 User Characteristics

Patients : Can book, view, or cancel appointments.

Doctors : Can manage schedules and view patient appointments.

Administrators : Manage users, doctors, and system configurations.

2.4 General Constraints

Must comply with GDPR data privacy laws.

System availability must be above 99%.

Secure authentication via MFA and HTTPS.

2.5 Assumptions and Dependencies

A stable internet connection is required.

Users must have a valid email address.

The hospital provides accurate scheduling data for doctors.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 Patient Account Management

Patients can create and log into their accounts securely.

Email verification must be completed before booking.

3.1.2 Appointment Scheduling

Patients can select a specialty, doctor, and available time slot.

The system must confirm the booking via email or SMS.

Patients can cancel or reschedule appointments within allowed limits.

3.1.3 Doctor Schedule Management

Doctors can define their availability.

They receive notifications when an appointment is created, canceled, or rescheduled.

3.1.4 Administration

Administrators can add or remove doctors, manage specialties, and monitor system usage.

They can view statistics such as appointment counts and user activity.

3.2 Non-Functional Requirements

3.2.1 Performance

The system should handle up to 200 concurrent users.

Average response time should not exceed 2 seconds.

3.2.2 Security

All communications must be encrypted (SSL/TLS).

MFA required for doctors and administrators.

An audit log must record all sensitive actions.

3.2.3 Reliability and Maintainability

Daily database backups are mandatory.

Code should be modular, well-documented, and easily maintainable.

4. Use Case Scenarios

4.1 Use Case 1 – Booking an Appointment

Actors: Patient, System
Description: The patient books an appointment online.
Steps:

The patient logs in.

Selects a specialty, doctor, date, and time.

The system shows available slots.

The patient confirms the booking.

The system sends a confirmation email.

4.2 Use Case 2 – Doctor Views Schedule

Actors: Doctor, System
Steps:

The doctor logs in.

Accesses their schedule dashboard.

Accepts, rejects, or edits appointments.

4.3 Use Case 3 – Admin Manages Users

Actors: Administrator, System
Steps:

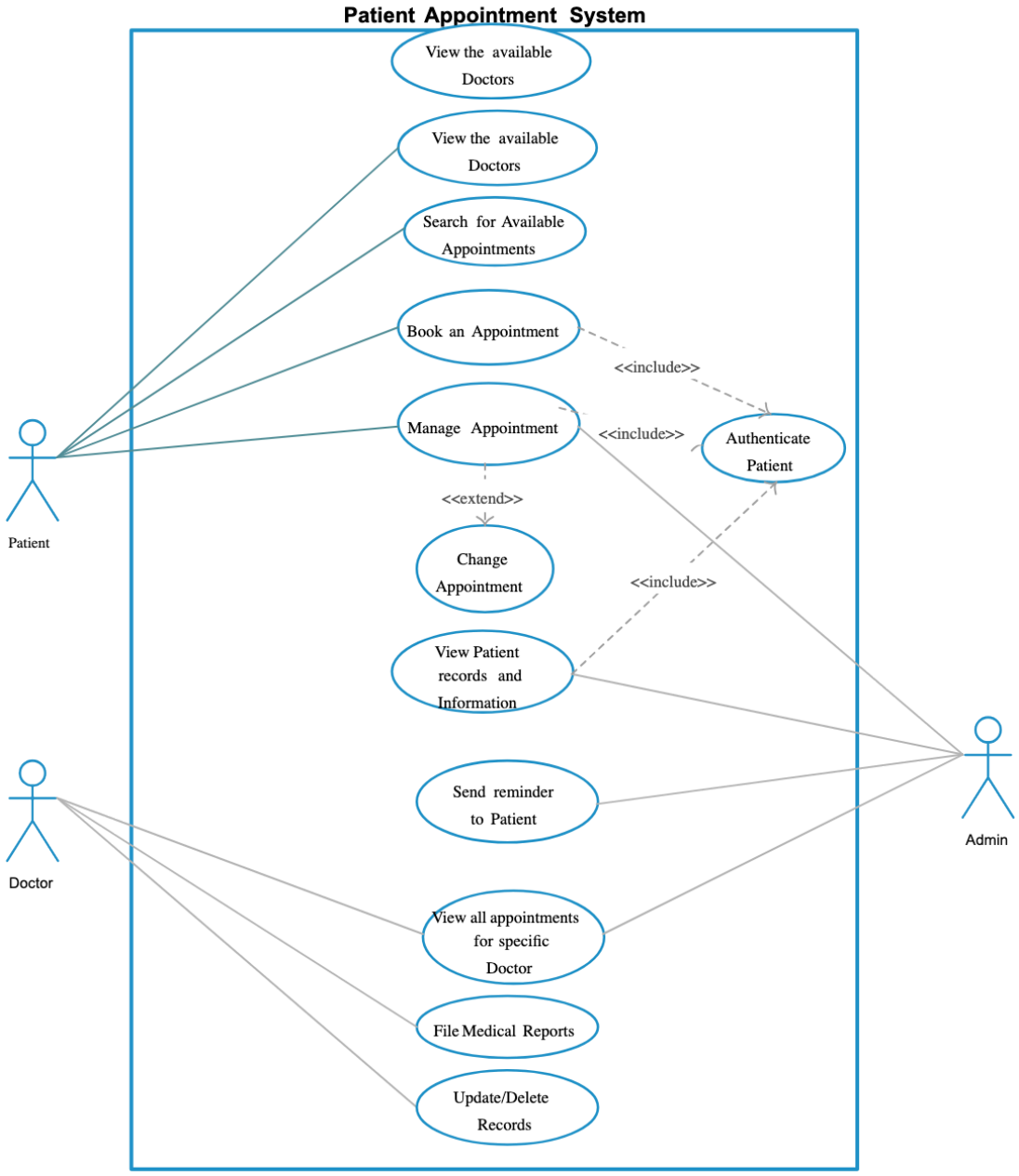
The administrator logs in.

Views the list of users.

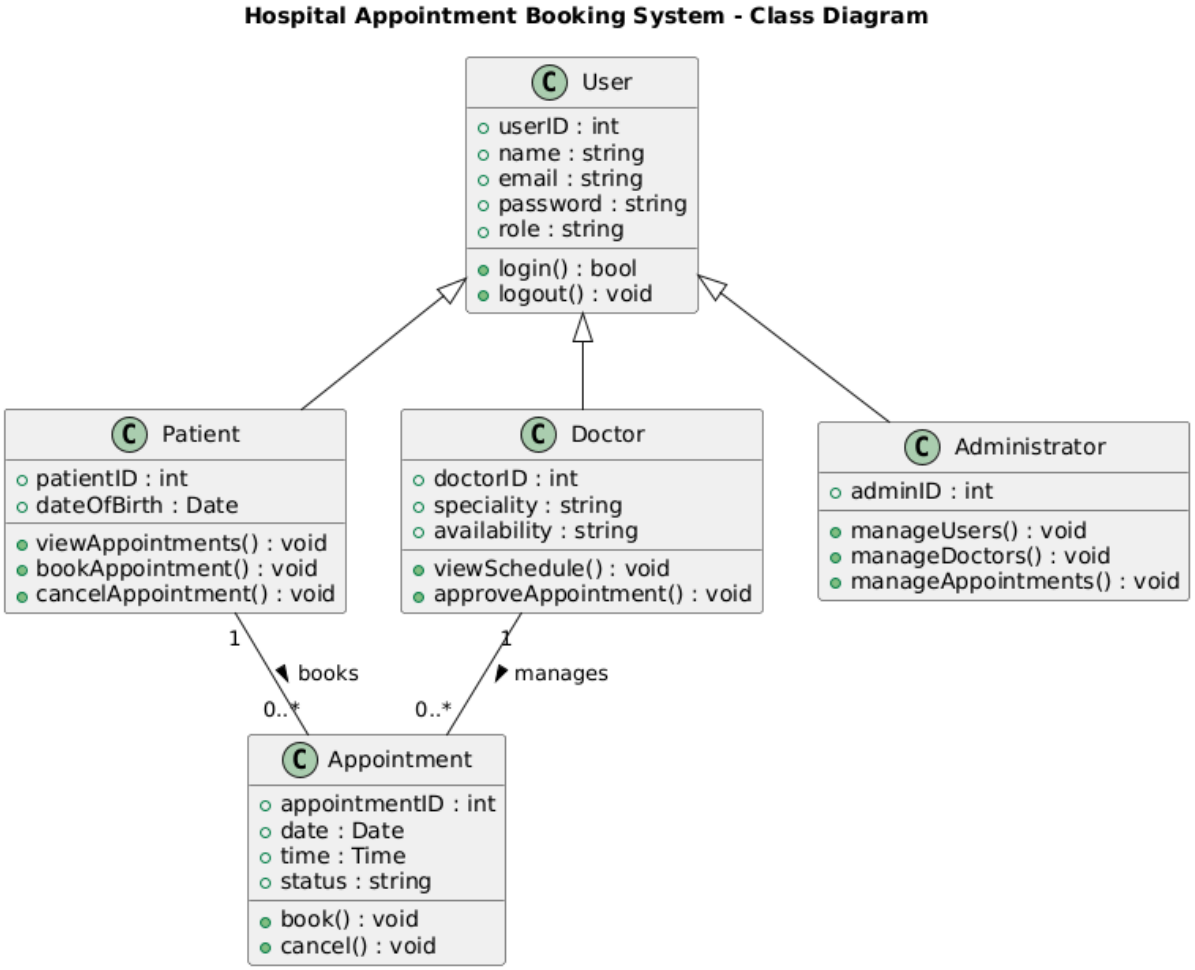
Adds, removes, or updates user information.

5. Modeling Diagrams

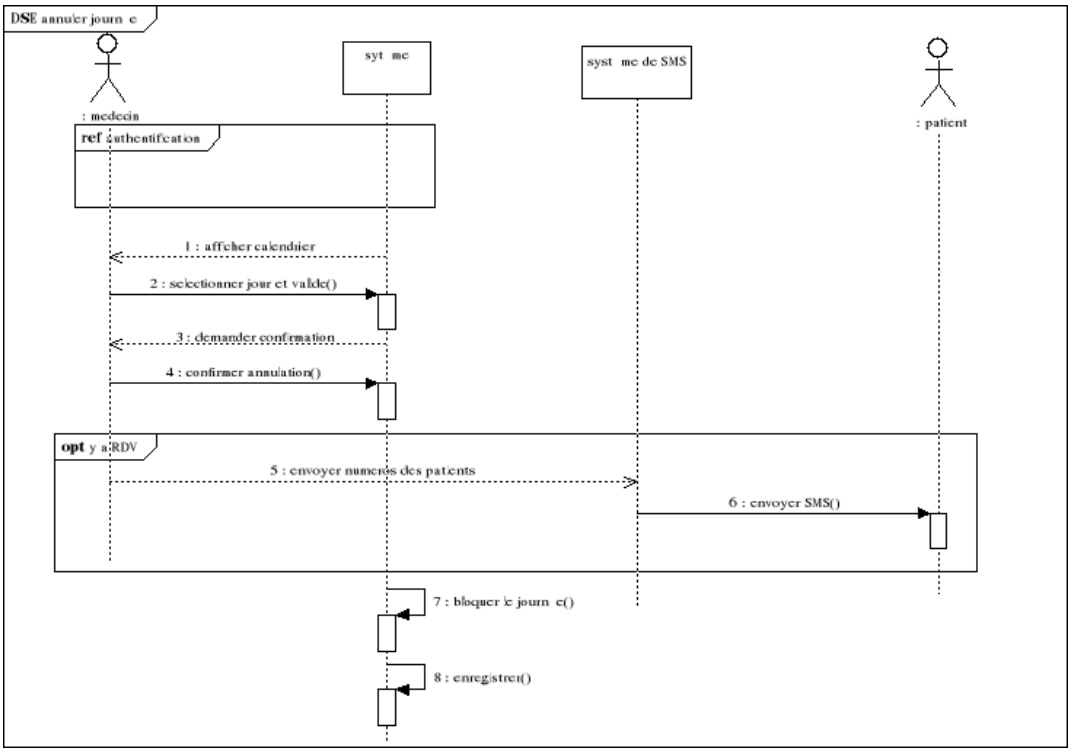
5.1 Use Case Diagram



5.2 Class Diagram



5.3 Sequence Diagram



6. External Interfaces

6.1 User Interfaces

Patient Portal: Appointment booking and history view.

Doctor Dashboard: Schedule management.

Admin Panel: Global system configuration.

6.2 Software Interfaces

RESTful API to connect frontend (React) and backend (Node.js).

Possible future integration with EHR systems via HL7 protocol.

6.3 Communication Interfaces

HTTPS protocol for all web interactions.

Secure SMTP for email notifications.

7. Other Requirements

7.1 Security and Privacy

Daily encrypted backups.

Automatic data deletion after X years of inactivity.

7.2 Compatibility

Compatible with Chrome, Firefox, Safari, and Edge.

7.3 Scalability

The system should support multiple hospitals or clinics in future expansions.