MediSync - Project Documentation

1. Business Requirements

The following list outlines the core functional requirements of the MediSync system:

1. **Patient Self-Registration:** Unregistered users must be able to create a patient account with personal details (Name, Date of Birth, Gender, Contact Info) and secure credentials.

2. **Doctor Management:** Admins can add new doctors to the system and manage their profile info

3. **Appointment Booking:** Patients must be able to search for available doctors by department and book open time slots. The system must prevent double-booking.

4. **Appointment Lifecycle:** The system must track the status of an appointment through defined stages: SCHEDULED, COMPLETED, CANCELLED, or NO_SHOW.

5. **Cancellation:** Patients and doctors can cancel appointments. A check must ensure appointments cannot be cancelled if they are already completed.

6. **Medical Records:** Upon completing an appointment, doctors must be able to generate a medical record containing a diagnosis, prescription, and treatment plan.

7. **Patient Health Profile:** The system must maintain a central profile for patients, including a list of Allergies (Medication, Food, etc.) to ensure safe treatment.

8. **Allergy Management:** The system must provide endpoints for performing CRUD actions for the Allergy entity

9. **Doctor Schedule Management:** Admins must be able to define recurring weekly working hours (e.g., Monday 09:00–17:00) for doctors to prevent bookings outside available shifts.

10. **User Management**: Admins must have the ability to activate or deactivate doctor and patient accounts (soft delete) to manage system access without losing data

11. **Department Management:** Admins can add, update and delete departments

12. **Login:** The system must provide endpoints for JWT authentication

2. MVP Features

Feature 1: Secure Authentication & Authorization

The system utilizes Spring Security and JWTs to secure the API. Users authenticate via the /api/auth/login endpoint to receive a token. Role guards and custom guards (AppointmentService ownership checks) ensure that users can only access data they are explicitly authorized to see (Patient A cannot view Patient B's history).

Feature 2: Appointment Scheduling

The scheduling functionality allows Admins to define working shifts (DoctorSchedule) for doctors. When a patient requests an appointment, the system validates that:

- The doctor is working on that day.
- The requested time falls within the doctor's start/end hours.
- The slot does not overlap with an existing booking.

Feature 3: Doctor & Department Management

Admins can onboard new medical staff, assign them to specific departments (e.g., Cardiology, Pediatrics), and configure their appointment duration (e.g., 15 mins vs. 30 mins). This directory allows patients to filter doctors based on their specific medical needs.

Feature 4: Clinical Record Management

Only Doctors with a SCHEDULED appointment can mark it as COMPLETED and create a MedicalRecord. This links the diagnosis and prescription directly to the specific visit, creating a permanent, queryable history for the patient.

Feature 5: Allergy Tracking

To support clinical decision-making, the system includes a robust Allergy Catalog. Admins can define global allergens (e.g., "Penicillin", "Peanuts"), and these can be linked to specific Patient profiles. This ensures doctors have immediate visibility into patient sensitivities during appointments.

3. Entity Descriptions and Relationships

1. User

Represents the base authentication entity for all system actors (Admin, Doctor, Patient). Stores login credentials and roles.

Relationships:

- One-to-One with Patient (A User can be a Patient).
- One-to-One with Doctor (A User can be a Doctor).

2. Patient

Stores specific profile data for patients (medical history, contact info, physical attributes).

Relationships:

- One-to-One with User (Links to login credentials).
- One-to-Many with Appointment (A patient can book multiple appointments).
- Many-to-Many with Allergy (A patient can have multiple allergies, and an allergy can affect multiple patients).
- Join Table: patients_allergies.

3. Doctor

Stores professional details for medical staff (specialization, license info).

Relationships:

- One-to-One with User (Links to login credentials).
- Many-to-One with Department (A doctor belongs to one specific department).
- One-to-Many with DoctorSchedule (A doctor has multiple weekly working shifts).
- One-to-Many with Appointment (A doctor attends to many appointments).
- One-to-One with Department (A doctor can be the "Head" of a department).

4. Department

Represents a hospital unit (e.g., Cardiology, Pediatrics) to organize doctors.

Relationships:

- One-to-Many with Doctor (Contains multiple doctors).
- One-to-One with Doctor (Has one Head of Department).

5. DoctorSchedule

Defines the recurring working hours for a doctor (e.g., "Mondays 09:00 - 17:00").

Relationships:

- Many-to-One with Doctor (Belongs to a specific doctor).

6. Appointment

Represents a scheduled meeting between a patient and a doctor at a specific time.

Relationships:

- Many-to-One with Patient (Booked by a patient).
- Many-to-One with Doctor (Assigned to a doctor).
- One-to-One with MedicalRecord (An appointment results in exactly one medical record upon completion).

7. MedicalRecord

Stores the clinical outcome of a visit, including diagnosis, prescription, and treatment plan.

Relationships:

- One-to-One with Appointment (Linked directly to the specific visit).

8. Allergy

A catalog of known allergens (e.g., Peanuts, Penicillin) used to tag patient profiles.

Relationships:

- Many-to-Many with Patient.