API Documentation: Hospital Management System

1. Introduction

This document provides a complete and detailed specification for the Hospital Management System API. It is intended for frontend developers and any other parties integrating with this system.

Base URL: All API endpoints are relative to the base URL. Example: https://api.yourdomain.com/api

Authentication: The API uses JWT (JSON Web Token) for authentication. For protected endpoints, the token must be included in the Authorization header of the request.

Header Format: Authorization: Bearer <YOUR_JWT_TOKEN>

Data Format: All request and response bodies are in application/json format, unless specified otherwise (e.g., for file uploads).

Error Responses: Errors are returned in a standardized JSON format.

Validation Error (400 Bad Request):

```json
{
  "errors": [
    "Error message 1.",
    "Error message 2."
  ]
}
```

General Error (404 Not Found, 401 Unauthorized, etc.):

```json
{
  "statusCode": 404,
  "message": "A descriptive error message."
}
```

2. Account Controller

Handles user registration and login functionalities.

2.1 Register New User

Description: Creates a new user account. By default, the user is assigned the "User" role.

Method & URL: POST /api/Account/Register

Authorization: Public (No authentication required).

Request Format: multipart/form-data

Body: RegisterDTO (see Appendix)

Response Format:

- 200 OK (Success):

```
JSON
{
  "messag": "Register Successfully",
  "model": { /* UserResponseDTO */ },
  "token": "string (JWT)",
  "role": "User"
}
```

- 400 Bad Request (Failure):

```
JSON
{ "errors": ["Email already exists."] }
```

2.2 Login User

Description: Authenticates a user and returns a JWT token upon successful login.

Method & URL: POST /api/Account/Login

Authorization: Public.

Request Format: application/json

Body: LoginDTO (see Appendix)

Response Format:

- 200 OK (Success):

```
JSON
{
  "message": "Login Successfully",
  "model": { /* UserResponseDTO */ },
  "token": "string (JWT)",
  "role": ["Role1", "Role2"]
}
```

- 400 Bad Request (Failure):

```
JSON
{ "errors": ["Invalid password."] }
```

3. Patient Controller

Manages patient-specific data and actions.

3.1 Create Patient Profile

Description: This endpoint has a dual function based on the caller's role:

For User role: Links a new patient profile to the currently authenticated user's account. Their role is then changed from "User" to "Patient".

For Admin or Staff roles: Creates a brand new user account and a patient profile simultaneously. This is used to register new patients who don't have an existing account.

Method & URL: POST /api/Patient/AddPatient

Authorization: User, Admin, Staff

Request Format: application/json

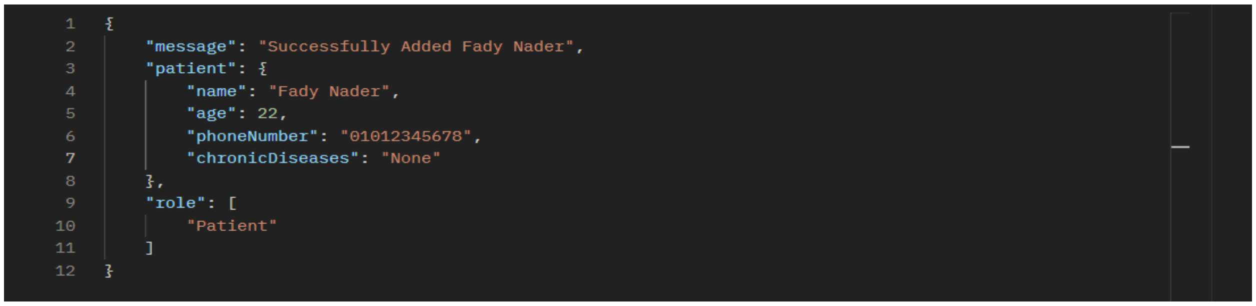
Body: PatientTopDTO (see Appendix)

If called by a User, provide the patientDTO object. The newPatientDTO object can be omitted.

If called by an Admin or Staff, you must provide both the patientDTO and the newPatientDTO objects.

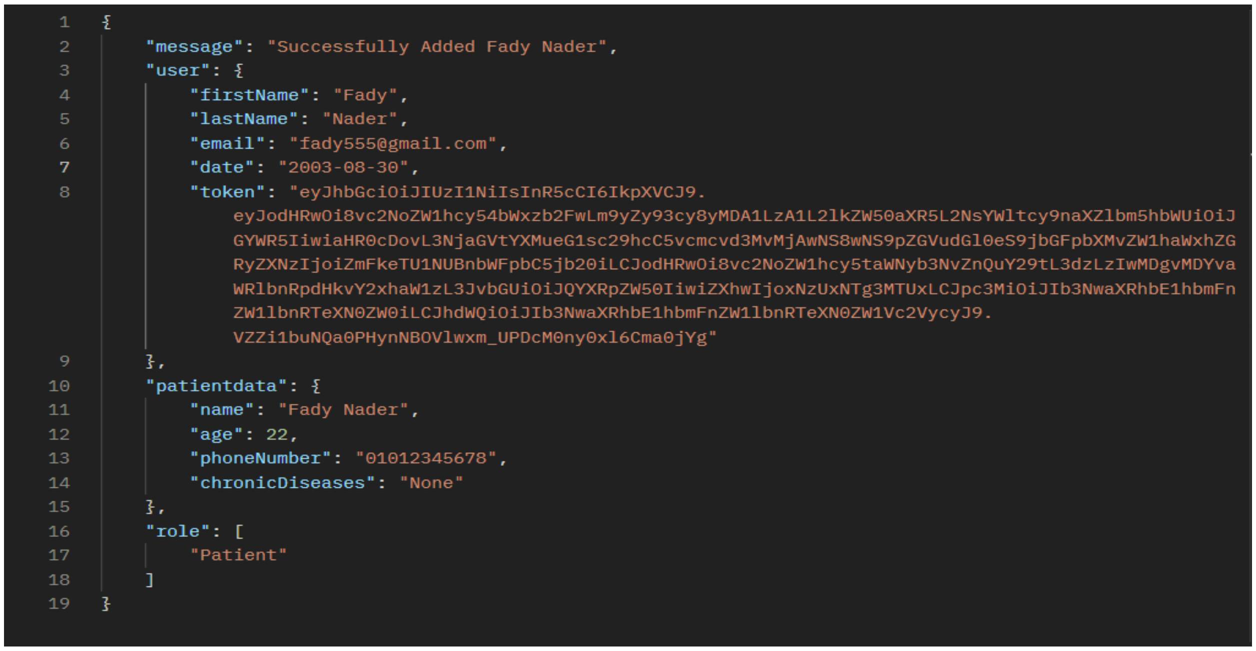
Response Format:

- 200 OK (Success - when called by a User):



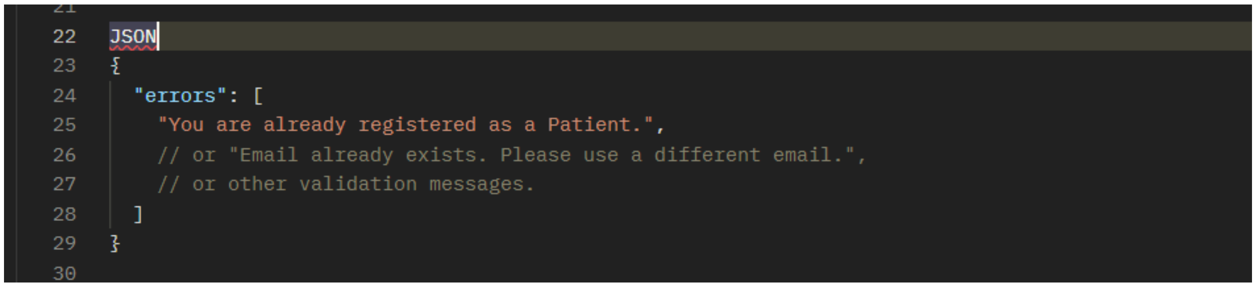
```json
{
    "message": "Successfully Added Fady Nader",
    "patient": {
        "name": "Fady Nader",
        "age": 22,
        "phoneNumber": "01012345678",
        "chronicDiseases": "None"
    },
    "role": [
        "Patient"
    ]
}
```

- 200 OK (Success - when called by Admin or Staff):



```
{
    "message": "Successfully Added Fady Nader",
    "user": {
        "firstName": "Fady",
        "lastName": "Nader",
        "email": "fady555@gmail.com",
        "date": "2003-08-30",
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
            eyJodHRwOi8vc2NoZW1hcy54bWxzb2FwLm9yZy93cy8yMDA1LzA1L2lkZW50aXR5L2NsYWltcy9naXZlbm5hbWUiOiJ
            GYWR5IiwiaHR0cDovL3NjaGVtYXMueG1sc29hcC5vcmcvd3MvMjAwNS8wNS9pZGVudGl0eS9jbGFpbXMvZW1haWxhZG
            RyZXNzIjoiZmFkeTU1NUBnbWFpbC5jb20iLCJodHRwOi8vc2NoZW1hcy5taWNyb3NvZnQuY29tL3dzLzIwMDgvMDYva
            WRlbnRpdHkvY2xhaW1zL3JvbGUiOiJQYXRpZW50IiwiZXhwIjoxNzUxNTg3MTUxLCJpc3MiOiJIb3NwaXRhbE1hbmFn
            ZW1lbnRTeXN0ZW0iLCJhdWQiOiJIb3NwaXRhbE1hbmFnZW1lbnRTeXN0ZW1Vc2VycyJ9.
            VZZi1buNQa0PHynNBOVlwxm_UPDcM0ny0xl6Cma0jYg"
    },
    "patientdata": {
        "name": "Fady Nader",
        "age": 22,
        "phoneNumber": "01012345678",
        "chronicDiseases": "None"
    },
    "role": [
        "Patient"
    ]
}
```

- 400 Bad Request (Failure):



```
JSON
{
  "errors": [
    "You are already registered as a Patient.",
    // or "Email already exists. Please use a different email.",
    // or other validation messages.
  ]
}
```

3.2 Get All Patients

Description: Retrieves a list of all patient profiles.

Method & URL: GET /api/Patient/GetAllPatients

Authorization: Admin, Staff

Request Format: None.

Response Format:

- 200 OK (Success): Array of PatientDTO
- 404 Not Found: { "statusCode": 404, "message": "No patients found." }

3.3 Get Patient by ID

Description: Retrieves a single patient's profile by their unique ID.

Method & URL: GET /api/Patient/GetPatientById/{id}

Authorization: Admin, Staff, Doctor

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): PatientDTO
- 404 Not Found: { "errors": ["Patient not found"] }

3.4 Update Patient Profile

Description: Updates an existing patient's profile information.

Method & URL: PUT /api/Patient/UpdatePatient/{id}

Authorization: Admin, Staff

Request Format: application/json

URL Parameter: id (integer)

Body: PatientDTO

Response Format:

- 200 OK (Success):

```json
{
  "message": "Successfully Updated Jane Doe",
  "patient": { /* PatientDTO */ }
}
```

3.5 Delete Patient

Description: Deletes a patient profile from the system.

Method & URL: DELETE /api/Patient/DeletePatient/{id}

Authorization: Admin

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success):

```
JSON
{
  "message": "Successfully Deleted Jane Doe",
  "patient": { /* PatientDTO */ }
}
```

3.6 Get Patient's Medical Reports

Description: Retrieves all medical reports associated with a specific patient.

Method & URL: GET /api/Patient/PatientMedicalReports/{id}

Authorization: Admin, Staff, Doctor, Patient

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): Array of PatientMedicalReportsDTO
- 404 Not Found: { "errors": ["No MedicalReport found for this Patient"] }

4. Doctor Controller

Manages doctor-specific data, including availability and appointments.

4.1 Add New Doctor

Description: Creates a new doctor profile and a corresponding user account with the "Doctor" role.

Method & URL: POST /api/Doctor/AddDoctor

Authorization: Admin

Request Format: application/json

Body: DoctorDTO (see Appendix)

Response Format:

- 200 OK (Success):

```
JSON
{
  "message": "Doctor added successfully",
  "doctor": { /* DoctorDTO */ },
  "role": ["Doctor"]
}
```

4.2 Get All Doctors

Description: Retrieves a list of all doctors, with an option to filter by specialization.

Method & URL: GET /api/Doctor/GetAllDoctors?SearchBySpecialization={specialization}

Authorization: Authenticated User.

Request Format:

Query Parameter (Optional): SearchBySpecialization (string)

Response Format:

- 200 OK (Success): Array of DoctorDTO
- 404 Not Found: { "errors": ["Not Found Doctors"] }

4.3 Get Doctor by ID

Description: Retrieves a single doctor's profile by their unique ID.

Method & URL: GET /api/Doctor/GetDoctorById/{id}

Authorization: Authenticated User.

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): DoctorDTO

4.4 Update Doctor

Description: Updates an existing doctor's information.

Method & URL: PUT /api/Doctor/UpdateDoctor/{id}

Authorization: Admin

Request Format: application/json

URL Parameter: id (integer)

Body: DoctorDTO

Response Format:

- 200 OK (Success):

```
JSON
{
  "message": "Successfully Updated Dr. Smith",
  "doctor": { /* DoctorDTO */ }
}
```

4.5 Delete Doctor

Description: Deletes a doctor's profile.

Method & URL: DELETE /api/Doctor/DeleteDoctor/{id}

Authorization: Admin

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success):

```
JSON
{
  "message": "Successfully Deleted Doctor with ID: 1 and Name: Dr. Smith"
}
```

4.6 Get Doctor's Appointments

Description: Retrieves all appointments assigned to a specific doctor.

Method & URL: GET /api/Doctor/DoctorAppointments/{id}

Authorization: Doctor

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): Array of DoctorAppointmentDetailsDTO
- 404 Not Found: { "errors": ["No appointments found for this doctor"] }

5. Appointment Controller

Manages the lifecycle of appointments.

5.1 Add New Appointment

Description: Allows a patient to book a new appointment. The initial status is "Pending".

Method & URL: POST /api/Appointment/AddAppointment

Authorization: Patient

Request Format: application/json

Body: AppointmentDTO (see Appendix)

Response Format:

- 200 OK (Success):

```
JSON
{
  "message": "Appointment Added Successfully ",
  "appointment": { /* AppointmentDetailsDTO */ }
}
```

5.2 Get All Appointments

Description: Retrieves a list of all appointments in the system.

Method & URL: GET /api/Appointment/GetAllAppointments

Authorization: Authenticated User.

Response Format:

- 200 OK (Success): Array of AppointmentDetailsDTO

5.3 Get Appointment by ID

Description: Retrieves the details of a single appointment.

Method & URL: GET /api/Appointment/GetAppointmentById/{id}

Authorization: Admin, Doctor, Patient

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): AppointmentDetailsDTO

5.4 Update Appointment

Description: Allows a patient to update the details of their appointment.

Method & URL: PUT /api/Appointment/UpdateAppointment/{id}

Authorization: Patient

Request Format: application/json

URL Parameter: id (integer)

Body: AppointmentDTO

Response Format:

- 200 OK (Success):

```JSON
{
  "message": "Appointment Update Successfully",
  "appointment": { /* AppointmentDetailsDTO */ }
}
```

5.5 Delete Appointment

Description: Allows a patient to cancel/delete their appointment.

Method & URL: DELETE /api/Appointment/DeleteAppointment/{id}

Authorization: Patient

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): { "message": "Appointment Deleted Successfully" }

5.6 Approve Appointment

Description: Allows a doctor or staff member to approve a "Pending" appointment.

Method & URL: POST /api/Appointment/ApproveAppointment/{id}

Authorization: Doctor, Staff

Request Format:

URL Parameter: id (integer)

Response Format:

200 OK (Success): { "message": "Appointment Approved Successfully", "appointment": { /* AppointmentDetailsDTO */ } }

5.7 Reject Appointment

Description: Allows a doctor or staff member to reject a "Pending" appointment.

Method & URL: POST /api/Appointment/RejectAppointment/{id}

Authorization: Doctor, Staff

Request Format:

URL Parameter: id (integer)

Response Format:

200 OK (Success): { "message": "Appointment Rejected Successfully", "appointment": { /* AppointmentDetailsDTO */ } }

6. Medical Report Controller

Manages medical reports created by doctors for patients.

6.1 Add Medical Report

Description: Allows a doctor to create a new medical report for a patient.

Method & URL: POST /api/MedicalReport/AddMedicalReport

Authorization: Doctor

Request Format: application/json

Body: MedicalReportDTO (see Appendix)

Response Format:

200 OK (Success): { "message": "MedicalReport Added Successfully", "medicalReport": { /* MedicalReportDetailsDTO */ } }

6.2 Get All Medical Reports

Description: Retrieves a list of all medical reports.

Method & URL: GET /api/MedicalReport/GetAllMedicalReport

Authorization: Doctor, Staff

Response Format:

- 200 OK (Success): Array of MedicalReportDetailsDTO

6.3 Get Medical Report by ID

Description: Retrieves a single medical report by its unique ID.

Method & URL: GET /api/MedicalReport/GetMedicalReportById/{id}

Authorization: Doctor, Staff

Request Format:

URL Parameter: id (integer)

Response Format:

- 200 OK (Success): MedicalReportDetailsDTO

6.4 Update Medical Report

Description: Allows a doctor to update an existing medical report.

Method & URL: PUT /api/MedicalReport/UpdateMedicalReport/{id}

Authorization: Doctor

Request Format: application/json

URL Parameter: id (integer)

Body: MedicalReportDTO

Response Format:

200 OK (Success): { "message": "MedicalReport Updated Successfully.", "medicalReport": { /* MedicalReportDetailsDTO */ } }

6.5 Delete Medical Report

Description: Deletes a medical report from the system.

Method & URL: DELETE /api/MedicalReport/DeleteMedicalReport/{id}

Authorization: Admin

Request Format:

URL Parameter: id (integer)

Response Format:

200 OK (Success): { "message": "MedicalReport Deleted Successfully." }

7. Dashboard Controller

Provides administrative statistics.

7.1 Get System Stats

Description: Retrieves overall statistics for the system.

Method & URL: GET /api/Dashboard/stats

Authorization: Admin

Response Format:

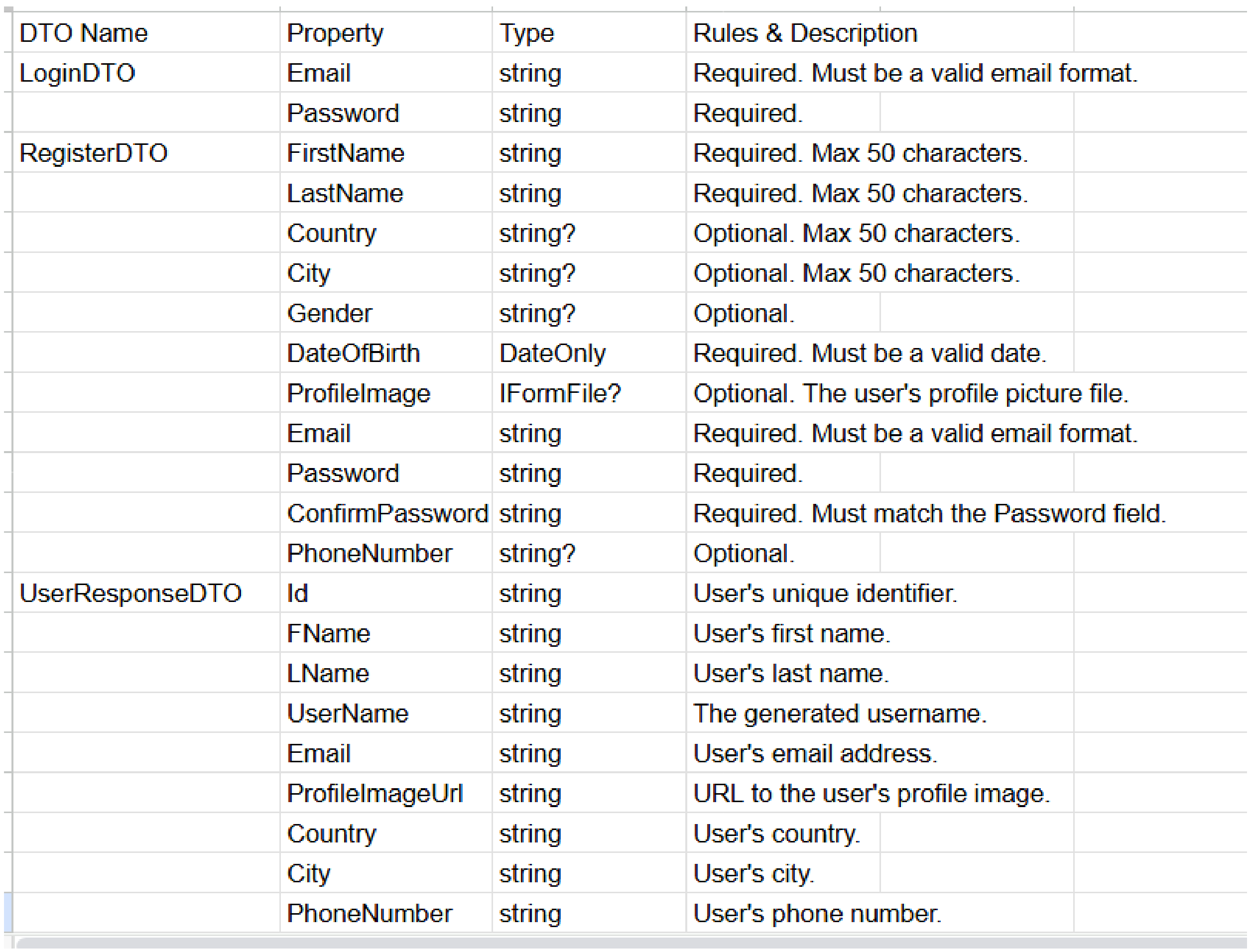
- 200 OK (Success): StatsDTO

```json
JSON
{
  "totalPatients": 10,
  "totalDoctors": 5,
  "totalAppointments": 25,
  "totalMedicalReports": 20
}
```

Appendix: Data Transfer Objects (DTOs)

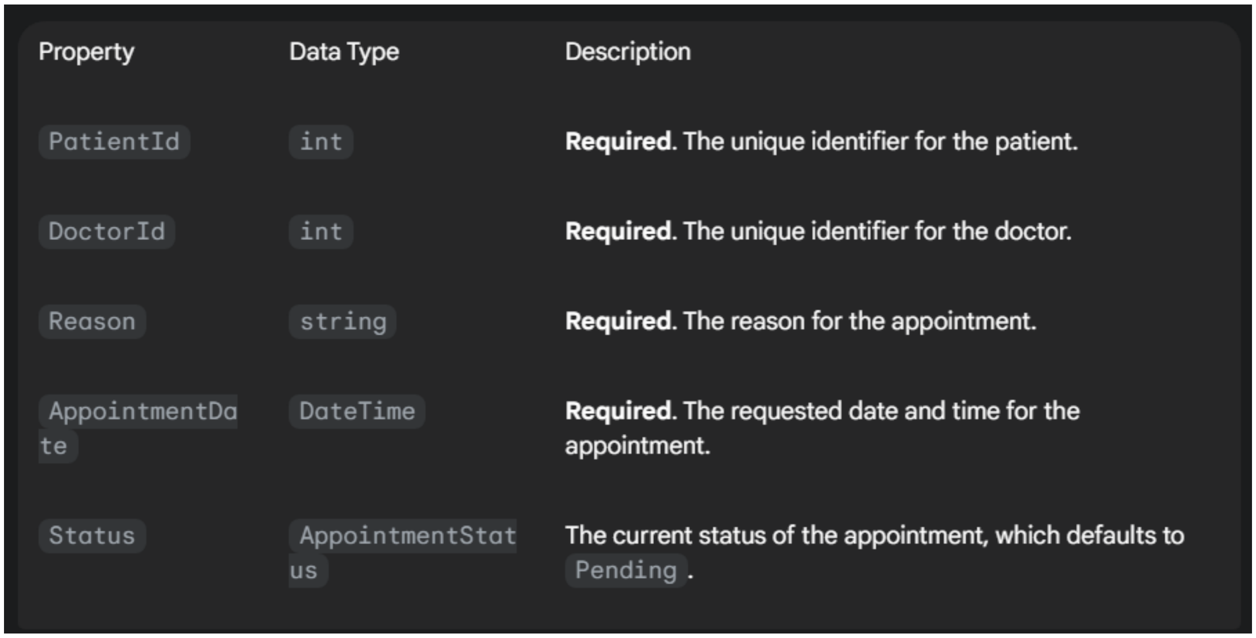
Identity DTOs

These DTOs are used for authentication and user management processes.



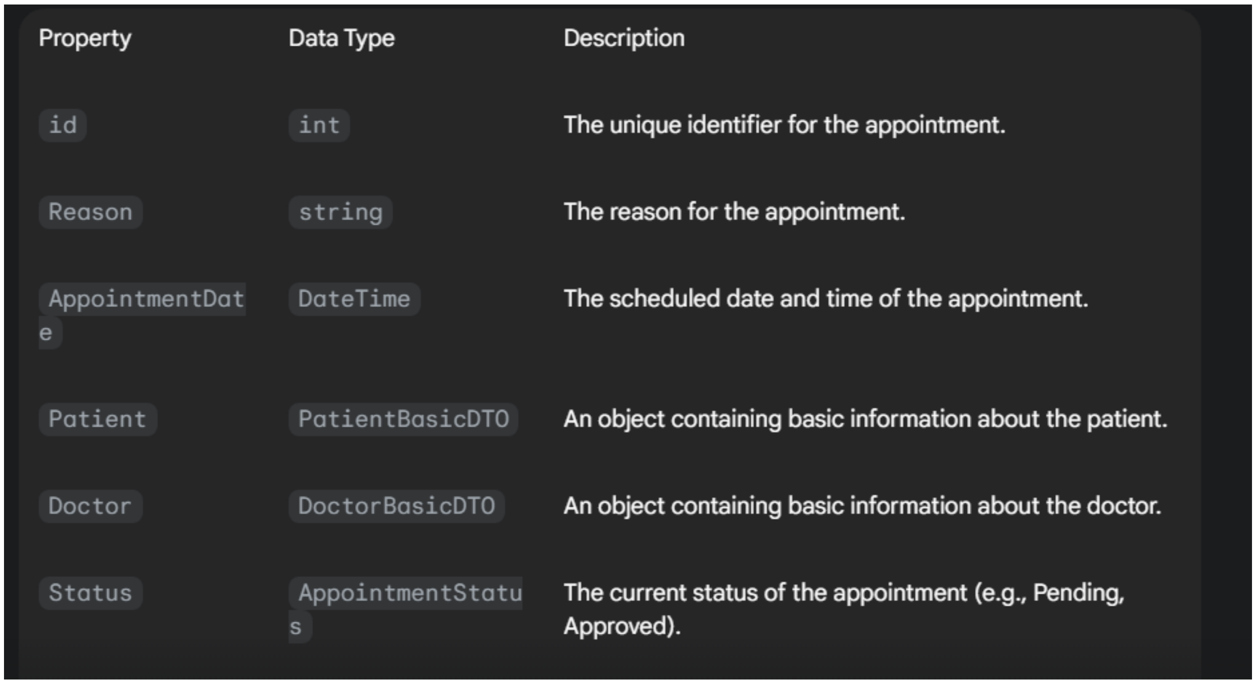
DTO Name	Property	Type	Rules & Description
LoginDTO	Email	string	Required. Must be a valid email format.
	Password	string	Required.
RegisterDTO	FirstName	string	Required. Max 50 characters.
	LastName	string	Required. Max 50 characters.
	Country	string?	Optional. Max 50 characters.
	City	string?	Optional. Max 50 characters.
	Gender	string?	Optional.
	DateOfBirth	DateOnly	Required. Must be a valid date.
	ProfileImage	IFormFile?	Optional. The user's profile picture file.
	Email	string	Required. Must be a valid email format.
	Password	string	Required.
	ConfirmPassword	string	Required. Must match the Password field.
	PhoneNumber	string?	Optional.
UserResponseDTO	Id	string	User's unique identifier.
	FName	string	User's first name.
	LName	string	User's last name.
	UserName	string	The generated username.
	Email	string	User's email address.
	ProfileImageUrl	string	URL to the user's profile image.
	Country	string	User's country.
	City	string	User's city.
	PhoneNumber	string	User's phone number.

AppointmentDTO



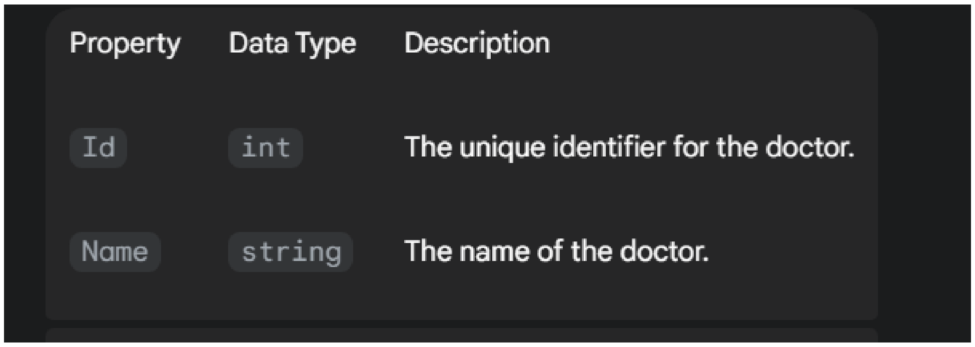
Property	Data Type	Description
`PatientId`	`int`	**Required.** The unique identifier for the patient.
`DoctorId`	`int`	**Required.** The unique identifier for the doctor.
`Reason`	`string`	**Required.** The reason for the appointment.
`AppointmentDate`	`DateTime`	**Required.** The requested date and time for the appointment.
`Status`	`AppointmentStatus`	The current status of the appointment, which defaults to `Pending`.

AppointmentDetailsDTO



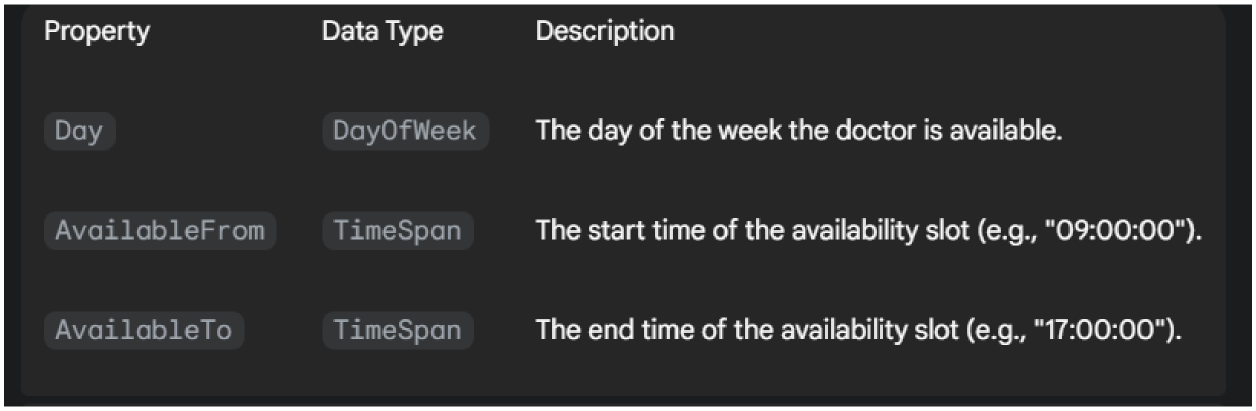
Property	Data Type	Description
`id`	`int`	The unique identifier for the appointment.
`Reason`	`string`	The reason for the appointment.
`AppointmentDate`	`DateTime`	The scheduled date and time of the appointment.
`Patient`	`PatientBasicDTO`	An object containing basic information about the patient.
`Doctor`	`DoctorBasicDTO`	An object containing basic information about the doctor.
`Status`	`AppointmentStatus`	The current status of the appointment (e.g., Pending, Approved).

DoctorBasicDTO



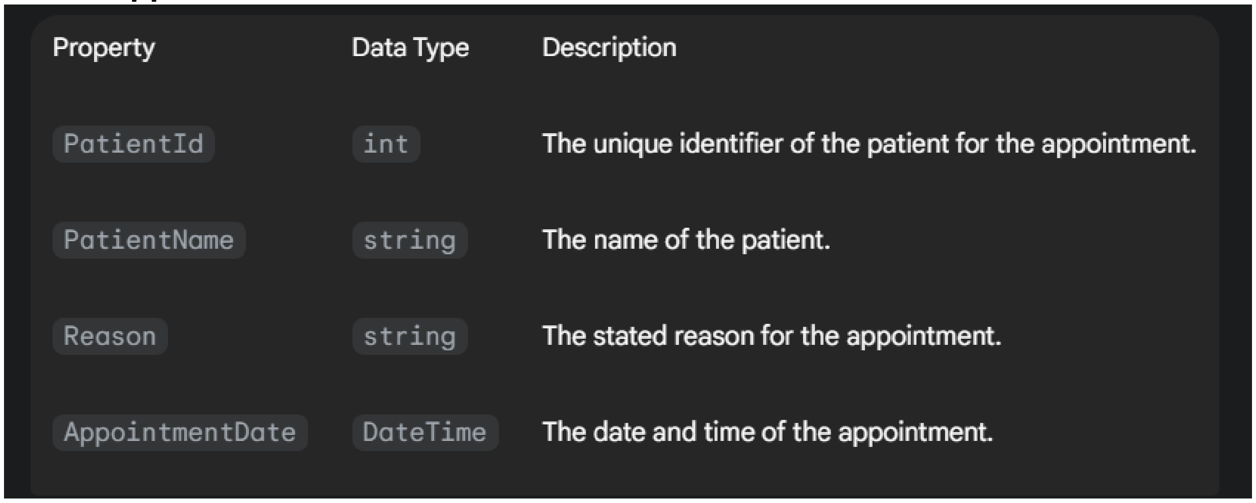
Property	Data Type	Description
`Id`	`int`	The unique identifier for the doctor.
`Name`	`string`	The name of the doctor.

DoctorAvailabilityDTO



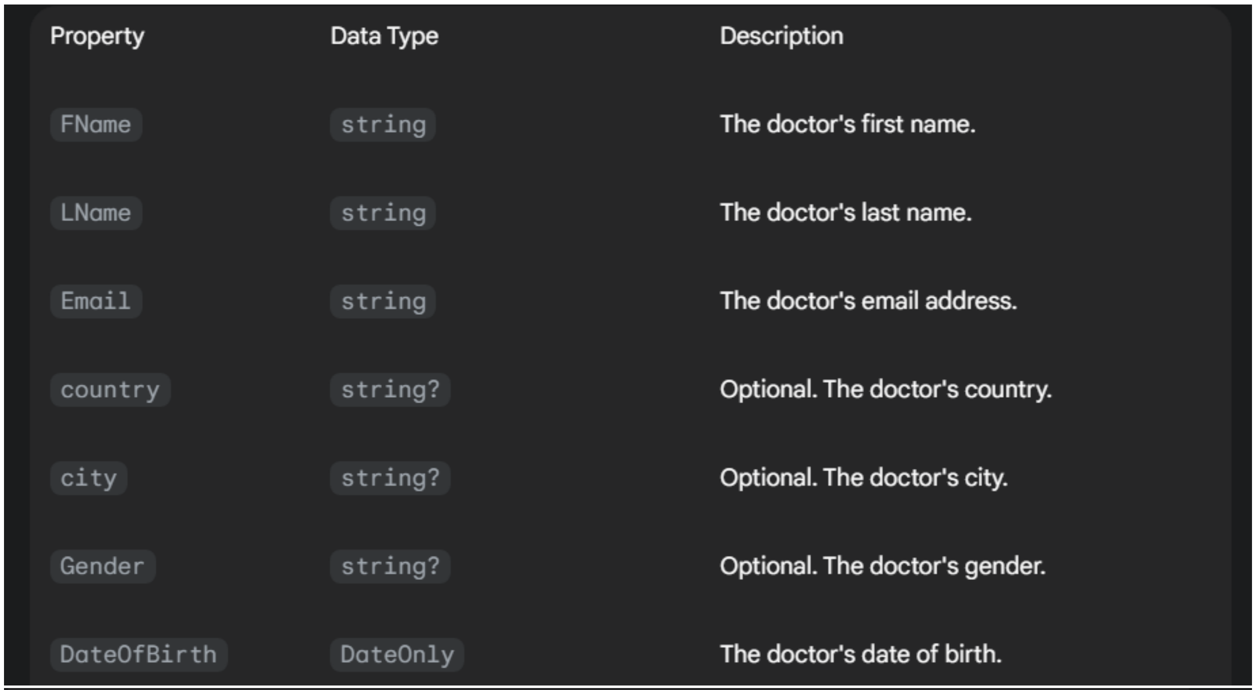
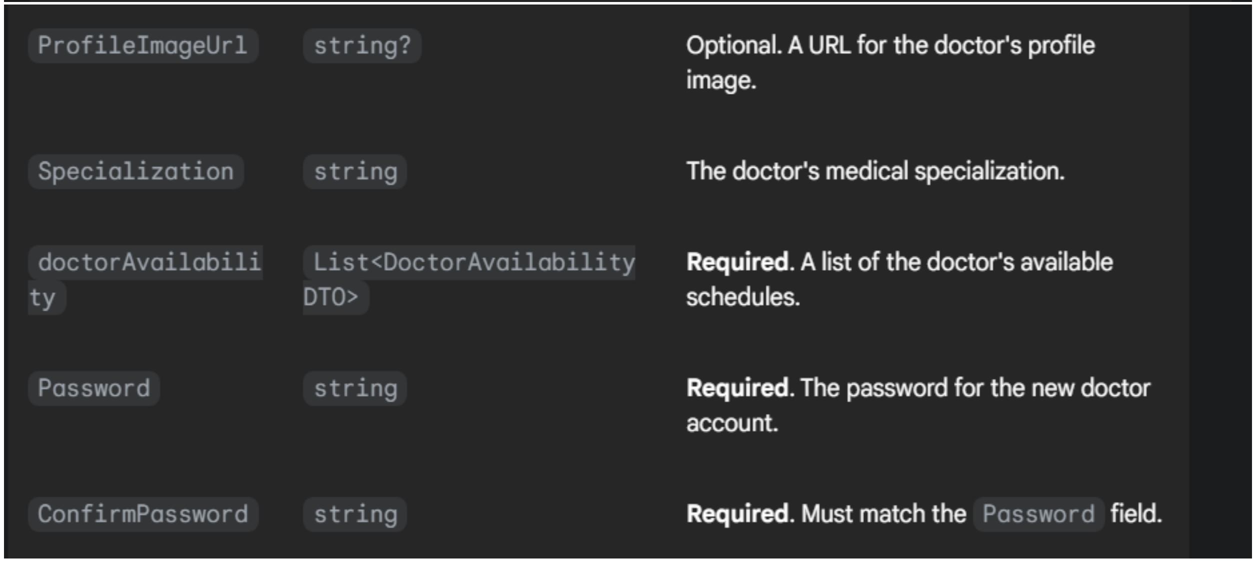
Property	Data Type	Description
`Day`	`DayOfWeek`	The day of the week the doctor is available.
`AvailableFrom`	`TimeSpan`	The start time of the availability slot (e.g., "09:00:00").
`AvailableTo`	`TimeSpan`	The end time of the availability slot (e.g., "17:00:00").

DoctorAppointmentDetailsDTO



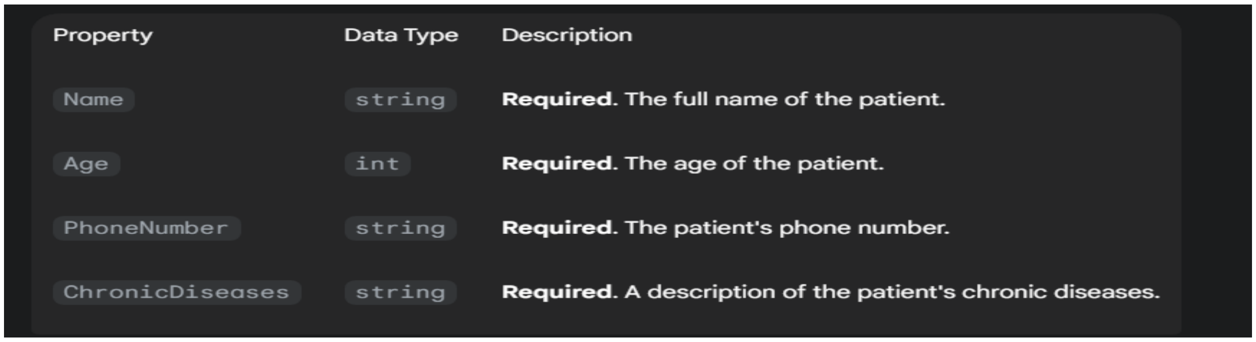
Property	Data Type	Description
`PatientId`	`int`	The unique identifier of the patient for the appointment.
`PatientName`	`string`	The name of the patient.
`Reason`	`string`	The stated reason for the appointment.
`AppointmentDate`	`DateTime`	The date and time of the appointment.

DoctorDTO

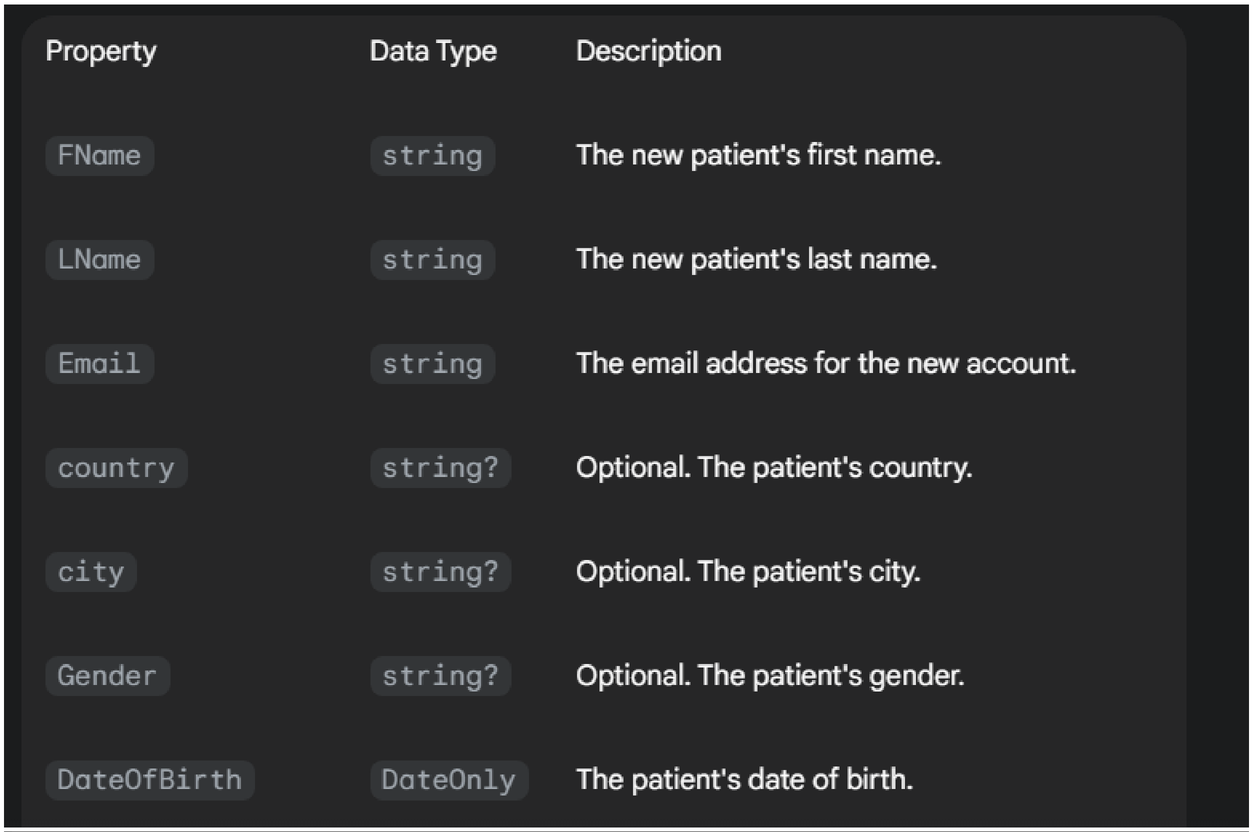
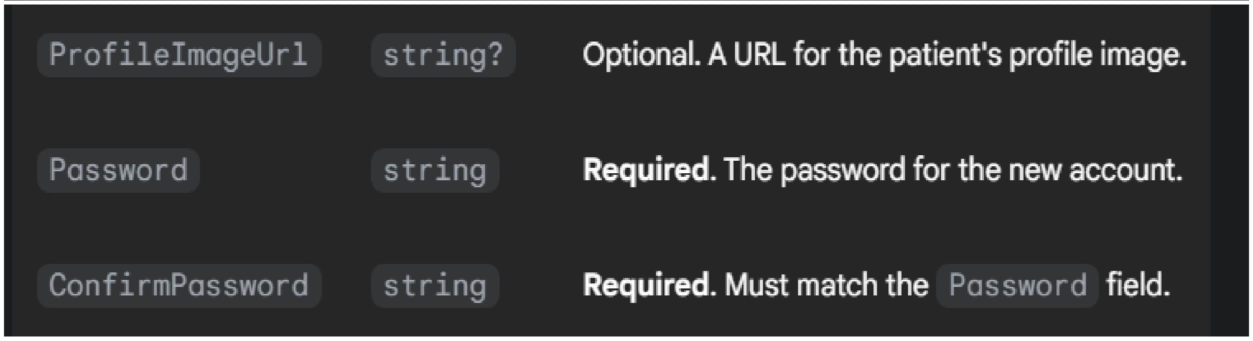
Property	Data Type	Description
`FName`	`string`	The doctor's first name.
`LName`	`string`	The doctor's last name.
`Email`	`string`	The doctor's email address.
`country`	`string?`	Optional. The doctor's country.
`city`	`string?`	Optional. The doctor's city.
`Gender`	`string?`	Optional. The doctor's gender.
`DateOfBirth`	`DateOnly`	The doctor's date of birth.
`ProfileImageUrl`	`string?`	Optional. A URL for the doctor's profile image.
`Specialization`	`string`	The doctor's medical specialization.
`doctorAvailability`	`List<DoctorAvailabilityDTO>`	**Required**. A list of the doctor's available schedules.
`Password`	`string`	**Required**. The password for the new doctor account.
`ConfirmPassword`	`string`	**Required**. Must match the `Password` field.

PatientDTO



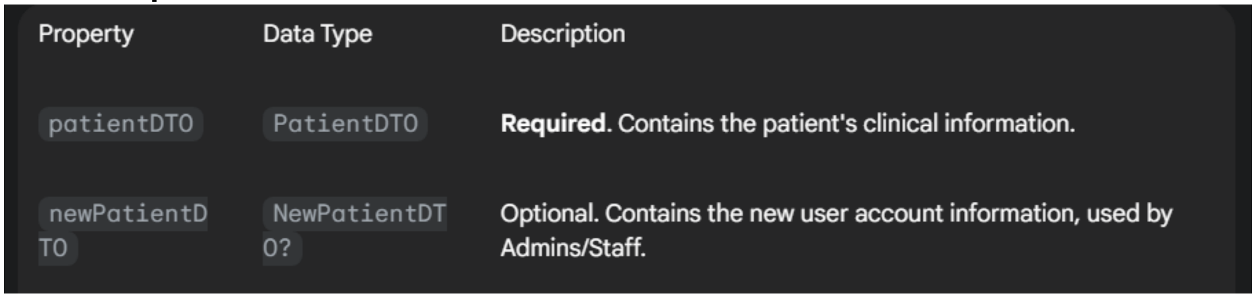
Property	Data Type	Description
`Name`	`string`	**Required**. The full name of the patient.
`Age`	`int`	**Required**. The age of the patient.
`PhoneNumber`	`string`	**Required**. The patient's phone number.
`ChronicDiseases`	`string`	**Required**. A description of the patient's chronic diseases.

NewPatientDTO

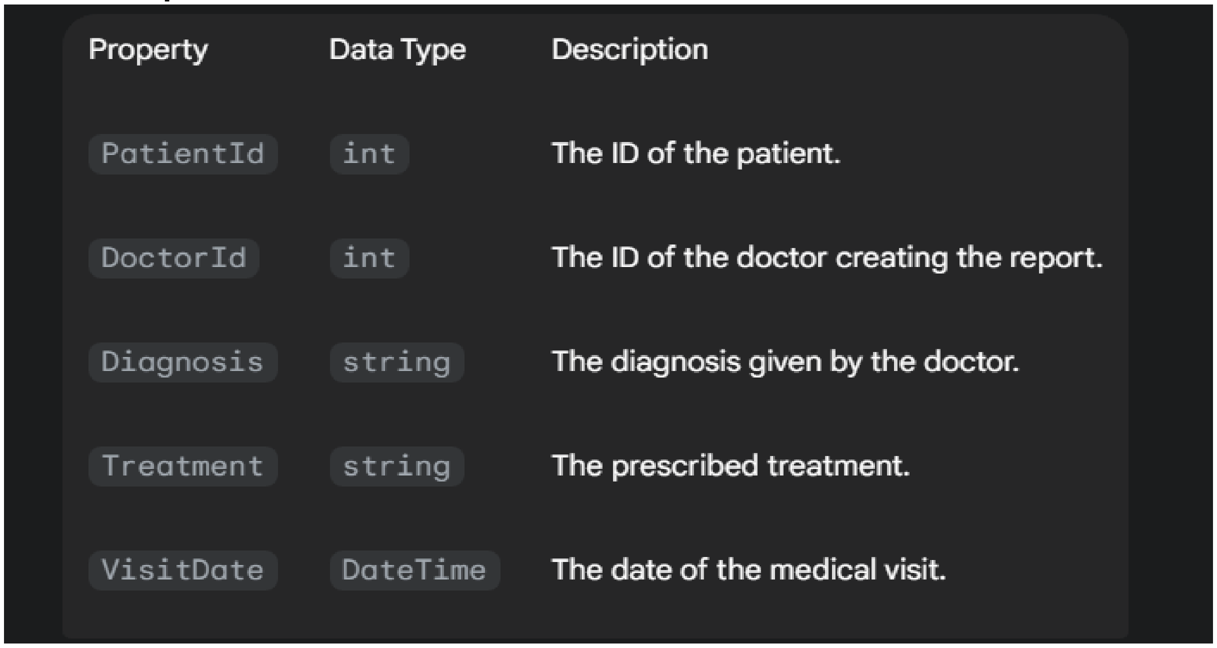
Property	Data Type	Description
`FName`	`string`	The new patient's first name.
`LName`	`string`	The new patient's last name.
`Email`	`string`	The email address for the new account.
`country`	`string?`	Optional. The patient's country.
`city`	`string?`	Optional. The patient's city.
`Gender`	`string?`	Optional. The patient's gender.
`DateOfBirth`	`DateOnly`	The patient's date of birth.
`ProfileImageUrl`	`string?`	Optional. A URL for the patient's profile image.
`Password`	`string`	**Required**. The password for the new account.
`ConfirmPassword`	`string`	**Required**. Must match the `Password` field.

PatientTopDTO



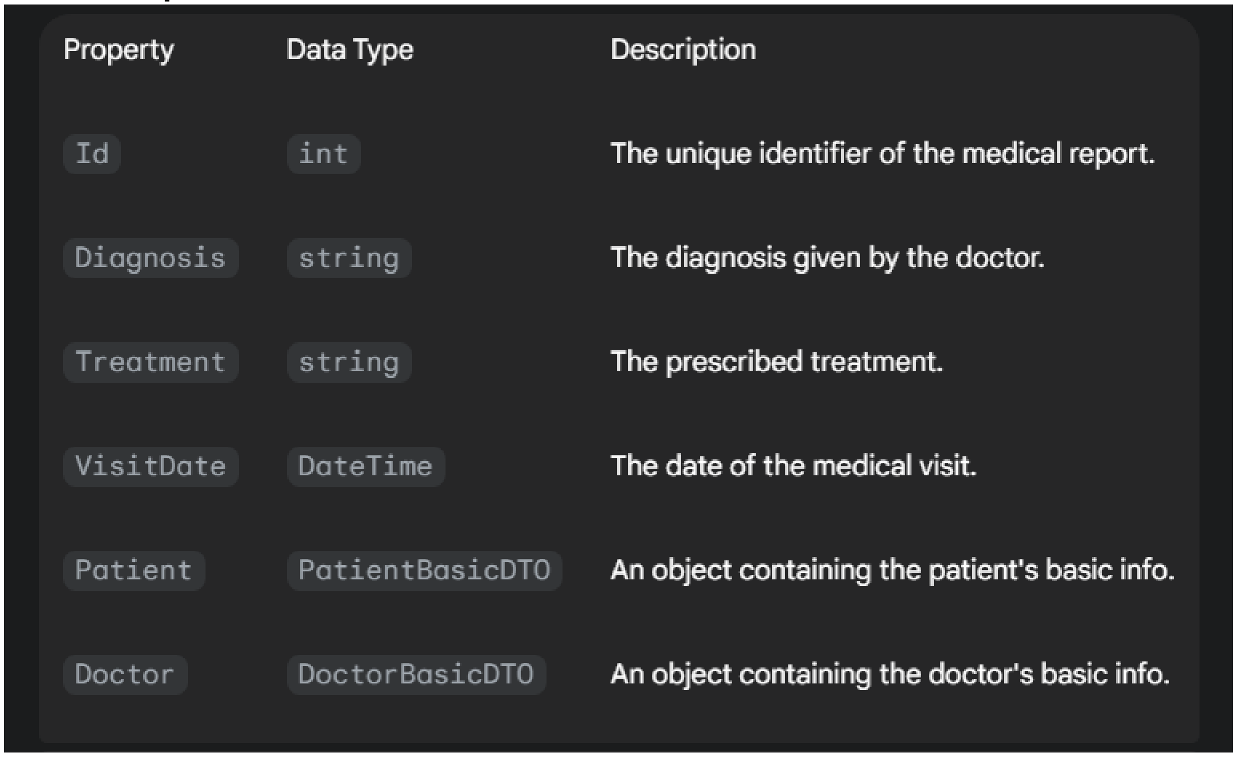
Property	Data Type	Description
`patientDTO`	`PatientDTO`	**Required**. Contains the patient's clinical information.
`newPatientDTO`	`NewPatientDTO?`	Optional. Contains the new user account information, used by Admins/Staff.

MedicalReportDTO



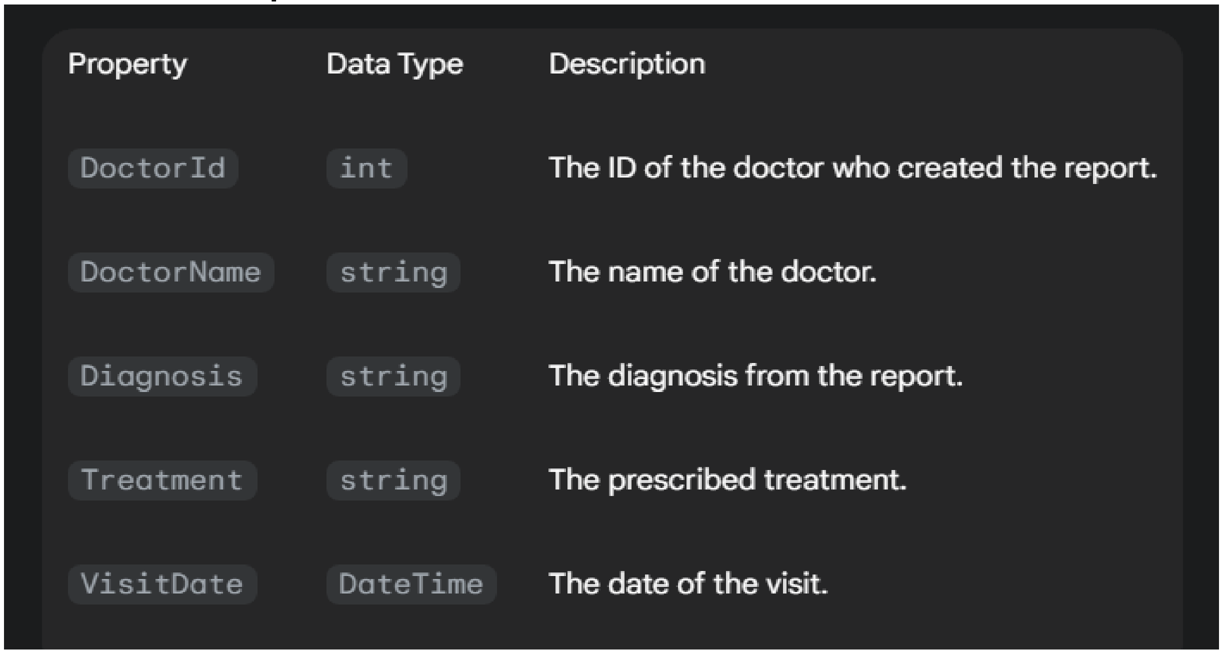
Property	Data Type	Description
`PatientId`	`int`	The ID of the patient.
`DoctorId`	`int`	The ID of the doctor creating the report.
`Diagnosis`	`string`	The diagnosis given by the doctor.
`Treatment`	`string`	The prescribed treatment.
`VisitDate`	`DateTime`	The date of the medical visit.

MedicalReportDetailsDTO



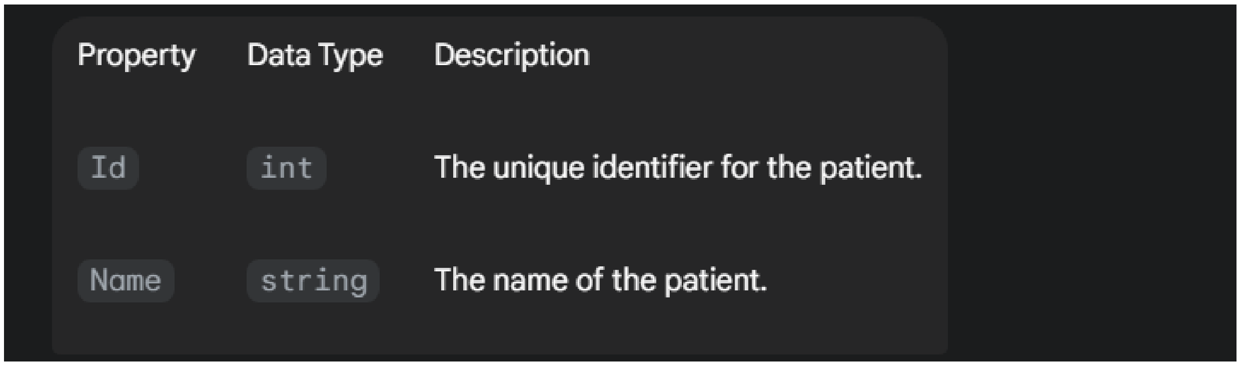
Property	Data Type	Description
`Id`	`int`	The unique identifier of the medical report.
`Diagnosis`	`string`	The diagnosis given by the doctor.
`Treatment`	`string`	The prescribed treatment.
`VisitDate`	`DateTime`	The date of the medical visit.
`Patient`	`PatientBasicDTO`	An object containing the patient's basic info.
`Doctor`	`DoctorBasicDTO`	An object containing the doctor's basic info.

PatientMedicalReportsDTO



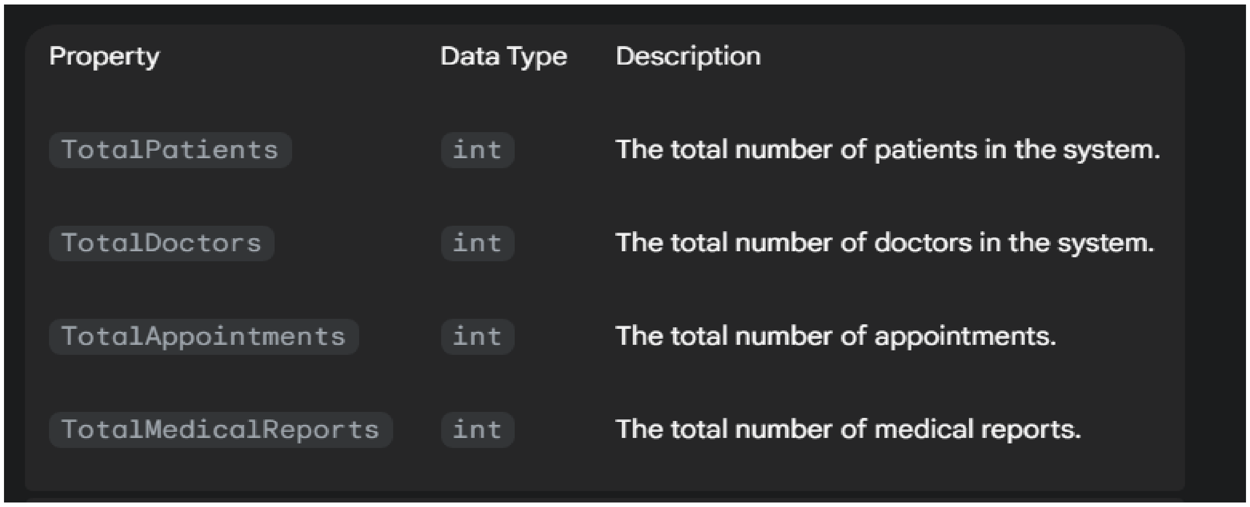
Property	Data Type	Description
`DoctorId`	`int`	The ID of the doctor who created the report.
`DoctorName`	`string`	The name of the doctor.
`Diagnosis`	`string`	The diagnosis from the report.
`Treatment`	`string`	The prescribed treatment.
`VisitDate`	`DateTime`	The date of the visit.

PatientBasicDTO



Property	Data Type	Description
`Id`	`int`	The unique identifier for the patient.
`Name`	`string`	The name of the patient.

StatsDTO



Property	Data Type	Description
`TotalPatients`	`int`	The total number of patients in the system.
`TotalDoctors`	`int`	The total number of doctors in the system.
`TotalAppointments`	`int`	The total number of appointments.
`TotalMedicalReports`	`int`	The total number of medical reports.