

# Doctor Booking System API

# Table of Contents

## **1. Introduction**

- Project Overview
- Description
- Important Notes

## **2. Database Schema**

## **3. Authentication**

## **4. Appointment Module**

- Create Appointment
- Update Appointment
- Doctor Update Status of Appointment
- View List of Appointments

## **5. Doctor Module**

- Create Doctor
- Update Doctor Info
- View Appointments
- Update Status of Appointment

## **6. Patient / User Module**

- Create Patient
- Update Patient Info
- Create Appointment
- Update Appointment
- View List of Their Appointments

# Introduction

## 1. Project Overview

Using the Doctor's Booking System, a Laravel-based online application, patients can schedule appointments with doctors. It offers a platform for effectively scheduling appointments and serves patients as well as doctors.

## 2. Description

The Doctor Booking System's APIs are described in this documentation along with how they work. It covers a number of modules, such as patient/user interactions, appointment management, doctor-specific features, and authentication.

## 3. Important Notes

- The project is able to be accessed on GitHub.
- The project follows to a particular design pattern that is described in the README.md file.
- Within the project, there are detailed database schema designs available.

# Database Schema



## 1) User Table

| doctor users |                     |
|--------------|---------------------|
| id           | bigint(20) unsigned |
| first_name   | varchar(255)        |
| last_name    | varchar(255)        |
| email        | varchar(255)        |
| phone        | int(10)             |
| image        | varchar(255)        |
| DOB          | date                |
| address      | varchar(255)        |
| gender       | varchar(255)        |
| height       | double(8,2)         |
| weight       | double(8,2)         |
| password     | varchar(255)        |
| device_token | varchar(255)        |
| deleted_at   | timestamp           |
| created_at   | timestamp           |
| updated_at   | timestamp           |

## 2) Doctor Table

|   | doctor doctors                         |
|---|--|
| 🔑 | id : bigint(20) unsigned               |
| 📄 | first_name : varchar(255)              |
| 📄 | last_name : varchar(255)               |
| 🔑 | email : varchar(255)                   |
| # | phone : int(10)                        |
| 📄 | image : varchar(255)                   |
| 📄 | speciality : varchar(255)              |
| 📄 | languages : varchar(255)               |
| 📄 | education : varchar(255)               |
| 📄 | password : varchar(255)                |
| 📅 | DOB : date                             |
| 🔍 | gender : enum('Male','Female','Other') |
| 📄 | device_token : varchar(255)            |
| 📅 | created_at : timestamp                 |
| 📅 | updated_at : timestamp                 |

## 3) Appointment table

|   | doctor appointments                                      |
|---|--|
| 🔑 | id : bigint(20) unsigned                                 |
| # | patient_id : int(11)                                     |
| # | doctor_id : int(11)                                      |
| 📅 | appointment_date : date                                  |
| 📅 | appointment_start_time : time                            |
| 📅 | appointment_end_time : time                              |
| 🔍 | status : enum('RSVP','approved','cancelled','postponed') |
| 📅 | created_at : timestamp                                   |
| 📅 | updated_at : timestamp                                   |

## Authentication

To access the APIs, users must be authenticated. Unauthorized access is restricted.

### **Doctor Model:**

- When a doctor registers or logs in, generate a unique device token for that doctor.
- Store this device token in the doctor's database record.

### **Patient Model:**

- When a patient registers or logs in, generate a unique device token for that patient.
- Store this device token in the patient's database record.

### **API Authentication:**

- For each API request, check the request headers to ensure that a device token is included.
- Verify the device token against the records in the doctor and patient models to determine if the user is a doctor or a patient.
- If the device token matches a doctor's token, allow access to doctor-specific API endpoints.
- If the device token matches a patient's token, allow access to patient-specific API endpoints.
- If the device token does not match any records, reject the request as unauthorized.

## Appointment Module

The Appointment Module is designed for patients to book and manage appointments. It includes four main APIs:

### 1. Create Appointment

**Endpoint:** /api/ addAppointment/

**Method:** POST

**Description:**

Allows patients to create appointments.

Prevents two patients from booking an appointment at the same date and time.

Appointments are RSVP until approved by the doctor.

Request Parameters

**patient\_id (integer):** ID of the patient creating the appointment.

**doctor\_id (integer):** ID of the doctor the patient wants to book an appointment with.

**appointment\_date (string, format: 'Y-m-d'):** Date of the appointment.

**Appointment\_start\_time (string, format: 'H:i'):** Start time of the appointment.

**appointment\_end\_time will be added automatically of 30 min**

**Response:**

Success (HTTP Status Code: 201 Created): Returns a success message upon successful appointment creation.

Error (HTTP Status Code: 400 Bad Request or 409 Conflict): Returns an error message if the appointment conflicts with an existing one.

### 2. Update Appointment

**Endpoint:** /api/ updateAppointment

**Method:** POST

**Description:**

Allows patients to update the date and time of their appointments.

Ensures that the new date and time do not conflict with existing appointments.

Request Parameters

**appointment\_date (string, format: 'Y-m-d'):** New date for the appointment.

**appointment\_start\_time (string, format: 'H:i'):** New start time for the appointment.

**appointment\_end\_time will be added automatically of 30 min**

### **Response**

Success (HTTP Status Code: 200 OK): Returns a success message upon successful update.

Error (HTTP Status Code: 400 Bad Request or 409 Conflict): Returns an error message if the new date and time conflict with existing appointments.

## **3. Doctor Update Status of Appointment**

**Endpoint:** /api/ changeStatus/{id}

**Method:** POST

### **Description:**

Allows doctors to update the status of appointments they manage.

Status options: 'approved', 'cancelled', 'postponed'.

Request Parameters

**id (integer, path):** ID of the appointment to update.

**status (string):** New status for the appointment.

**appointment\_date (date, format: 'Y-m-d') [Required if status is 'postponed']:** New date for the appointment.

**appointment\_start\_time (time, format: 'H:i') [Required if status is 'postponed']:** New start time for the appointment.

**appointment\_end\_time will be added automatically of 30 min**

### **Response**

Success (HTTP Status Code: 200 OK): Returns a success message upon successful status update.

Error (HTTP Status Code: 401 Unauthorized or 404 Not Found): Returns an error message if the appointment or doctor is not valid.

## **4. View All List of Appointments**

**Endpoint:** /api/ listAllAppointments

**Method:** POST

### **Description:**

Allows patients to view a list of their appointments.

Provides filtering by date.

Returns user-specific appointment information.



## **Request Parameters**

**date (string, format: 'Y-m-d'):** Optional date for filtering appointments.

## **Response**

Success (HTTP Status Code: 200 OK): Returns a list of appointments based on the filter criteria.

Error (HTTP Status Code: 401 Unauthorized or 404 Not Found): Returns an error message if the user or appointments are not found.

## Patient / User Module

The Patient/User Module allows users to interact with the system. It includes five main APIs:

### 1. Create Patient

**Endpoint:** /api/ adminSignup

**Method:** POST

**Description:**

Allows the creation of patient profiles.

Requires essential patient information such as name, contact details, date of birth, and more.

**Request Parameters**

**first\_name (string):** First name of the patient.

**last\_name (string):** Last name of the patient.

**email (string, format: email):** Email address of the patient.

**phone (string):** Phone number of the patient.

**DOB (date, format: 'Y-m-d'):** Date of birth of the patient.

**gender (string):** Gender of the patient.

**device\_token (string):** Device token for authentication.

**image (file):** Profile image of the patient.

**Response**

Success (HTTP Status Code: 201 Created): Returns a success message upon successful patient profile creation.

Error (HTTP Status Code: 400 Bad Request): Returns an error message if the patient profile creation fails.

### 2. Update Patient Info

**Endpoint:** /api/ adminUpdate

**Method:** POST

**Description:**

Allows patients to update their profile information.

**Request Parameters**

Any parameters that need to be updated (e.g., first\_name, last\_name, etc.).

**Response**

Success (HTTP Status Code: 200 OK): Returns a success message upon successful profile update.

Error (HTTP Status Code: 400 Bad Request): Returns an error message if the update fails.

### 3. User Login

**Endpoint:** /api/userLogin

**Method:** POST

**Description:**

Allows users (patients) to log in using their email and password.

**Request Parameters:**

**email (string, format: email) [Required]:** Email address of the user.

**password (string) [Required]:** User's password.

**Response:**

Success (HTTP Status Code: 200 OK):

Returns a success message along with an access token upon successful login.

Error (HTTP Status Code: 401 Unauthorized):

Returns an error message if the provided credentials are invalid.

Error (HTTP Status Code: 400 Bad Request):

Returns an error message if the required parameters (email and password) are missing.

### 4. Create Appointment

**Endpoint:** /api/addAppointment

**Method:** POST

**Description:**

Allows patients to create appointments.

Prevents two patients from booking an appointment at the same date and time.

Appointments are RSVP until approved by the doctor.

**Request Parameters**

**patient\_id (integer):** ID of the patient creating the appointment.

**doctor\_id (integer):** ID of the doctor the patient wants to book an appointment with.

**appointment\_date (string, format: 'Y-m-d'):** Date of the appointment.

**appointment\_start\_time (string, format: 'H:i'):** Start time of the appointment.

**appointment\_end\_time will be added automatically of 30 min**

**Response**

Success (HTTP Status Code: 201 Created): Returns a success message upon successful appointment creation.

Error (HTTP Status Code: 400 Bad Request or 409 Conflict): Returns an error message if the appointment conflicts with an existing one.

## 5. Update Appointment

**Endpoint:** /api/ updateAppointment

**Method:** POST

### Description:

Allows patients to update the date and time of their appointments.

Ensures that the new date and time do not conflict with existing appointments.

### Request Parameters

**appointment\_date (string, format: 'Y-m-d'):** New date for the appointment.

**Appointment\_start\_time (string, format: 'H:i'):** New start time for the appointment.

**appointment\_end\_time will be added automatically of 30 min**

### Response

Success (HTTP Status Code: 200 OK): Returns a success message upon successful update.

Error (HTTP Status Code: 400 Bad Request or 409 Conflict): Returns an error message if the new date and time conflict with existing appointments.

## 6. View List of Their Appointments

**Endpoint:** /api/ listAppointmentsByUser

**Method:** POST

### Description:

Allows patients to view a list of their appointments.

Provides filtering by date.

Returns user-specific appointment information.

### Request Parameters

**date (string, format: 'Y-m-d'):** Optional date for filtering appointments.

### Response

Success (HTTP Status Code: 200 OK): Returns a list of appointments based on the filter criteria.

Error (HTTP Status Code: 401 Unauthorized or 404 Not Found): Returns an error message if the user or appointments are not found.

## Doctor Module

The Doctor Module is designed for doctors to manage their appointments and doctor-specific information. It includes four main APIs:

### 1. Create Doctor

**Endpoint:** /api/adminSignup

**Method:** POST

**Description:**

Allows the creation of doctor profiles.

Requires essential doctor information such as name, contact details, specialty, and more.

**Request Parameters**

**first\_name (string):** First name of the doctor.

**last\_name (string):** Last name of the doctor.

**email (string, format: email):** Email address of the doctor.

**phone (string):** Phone number of the doctor.

**speciality (string):** Specialty of the doctor.

**languages (string):** Languages spoken by the doctor.

**education (string):** Educational qualifications of the doctor.

**DOB (date, format: 'Y-m-d'):** Date of birth of the doctor.

**gender (string):** Gender of the doctor.

**device\_token (string):** Device token for authentication.

**image (file):** Profile image of the doctor.

**Response**

Success (HTTP Status Code: 201 Created): Returns a success message upon successful doctor profile creation.

Error (HTTP Status Code: 400 Bad Request): Returns an error message if the doctor profile creation fails.

### 2. Update Doctor Info

**Endpoint:** /api/adminUpdate

**Method:** POST

**Description:**

Allows doctors to update their profile information.

**Request Parameters**

Any parameters that need to be updated (e.g., first\_name, last\_name, etc.).

## **Response**

Success (HTTP Status Code: 200 OK): Returns a success message upon successful profile update.

Error (HTTP Status Code: 400 Bad Request): Returns an error message if the update fails.

## **3.Doctor Login**

**Endpoint:** /api/adminLogin

**Method:** POST

### **Description:**

Allows doctors to log in using their email and password.

### **Request Parameters:**

**email (string, format: email) [Required]:** Email address of the doctor.

**password (string) [Required]:** Doctor's password.

### **Response:**

Success (HTTP Status Code: 200 OK):

Returns a success message along with an access token upon successful login.

Error (HTTP Status Code: 401 Unauthorized):

Returns an error message if the provided credentials are invalid.

Error (HTTP Status Code: 400 Bad Request):

Returns an error message if the required parameters (email and password) are missing.

## **4. View Appointments**

**Endpoint:** /api/listAppointmentsByDoctor

**Method:** POST

### **Description:**

Allows doctors to view a list of their appointments.

Provides filtering by date.

Returns user-specific appointment information.

### **Request Parameters**

**date (string, format: 'Y-m-d'):** Optional date for filtering appointments.

### **Response**

Success (HTTP Status Code: 200 OK): Returns a list of appointments based on the filter criteria.

Error (HTTP Status Code: 401 Unauthorized or 404 Not Found): Returns an error message if the doctor or appointments are not found.

## 5. Update Status of Appointment (Doctor)

**Endpoint:** /api/ changeStatus /{id}

**Method:** POST

### Description:

Allows doctors to update the status of appointments they manage.

Status options: 'approved', 'cancelled', 'postponed'.

### Request Parameters

**status (string):** New status for the appointment.

**appointment\_date (date, format: 'Y-m-d') [Required if status is 'postponed']:** New date for the appointment.

**appointment\_start\_time (time, format: 'H:i') [Required if status is 'postponed']:** New start time for the appointment

**appointment\_end\_time will be added automatically of 30 min**

### Response

Success (HTTP Status Code: 200 OK): Returns a success message upon successful status update.

Error (HTTP Status Code: 401 Unauthorized or 404 Not Found): Returns an error message if the appointment or doctor is not valid.