Case Study On

MediBook – Doctor Appointment Booking System

INTRODUCTION

- MediBook is a web-based doctor appointment booking system that automates appointment scheduling.
- It allows patients to book, view, and cancel appointments online.
- The system reduces long waiting times and manual hospital processes.
- It minimizes administrative workload and improves appointment management.
- MediBook provides an efficient and user-friendly solution for patients and doctors.

ABSTRAT

The **MediBook – Doctor Appointment Booking System** is developed to provide a seamless and user-friendly platform for managing doctor appointments online. Patients can register, log in, view available doctors, book appointments, view their booked appointments, and cancel appointments when required. Doctors can register, log in, and view the list of patients who have booked appointments with them, including appointment date and time. The application is developed using Spring Boot, Spring Data JPA (Hibernate), MySQL, Postman, and HTML/CSS/JavaScript for frontend interaction.This system ensures secure, role-based access and real-time appointment handling using REST APIs.

CLIENT REQUIREMENT

- A web-based Doctor Appointment Booking System named MediBook.
- A system that allows patients to register, log in, and manage their appointments online.
- A doctor module where doctors can register, log in, and view appointments booked by patients.
- An appointment management module that enables patients to book, view, and cancel appointments.
- A role-based dashboard system for patients and doctors with separate functionalities.
- A platform that displays available doctors along with their specialization details.
- A scheduling mechanism to avoid appointment conflicts and overlapping bookings.

TECHNICAL FEATURES

- RESTful APIs using Spring Boot
- Layered Architecture (Controller, Service, Repository)
- Session-based login handling (Session Storage)
- JSON-based data exchange
- CRUD operations
- Exception handling
- Role-based dashboard navigation

TECHNOLOGIES AND TOOLS USED

Backend
Java 17
Spring Boot 3
Spring Data JPA (Hibernate)
Maven
REST APIs
Apache Tomcat (Embedded)

Database
MySQL 8.0

Frontend
HTML
CSS
JavaScript

Tools
Spring Tool Suite (STS) 4
Postman
MySQL Workbench
Google Chrome

SYSTEM REQUIREMENTS

Software Requirements

Operating System: Windows 10+

Java JDK 17

MySQL Server 8.0

Spring Tool Suite (STS)

Web Browser (Chrome)

Hardware Requirements

Processor: Intel i3 or above

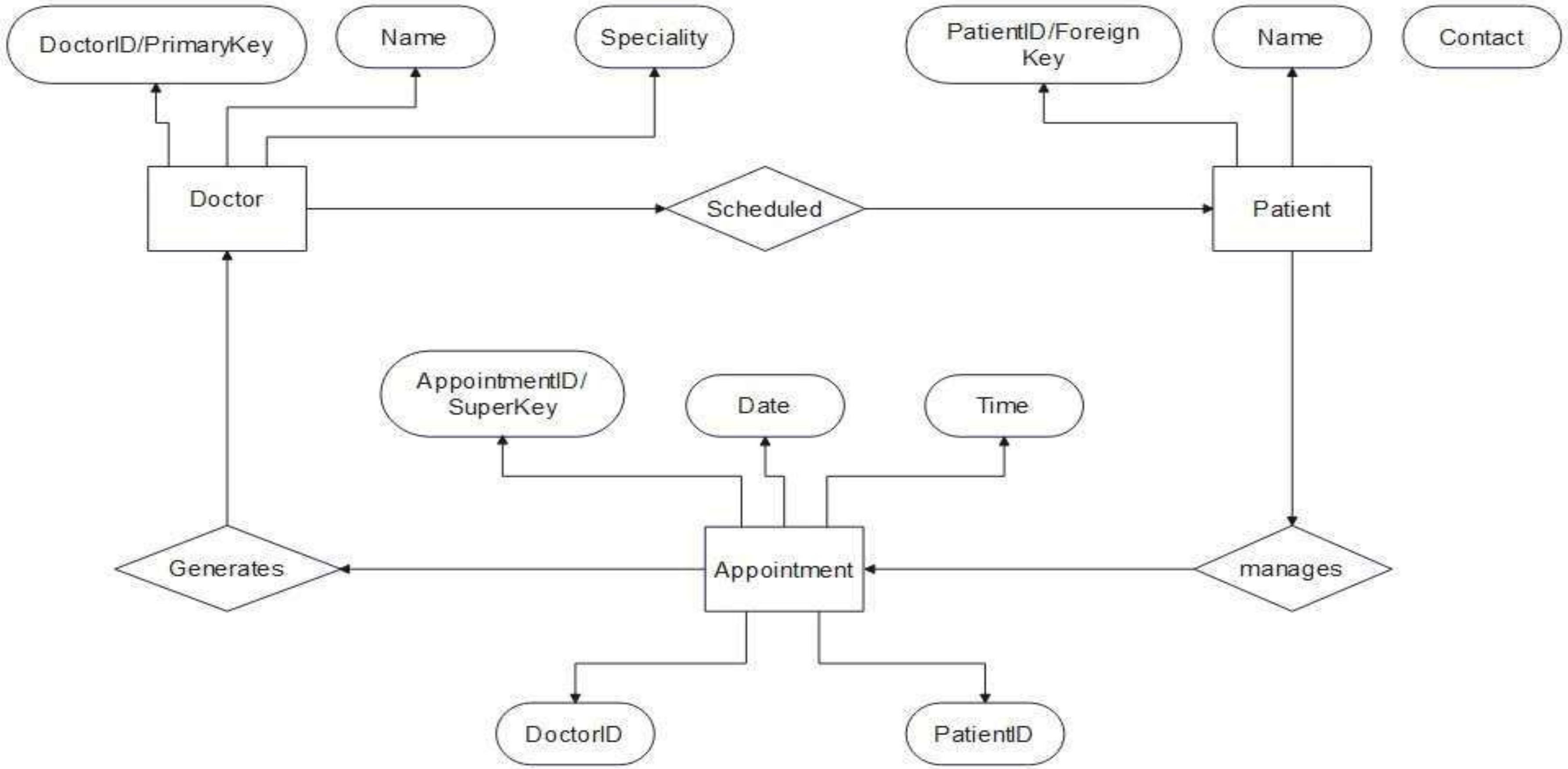
RAM: Minimum 4 GB

Hard Disk: 10 GB free space

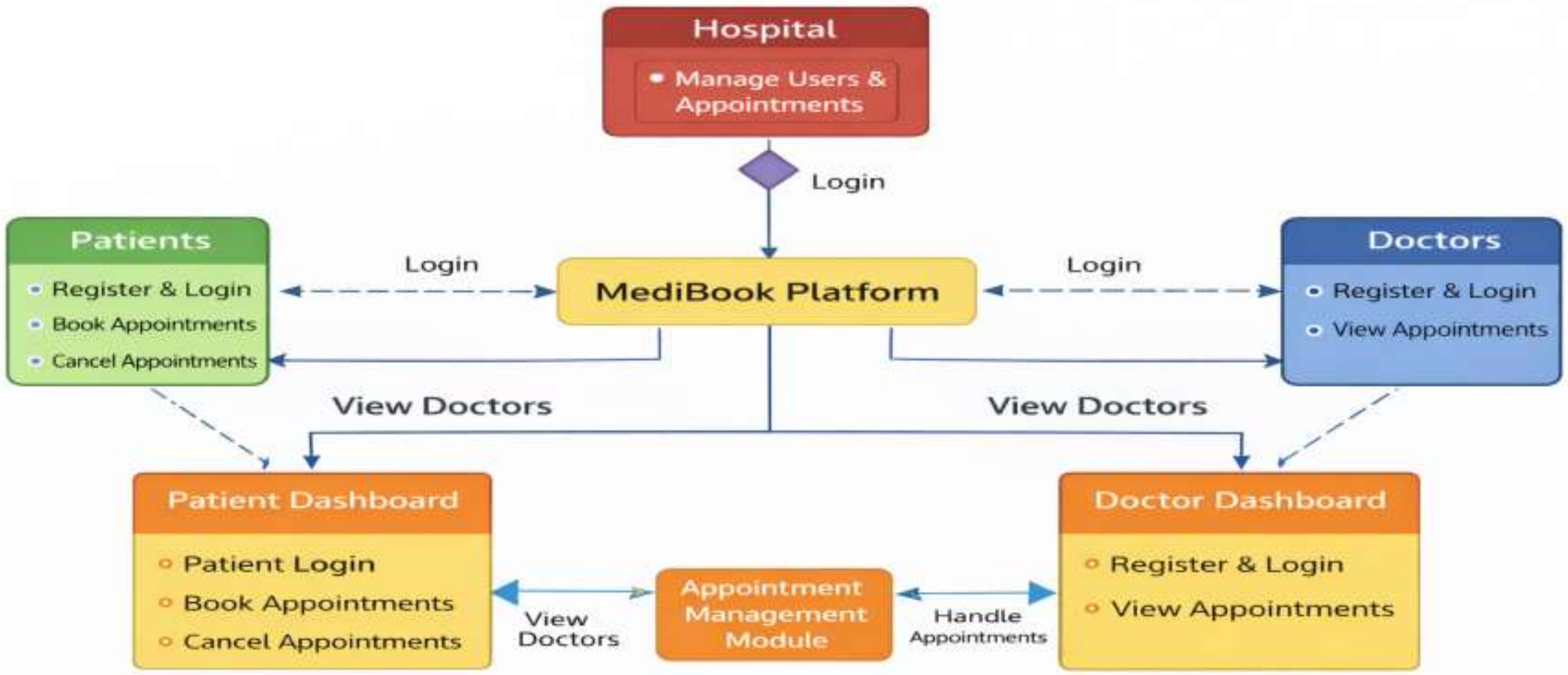
PROJECT MODULE

- Patient Module
- Doctor Module
- Appointment Module
- Home / Navigation Module

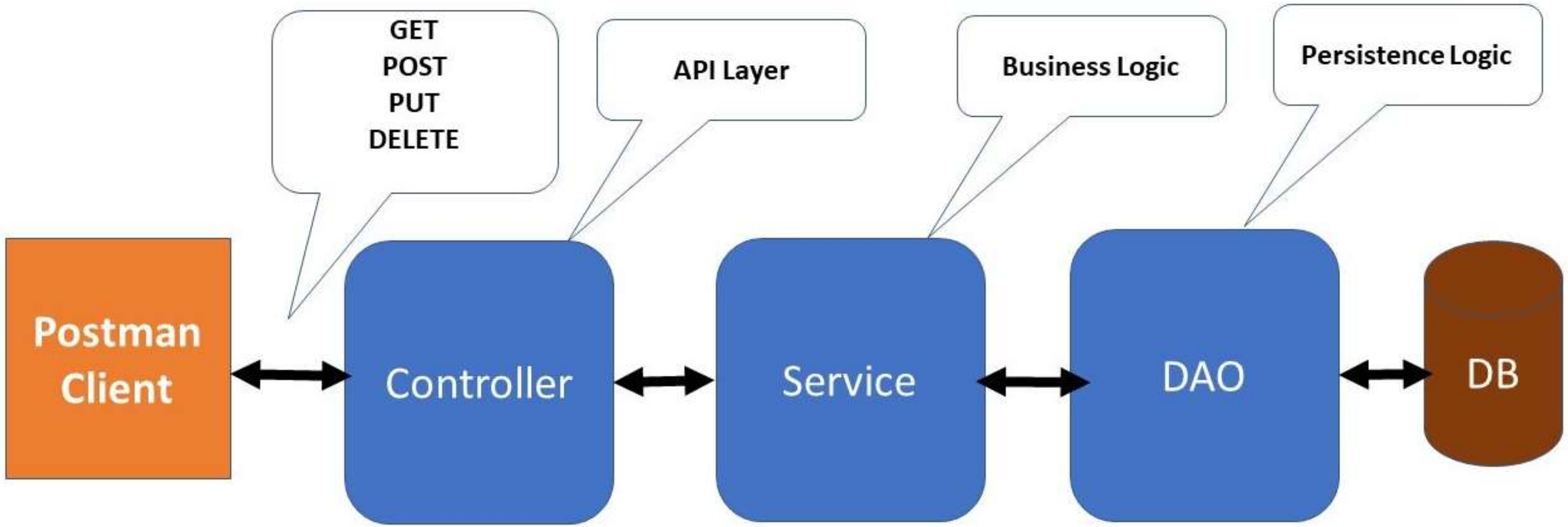
ER DIAGRAM


DoctorID/PrimaryKey
Name
Speciality
PatientID/Foreign Key
Name
Contact
Doctor
Scheduled
Patient
AppointmentID/ SuperKey
Date
Time
Generates
Appointment
manages
DoctorID
PatientID

Client – Server Architecture



Spring Boot APP



HOME MODULE

Acts as the entry point of the MediBook system.

Provides navigation options for:

- Patient
- Doctor
- Hospital

Redirects users based on login status.

Improves usability by guiding users to the correct dashboard.

Ensures easy access to different system roles.

HOSPITAL MODULE

- Allows hospital staff to view registered doctors.
- Displays doctor details such as name and specialization.
- Helps in managing overall appointment flow.
- Acts as a supervisory interface for hospital operations.
- Improves coordination between doctors and patients

DOCTOR MODULE

- Allows doctors to register and log in to the system.
- Displays a personalized Doctor Dashboard.
- Enables doctors to view:
- List of patients who booked appointments
- Appointment date and time
- Helps doctors manage their daily schedules efficiently.
- Reduces manual tracking of patient appointments.

PATIENT MODULE

- Allows patients to register and log in securely.
- Displays available doctors with specialization details.
- Enables patients to:
- Book appointments
- View booked appointments
- Cancel appointments if needed
- Prevents overlapping appointments.
- Provides a user-friendly interface for appointment management.

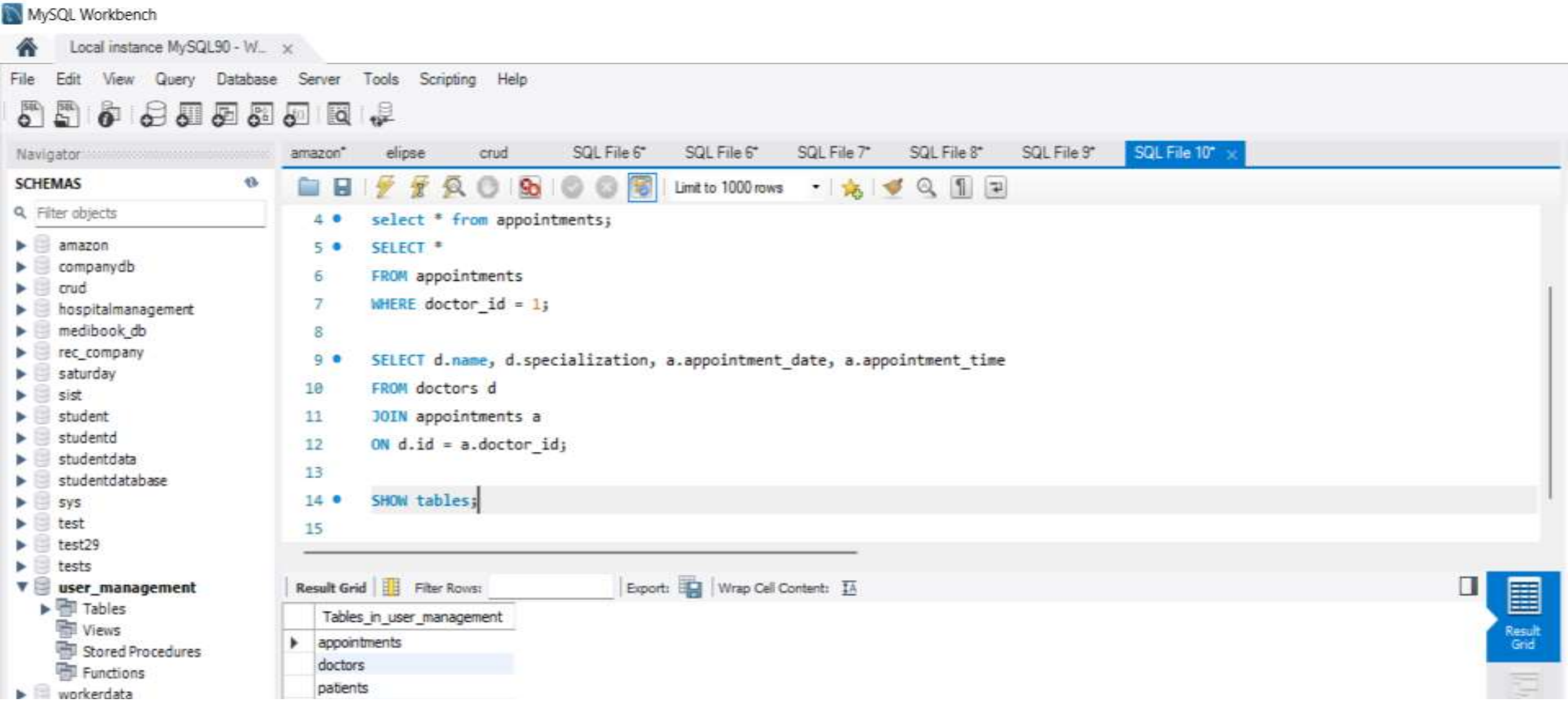
Http Request Methods

HTTP request methods are used to perform different operations such as fetching data, creating records, and deleting records in the MediBook system.

GET	http://localhost:8080/api/doctors	View doctors
POST	http://localhost:8080/api/appointments/book	Book appointment
GET	http://localhost:8080/api/appointments/patient	View patient appointments
GET	http://localhost:8080/api/appointments/doctor/id	View doctor appointments
DELETE	http://localhost:8080/api/appointments/id	Cancel appointment

DATA DICTIONARY

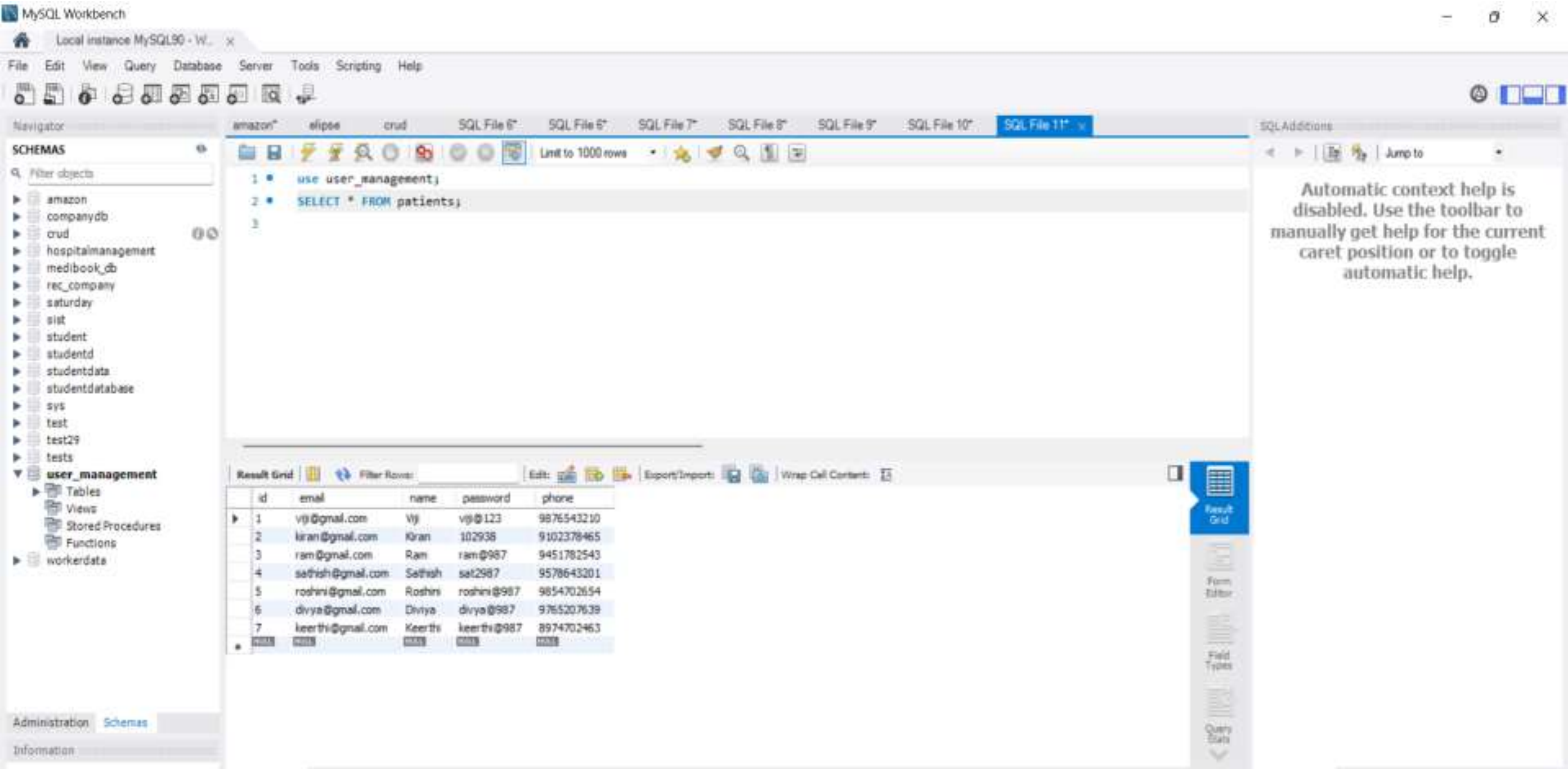
TABLES OF DATABASE



```
select * from appointments;
SELECT *
FROM appointments
WHERE doctor_id = 1;

SELECT d.name, d.specialization, a.appointment_date, a.appointment_time
FROM doctors d
JOIN appointments a
ON d.id = a.doctor_id;

SHOW tables;

```

Tables_in_user_management
appointments
doctors
patients

PATIENT DATABASE

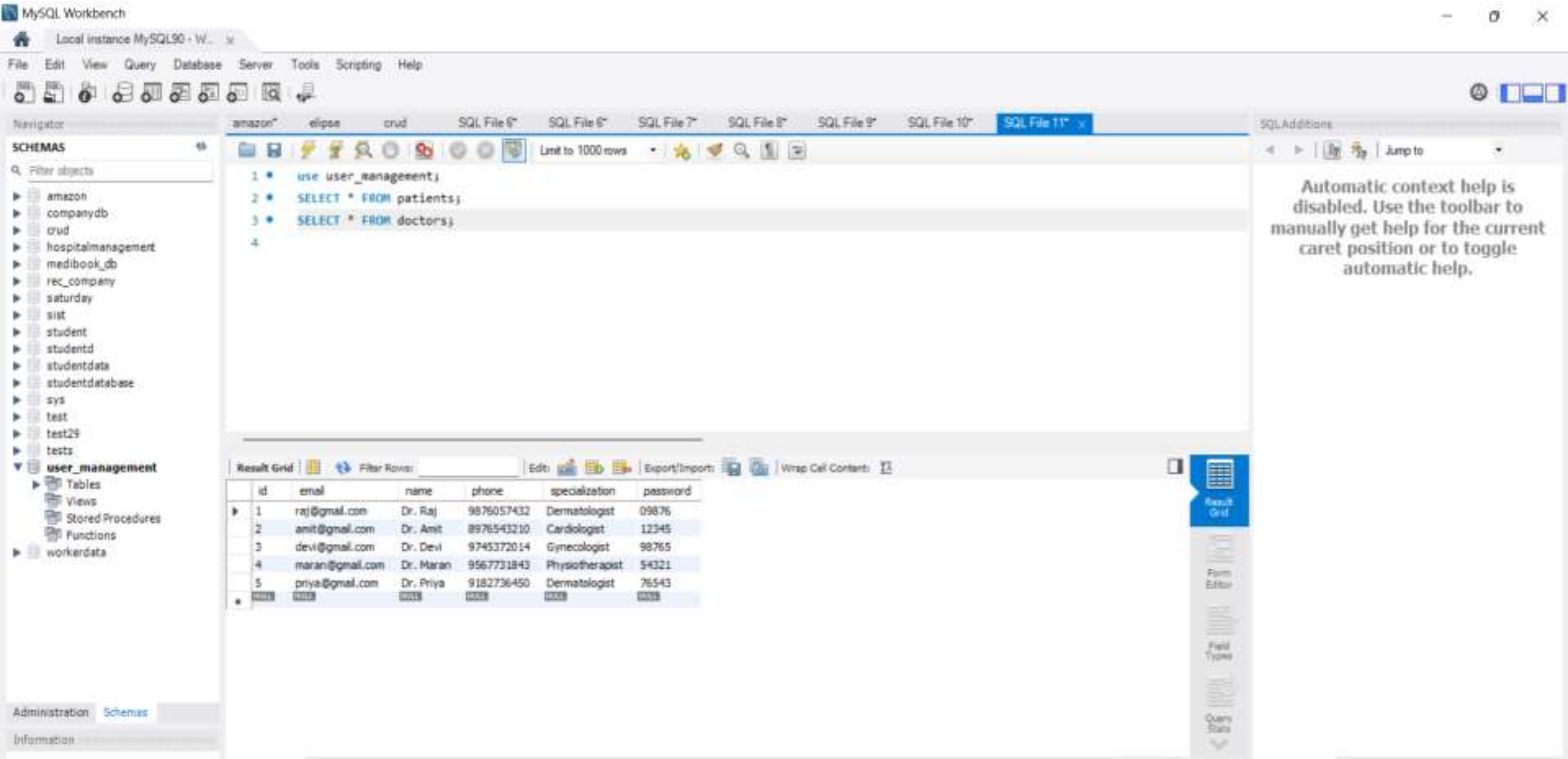


```
use user_management;
SELECT * FROM patients;
```

id	email	name	password	phone
1	viji@gmail.com	Viji	viji@123	9876543210
2	kiran@gmail.com	Kiran	102938	9102378465
3	ram@gmail.com	Ram	ram@987	9451782543
4	sathish@gmail.com	Sathish	sat2987	9578643201
5	roshini@gmail.com	Roshini	roshini@987	9854702654
6	divya@gmail.com	Divya	divya@987	9765207639
7	keerthi@gmail.com	Keerthi	keerthi@987	8974702463
NULL	NULL	NULL	NULL	NULL

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

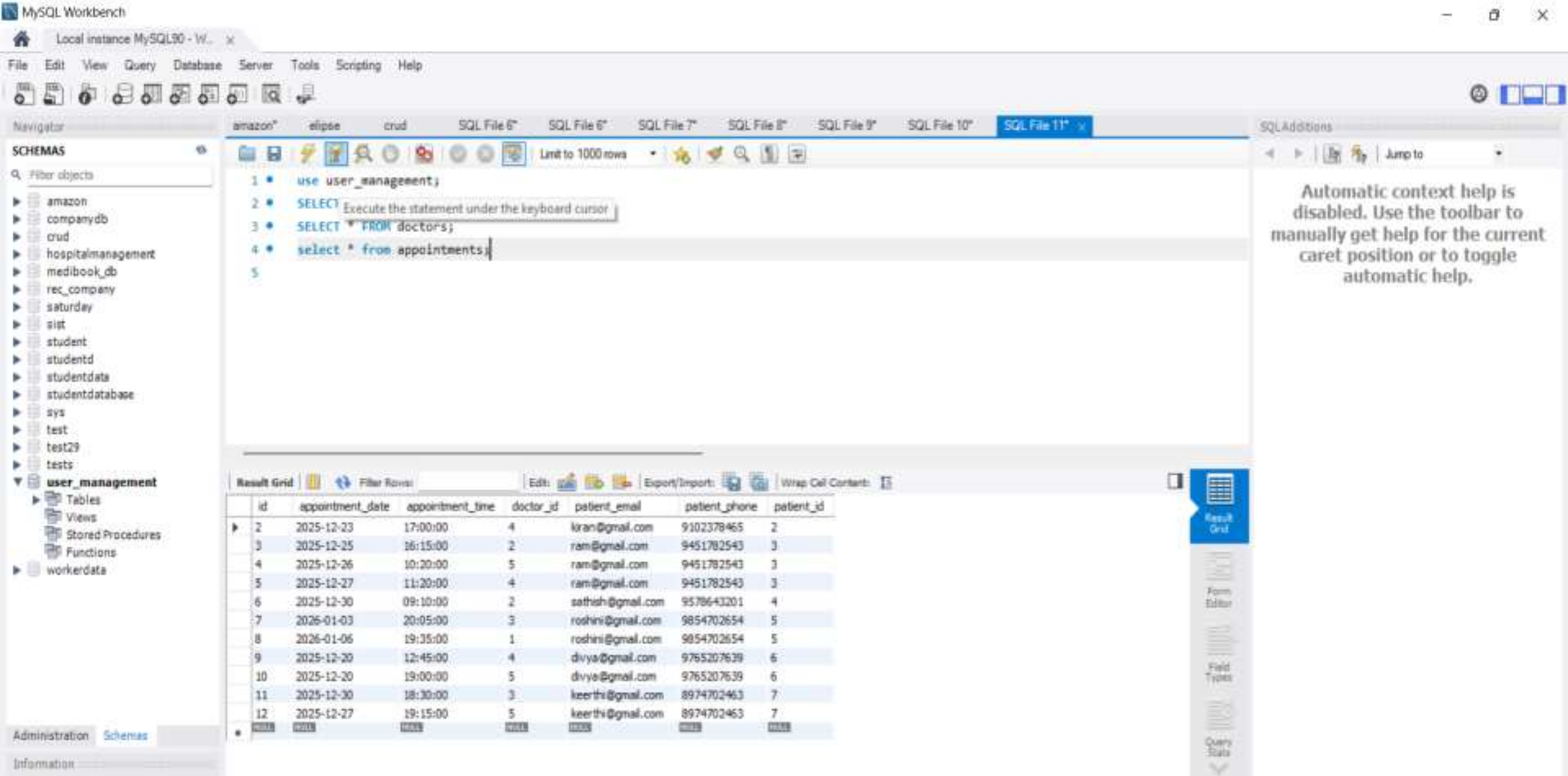
DOCTOR DATABASE



```
use user_management;
SELECT * FROM patients;
SELECT * FROM doctors;
```

id	email	name	phone	specialization	password
1	raj@gmail.com	Dr. Raj	9876057432	Dermatologist	09876
2	amit@gmail.com	Dr. Amit	8976543210	Cardiologist	12345
3	devi@gmail.com	Dr. Devi	9745372014	Gynecologist	98765
4	maran@gmail.com	Dr. Maran	9567731843	Physiotherapist	54321
5	priya@gmail.com	Dr. Priya	9182736450	Dermatologist	76543
NULL	NULL	NULL	NULL	NULL	NULL

APPOINTMENT DATABASE

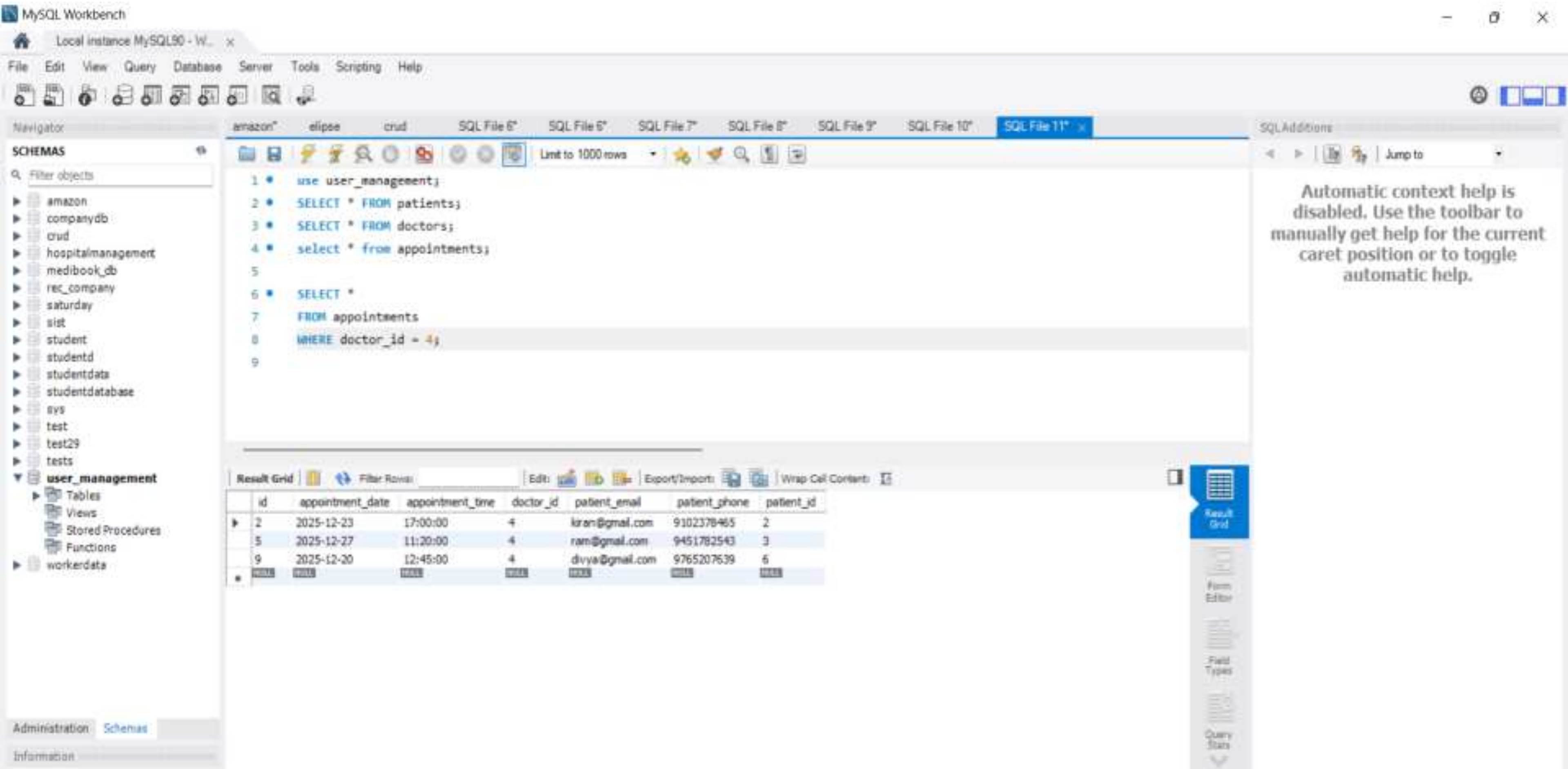


```sql
use user_management;
SELECT
SELECT * FROM doctors;
select * from appointments;
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

id	appointment_date	appointment_time	doctor_id	patient_email	patient_phone	patient_id
2	2025-12-23	17:00:00	4	kiran@gmail.com	9102378465	2
3	2025-12-25	16:15:00	2	ram@gmail.com	9451782543	3
4	2025-12-26	10:20:00	5	ram@gmail.com	9451782543	3
5	2025-12-27	11:20:00	4	ram@gmail.com	9451782543	3
6	2025-12-30	09:10:00	2	sathish@gmail.com	9578643201	4
7	2026-01-03	20:05:00	3	roshini@gmail.com	9854702654	5
8	2026-01-06	19:35:00	1	roshini@gmail.com	9854702654	5
9	2025-12-20	12:45:00	4	divya@gmail.com	9765207639	6
10	2025-12-20	19:00:00	5	divya@gmail.com	9765207639	6
11	2025-12-30	18:30:00	3	keerthi@gmail.com	8974702463	7
12	2025-12-27	19:15:00	5	keerthi@gmail.com	8974702463	7
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Appointments for a Specific Doctor

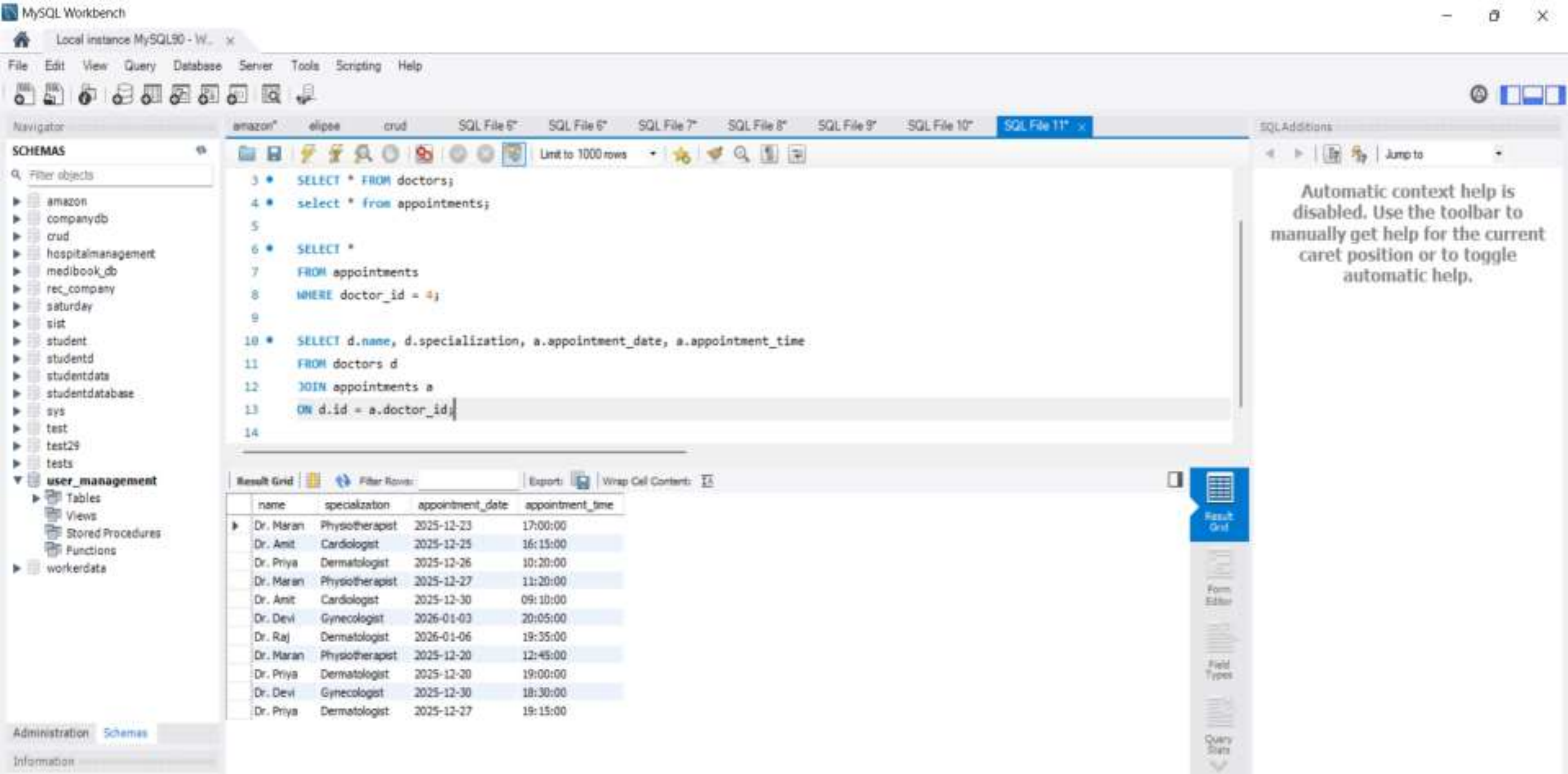


```
use user_management;
SELECT * FROM patients;
SELECT * FROM doctors;
select * from appointments;

SELECT *
FROM appointments
WHERE doctor_id = 4;
```

id	appointment_date	appointment_time	doctor_id	patient_email	patient_phone	patient_id
2	2025-12-23	17:00:00	4	kiran@gmail.com	9102378465	2
5	2025-12-27	11:20:00	4	ram@gmail.com	9451782543	3
9	2025-12-20	12:45:00	4	divya@gmail.com	9765207639	6
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

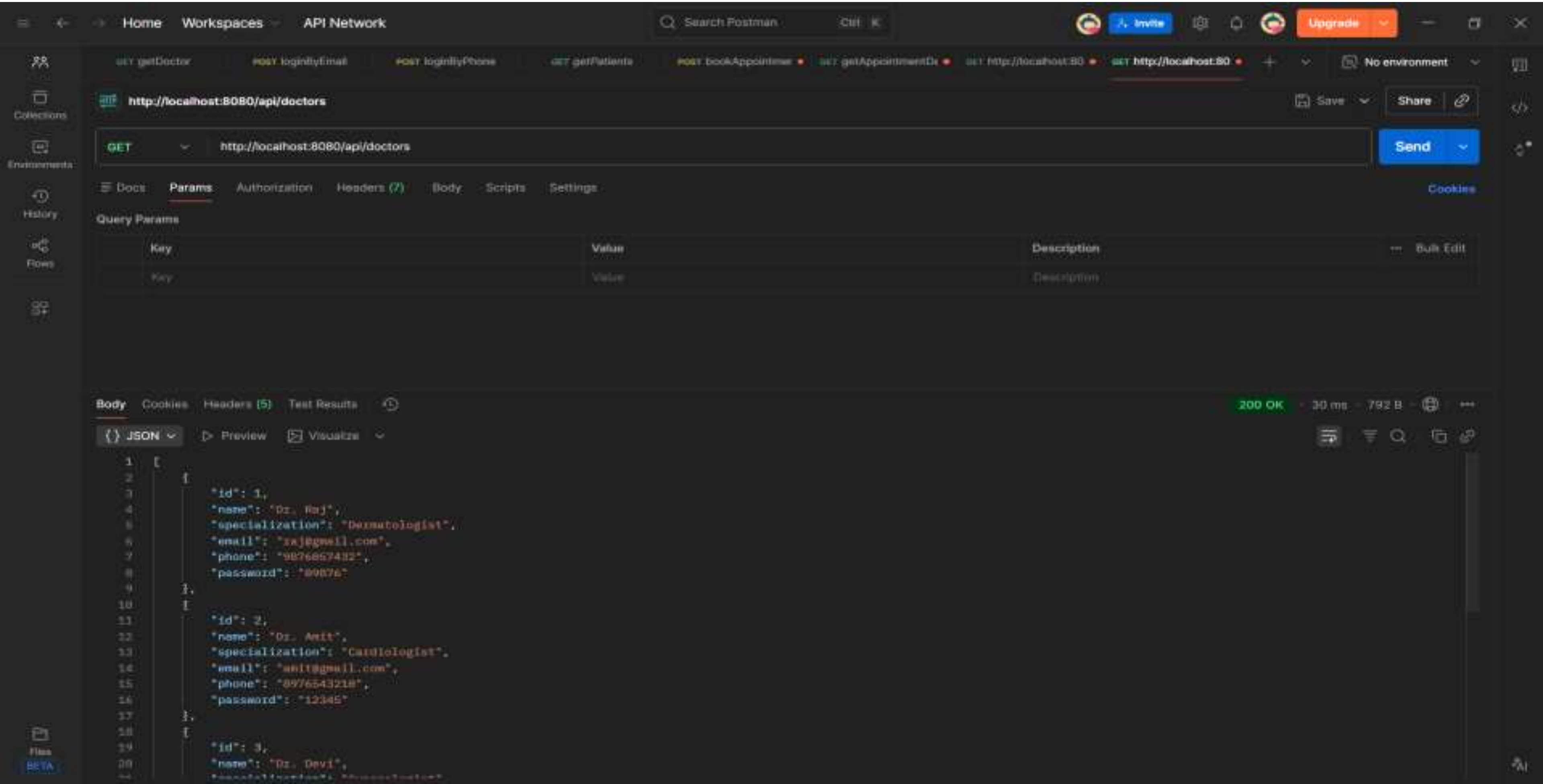
HOSPITAL DATABASE



```sql
SELECT * FROM doctors;
select * from appointments;

SELECT *
FROM appointments
WHERE doctor_id = 4;

SELECT d.name, d.specialization, a.appointment_date, a.appointment_time
FROM doctors d
JOIN appointments a
ON d.id = a.doctor_id;
```

name	specialization	appointment_date	appointment_time
Dr. Maran	Physiotherapist	2025-12-23	17:00:00
Dr. Amit	Cardiologist	2025-12-25	16:15:00
Dr. Priya	Dermatologist	2025-12-26	10:20:00
Dr. Maran	Physiotherapist	2025-12-27	11:20:00
Dr. Amit	Cardiologist	2025-12-30	09:10:00
Dr. Devi	Gynecologist	2026-01-03	20:05:00
Dr. Raj	Dermatologist	2026-01-06	19:35:00
Dr. Maran	Physiotherapist	2025-12-20	12:45:00
Dr. Priya	Dermatologist	2025-12-20	19:00:00
Dr. Devi	Gynecologist	2025-12-30	18:30:00
Dr. Priya	Dermatologist	2025-12-27	19:15:00

GET METHOD FOR VIEWING DOCTORS



http://localhost:8080/api/doctors

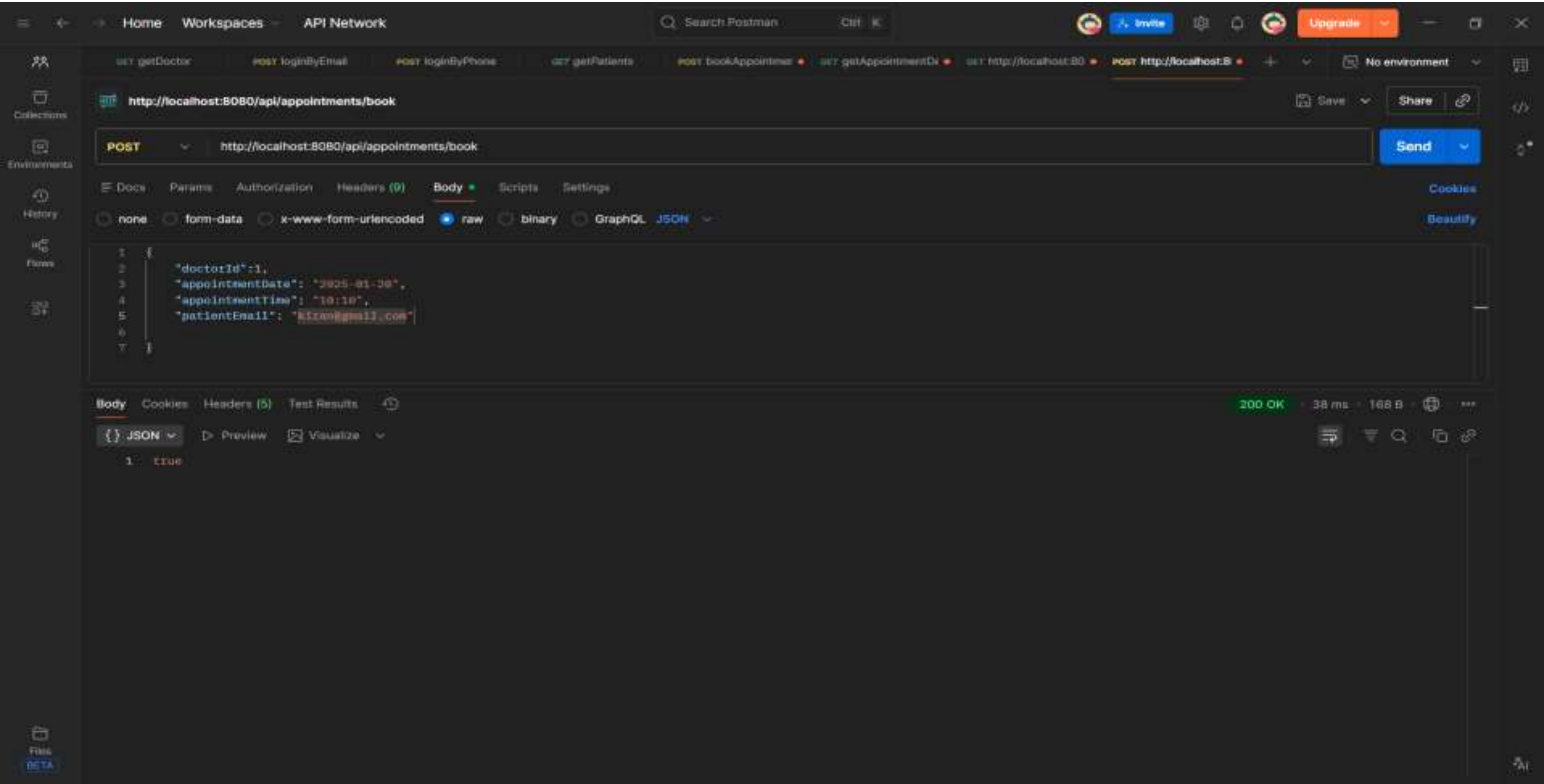
GET http://localhost:8080/api/doctors

Query Params

200 OK · 30 ms · 792 B

```
[
    {
        "id": 1,
        "name": "Dr. Raj",
        "specialization": "Dermatologist",
        "email": "raj@gmail.com",
        "phone": "9876857432",
        "password": "09876"
    },
    {
        "id": 2,
        "name": "Dr. Amit",
        "specialization": "Cardiologist",
        "email": "amit@gmail.com",
        "phone": "8976543218",
        "password": "12345"
    },
    {
        "id": 3,
        "name": "Dr. Devi",
```

POST METHOD FOR BOOKING APPOINTMENT



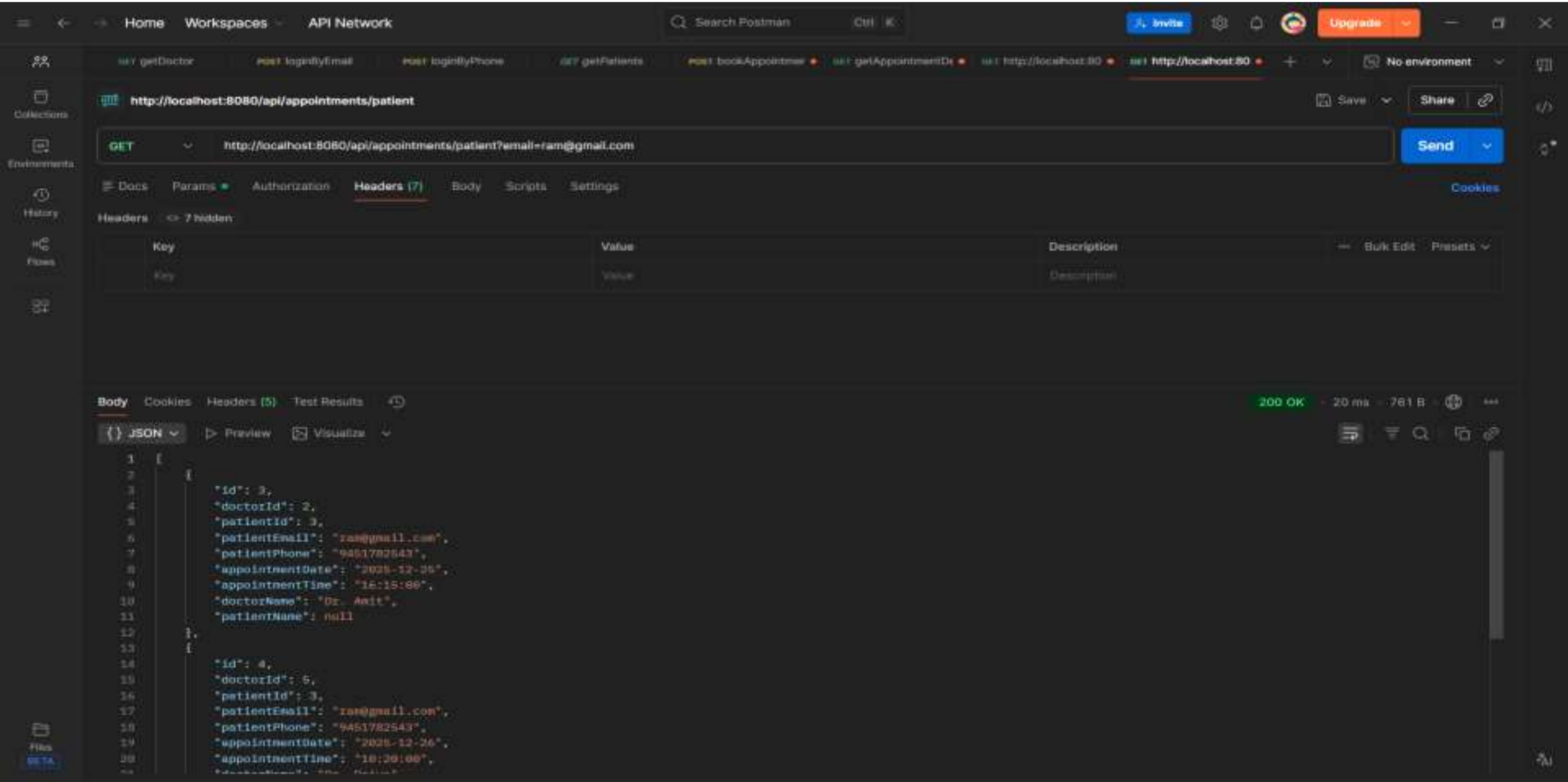
POST http://localhost:8080/api/appointments/book

```
{
    "doctorId":1,
    "appointmentDate": "2025-01-20",
    "appointmentTime": "10:10",
    "patientEmail": "kiran@gmail.com"

}
```

200 OK · 38 ms · 168 B

```
true
```

GET METHOD FOR VIEWING PATIENT APPOINTMENTS



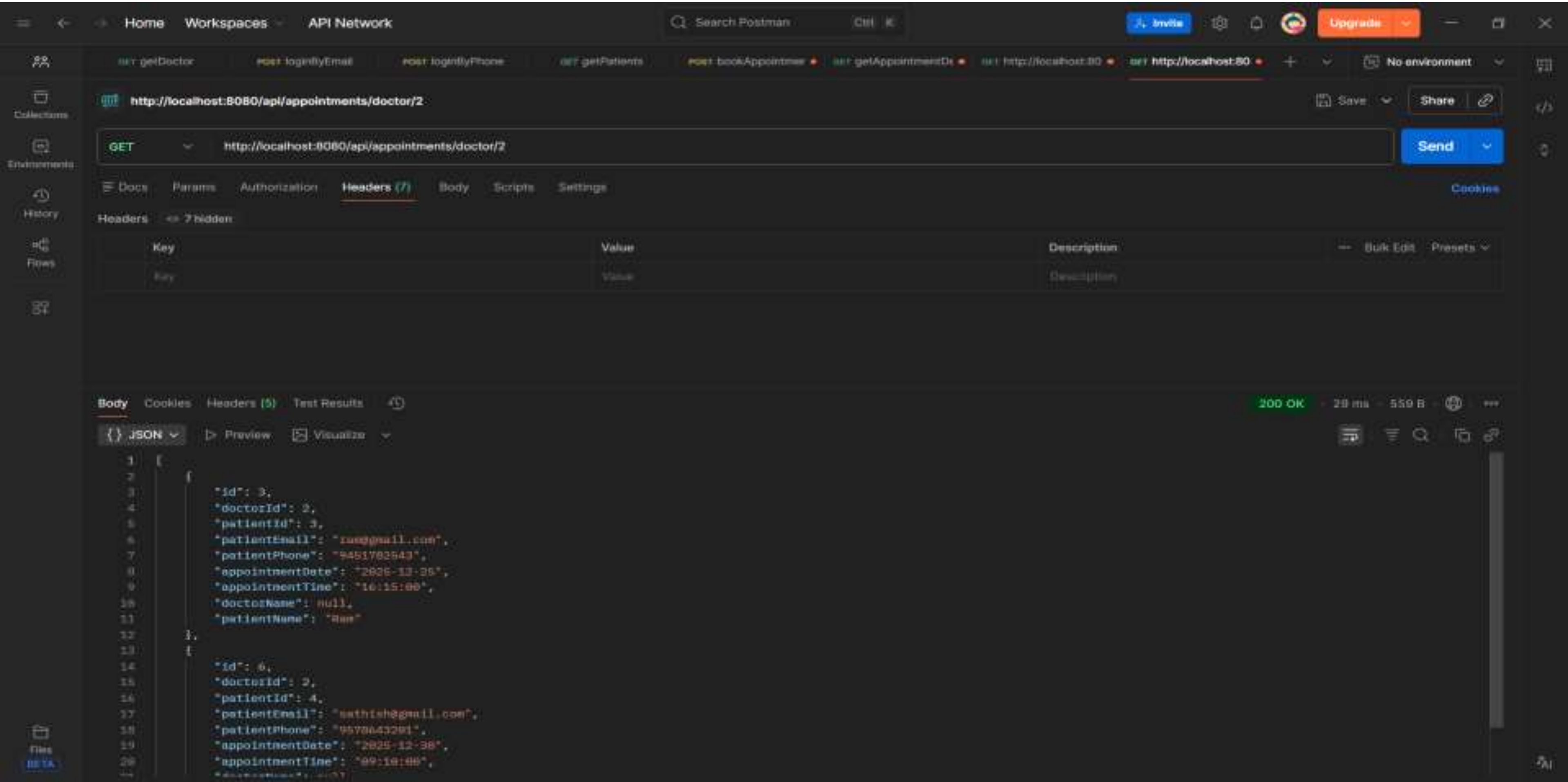
http://localhost:8080/api/appointments/patient

GET http://localhost:8080/api/appointments/patient?email=ram@gmail.com

200 OK · 20 ms · 761 B

```
[
    {
        "id": 3,
        "doctorId": 2,
        "patientId": 3,
        "patientEmail": "ram@gmail.com",
        "patientPhone": "9451782543",
        "appointmentDate": "2025-12-25",
        "appointmentTime": "16:15:00",
        "doctorName": "Dr. Amit",
        "patientName": null
    },
    {
        "id": 4,
        "doctorId": 5,
        "patientId": 3,
        "patientEmail": "ram@gmail.com",
        "patientPhone": "9451782543",
        "appointmentDate": "2025-12-26",
        "appointmentTime": "10:20:00",
```

GET METHOD FOR VIEWING DOCTOR APPOINTMENTS



GET http://localhost:8080/api/appointments/doctor/2

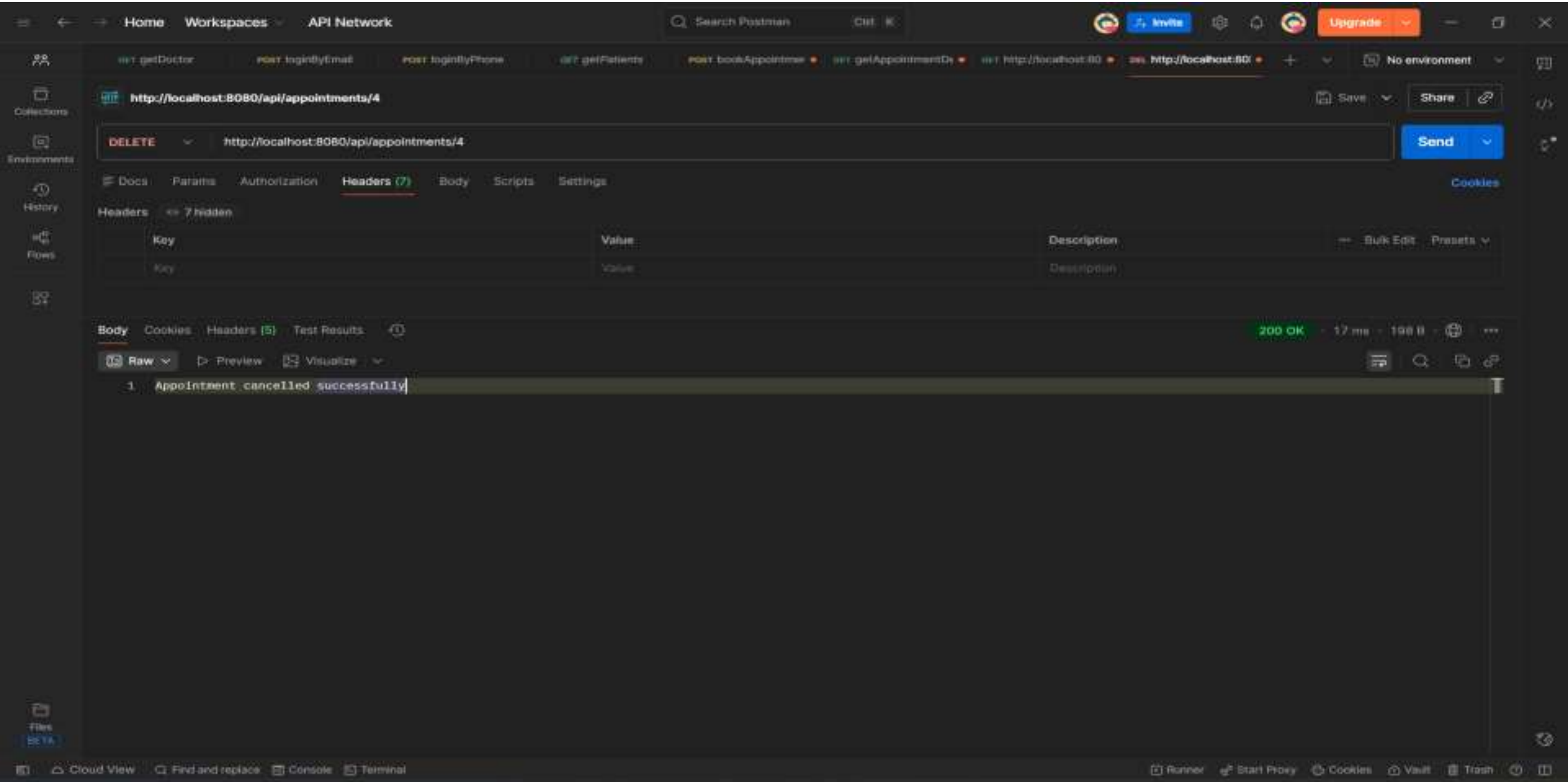
Headers · 7 hidden

Key	Value	Description
Key	Value	Description

200 OK · 29 ms · 559 B

```
[
    {
        "id": 3,
        "doctorId": 2,
        "patientId": 3,
        "patientEmail": "ram@gmail.com",
        "patientPhone": "9451782543",
        "appointmentDate": "2026-12-25",
        "appointmentTime": "16:15:00",
        "doctorName": null,
        "patientName": "Ram"
    },
    {
        "id": 6,
        "doctorId": 2,
        "patientId": 4,
        "patientEmail": "sathish@gmail.com",
        "patientPhone": "9578643201",
        "appointmentDate": "2025-12-30",
        "appointmentTime": "09:10:00",
```

DELETE METHOD FOR CANCELLING APPOINTMENT

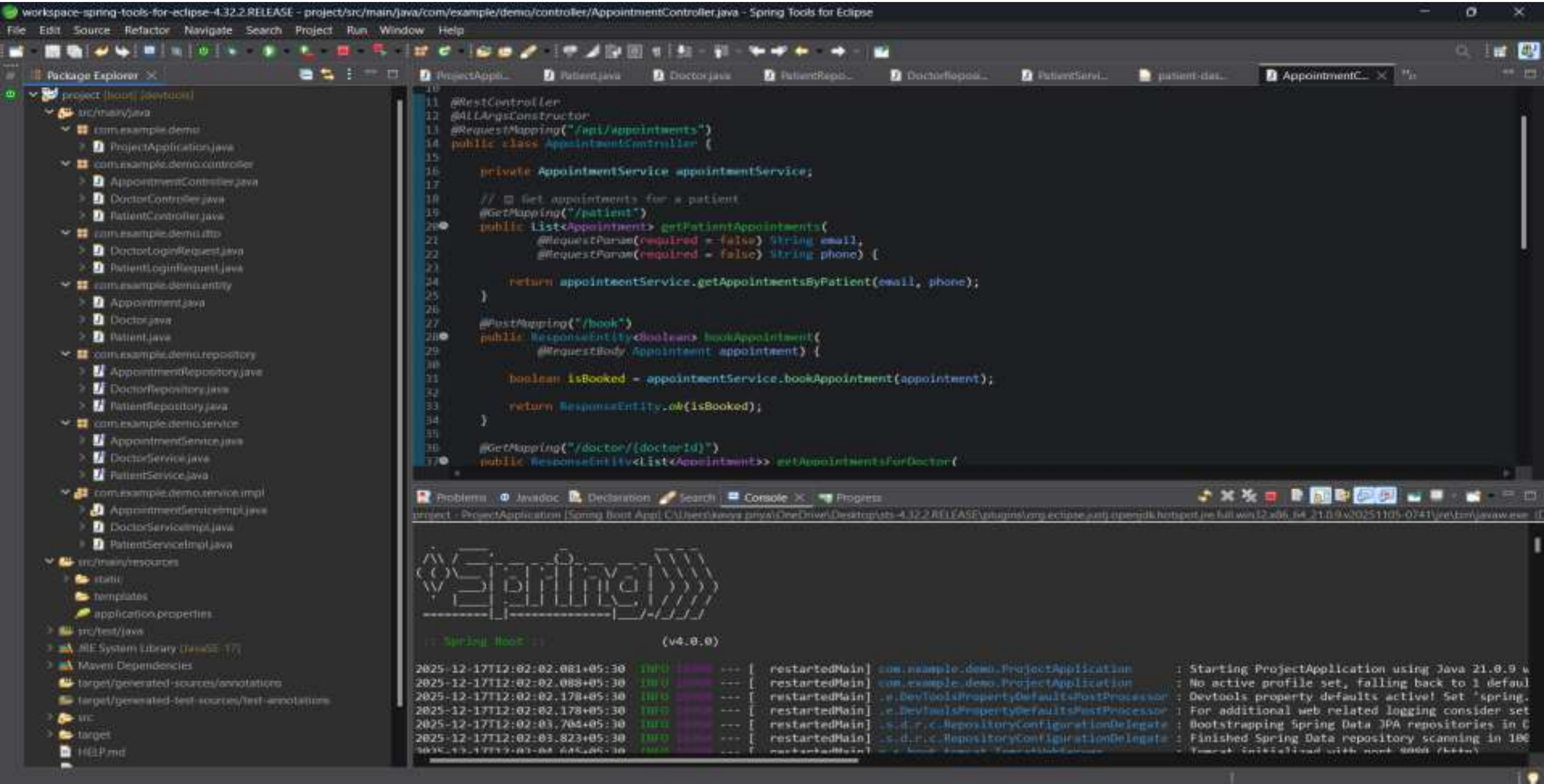


http://localhost:8080/api/appointments/4

DELETE http://localhost:8080/api/appointments/4

```
Appointment cancelled successfully
```

WORKING OF SPRING TOOL



```java
@RestController
@AllArgsConstructor
@RequestMapping("/api/appointments")
public class AppointmentController {

    private AppointmentService appointmentService;

    // ▢ Get appointments for a patient
    @GetMapping("/patient")
    public List<Appointment> getPatientAppointments(
            @RequestParam(required = false) String email,
            @RequestParam(required = false) String phone) {

        return appointmentService.getAppointmentsByPatient(email, phone);
    }

    @PostMapping("/book")
    public ResponseEntity<Boolean> bookAppointment(
            @RequestBody Appointment appointment) {

        boolean isBooked = appointmentService.bookAppointment(appointment);

        return ResponseEntity.ok(isBooked);
    }

    @GetMapping("/doctor/{doctorId}")
    public ResponseEntity<List<Appointment>> getAppointmentsForDoctor(
```

```
project - ProjectApplication [Spring Boot App] C:\Users\kavya priya\OneDrive\Desktop\sts-4.32.2.RELEASE\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.9.v20251105-0741\jre\bin\javaw.exe

 :: Spring Boot ::                (v4.0.0)

2025-12-17T12:02:02.081+05:30  INFO [illegible] --- [  restartedMain] com.example.demo.ProjectApplication       : Starting ProjectApplication using Java 21.0.9 w
2025-12-17T12:02:02.088+05:30  INFO [illegible] --- [  restartedMain] com.example.demo.ProjectApplication       : No active profile set, falling back to 1 defaul
2025-12-17T12:02:02.178+05:30  INFO [illegible] --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.
2025-12-17T12:02:02.178+05:30  INFO [illegible] --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider set
2025-12-17T12:02:03.704+05:30  INFO [illegible] --- [  restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in D
2025-12-17T12:02:03.823+05:30  INFO [illegible] --- [  restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 10
```