FRACTO

-Parth Tinna

Table of Contents

1. Introduction and Project Goals
2. System Architecture
3. Workflow and Design Diagrams
4. Application Modules & API Overview
5. Database Schema and Data Handling
6. Testing Strategy
7. Conclusion

---

1. Introduction and Project Goals

Problem Context

In many hospitals and clinics, patients still depend on outdated methods to book appointments, such as phone calls or in-person scheduling. These approaches are time-consuming, prone to errors, and offer little to no transparency about doctor availability. Administrators, meanwhile, must manually track appointments, often resulting in double-bookings or missed slots.

Project Goals

The Fracto system was developed with the following intentions:

- Deliver a **web-based platform** that simplifies appointment booking for patients.

- Allow **administrators to efficiently manage** doctors, patients, and schedules.
- Guarantee **security** using JWT authentication and role-based permissions.
- Provide **real-time updates** through SignalR to improve communication.
- Enhance user engagement with a **doctor rating and feedback system**.
- Support **profile image uploads** to maintain professional and user identities.

---

2. System Architecture

Chosen Technologies

- **Frontend**: Angular 16 with Angular Material for responsive, interactive UI.
- **Backend**: ASP.NET Core Web API (.NET 9) with EF Core ORM for smooth data operations.
- **Database**: SQL Server for production; SQLite during development for simplicity.
- **Security**: JWT-based authentication tokens with embedded claims.
- **File Storage**: Local file system (wwwroot/uploads).

Interaction Model

1. **Patient/Admin** accesses Angular frontend.
2. Requests are routed to **ASP.NET Core API endpoints**.
3. The backend uses **EF Core** to query and update the database.
4. Uploaded files are stored and retrieved from the server directory.

---

3. Workflow and Design Diagrams

Application Workflow

- **Patient Workflow**: Register → Login → Select City → Search Doctor → Choose Slot → Book Appointment → Receive Confirmation .

- **Admin Workflow**: Login → Manage Users → Add/Edit Doctors → Approve/Reject Appointments

Design Flow

- **Frontend** handles navigation, form validation, and API requests.
- **Backend** provides endpoints for each feature (Auth, Users, Doctors, Appointments, Ratings).
- **Database** maintains relationships between users, doctors, and appointments.
- **SignalR Hub** ensures real-time synchronization.

---

4. Application Modules & API Overview

Angular Modules

- **Authentication**: Components for Login and Register with Guards for route protection.
- **User Section**: DoctorSearchComponent, AppointmentBookComponent, UserAppointmentsComponent.
- **Admin Section**: Components for User Management, Doctor Management, Specialization Management, and Appointment Oversight.
- **Shared Utilities**: Navbar, Snackbar notifications, and FileUpload component.

Web API Controllers

- **AuthController** → Register & Login.
- **UsersController** → Manage patients (CRUD, Admin only).
- **DoctorsController** → CRUD, search by filters, and timeslot availability.
- **AppointmentsController** → Book, reschedule, cancel, approve, reject.

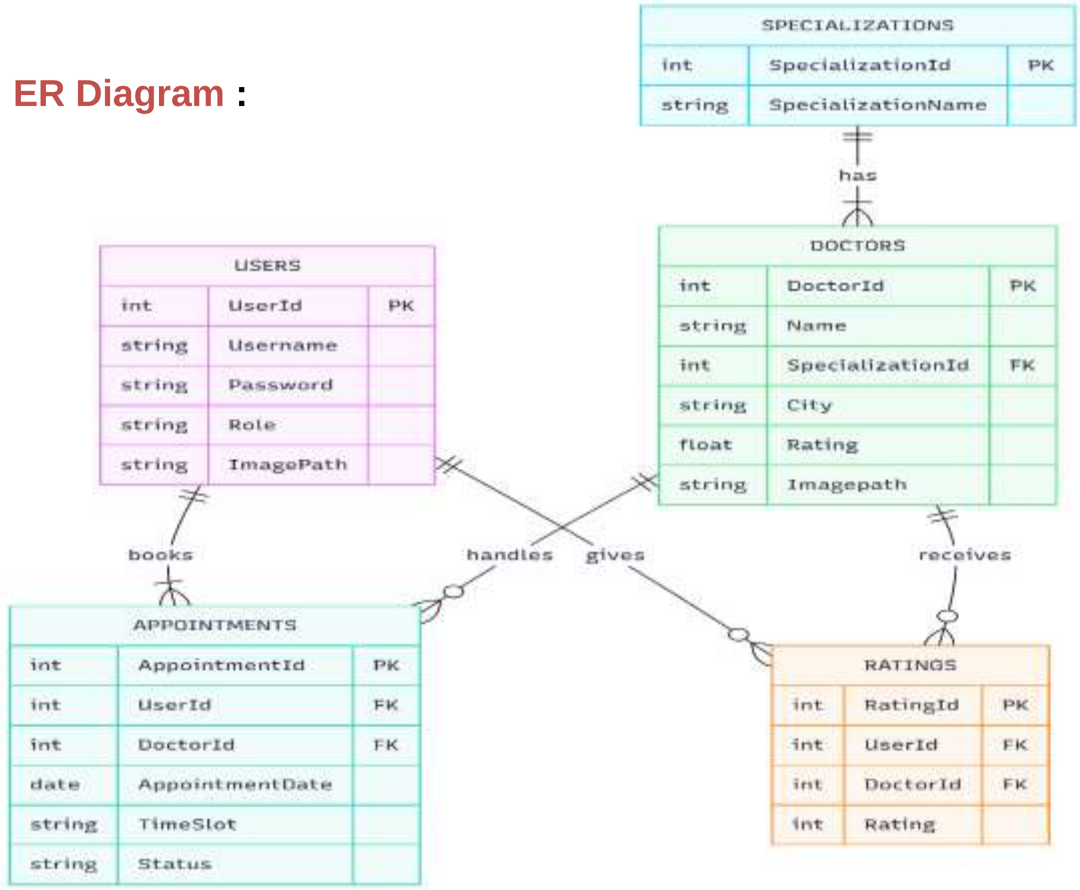
---

5. Database Schema and Data Handling

Core Tables

- **Users**: UserId, Username, Password, Role, ProfileImagePath.
- **Doctors**: DoctorId, Name, SpecializationId, City, Rating, ProfileImagePath.
- **Specializations**: SpecializationId, Name.
- **Appointments**: AppointmentId, UserId, DoctorId, AppointmentDate, TimeSlot, Status.
- **Ratings**: RatingId, DoctorId, UserId, Rating, Comment.

ER Diagram :



SPECIALIZATIONS		
int	SpecializationId	PK
string	SpecializationName	

DOCTORS		
int	DoctorId	PK
string	Name	
int	SpecializationId	FK
string	City	
float	Rating	
string	Imagepath	

USERS		
int	UserId	PK
string	Username	
string	Password	
string	Role	
string	ImagePath	

APPOINTMENTS		
int	AppointmentId	PK
int	UserId	FK
int	DoctorId	FK
date	AppointmentDate	
string	TimeSlot	
string	Status	

RATINGS		
int	RatingId	PK
int	UserId	FK
int	DoctorId	FK
int	Rating	

Data Handling Strategies

- Referential integrity enforced with **foreign keys**.
- Indexes on **UserId, DoctorId, SpecializationId** for faster queries.
- Frontend caches data like specializations to reduce repeated API calls.

---

6. Testing Strategy

Backend

- **Frameworks**: xUnit with Moq.
- **Approach**: EF Core InMemory database to test controllers without affecting production.
- **Scenarios Tested**:

    - User registration and login produce valid JWTs.
    - Booking an appointment creates a record.
    - Admin approval updates appointment status correctly.

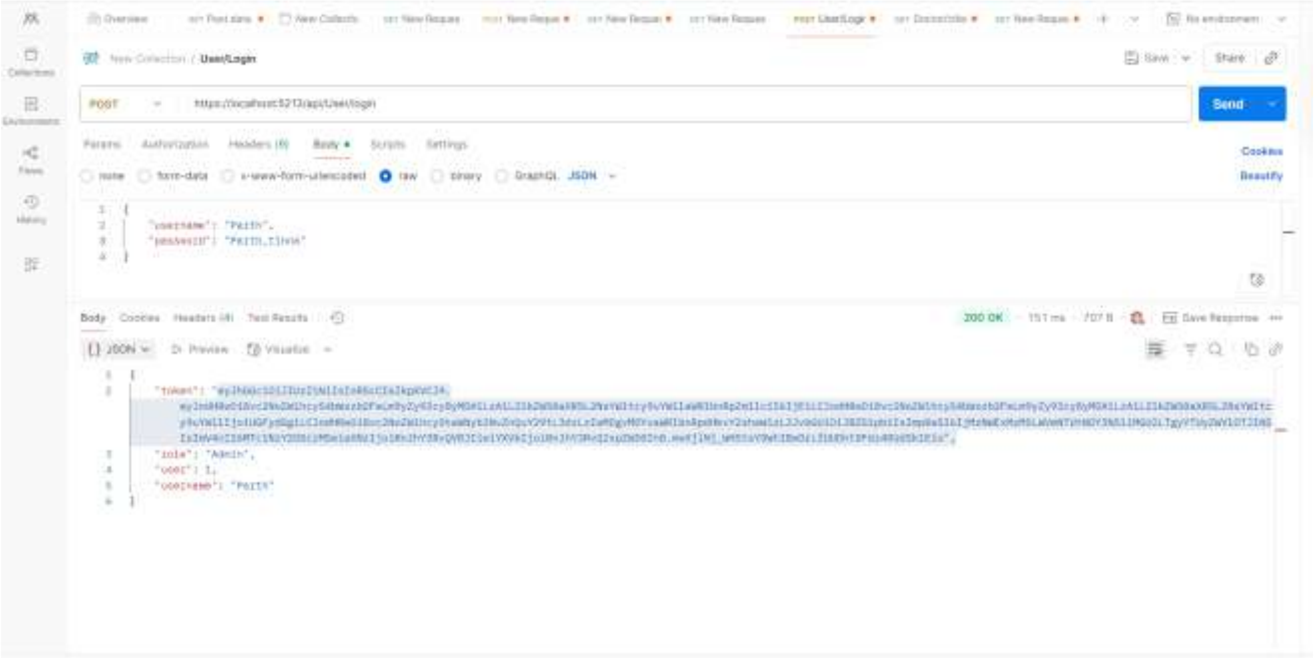
Frontend

- **Tools**: Jasmine + Karma.
- **Unit Tests**:
    - Login form validates required fields.
    - Appointment booking triggers the correct API call.
    - Doctor search renders accurate data.

Integration

- **Postman** used for testing REST APIs.
- Manual linking tests with Angular frontend and ASP.NET backend.
- Future E2E scope with Cypress.

7. End to End Testing via Post Man



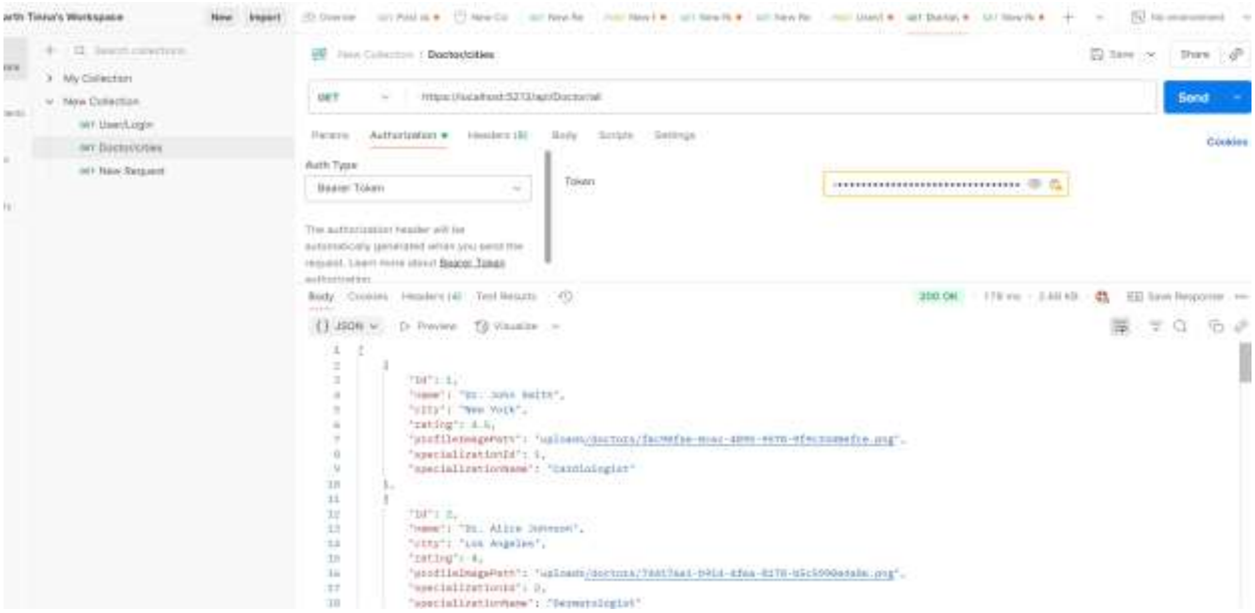
New Collection / User/Login

POST https://localhost:5213/api/User/login

```
{
  "username": "Parth",
  "password": "[illegible]"
}
```

200 OK · 151 ms · 707 B

```
{
  "token": "[illegible]",
  "role": "Admin",
  "userId": 1,
  "username": "Parth"
}
```



GET https://localhost:5213/api/Doctor/all

Auth Type: Bearer Token

Token: ••••••••••••••••••••••••••••••••

The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.

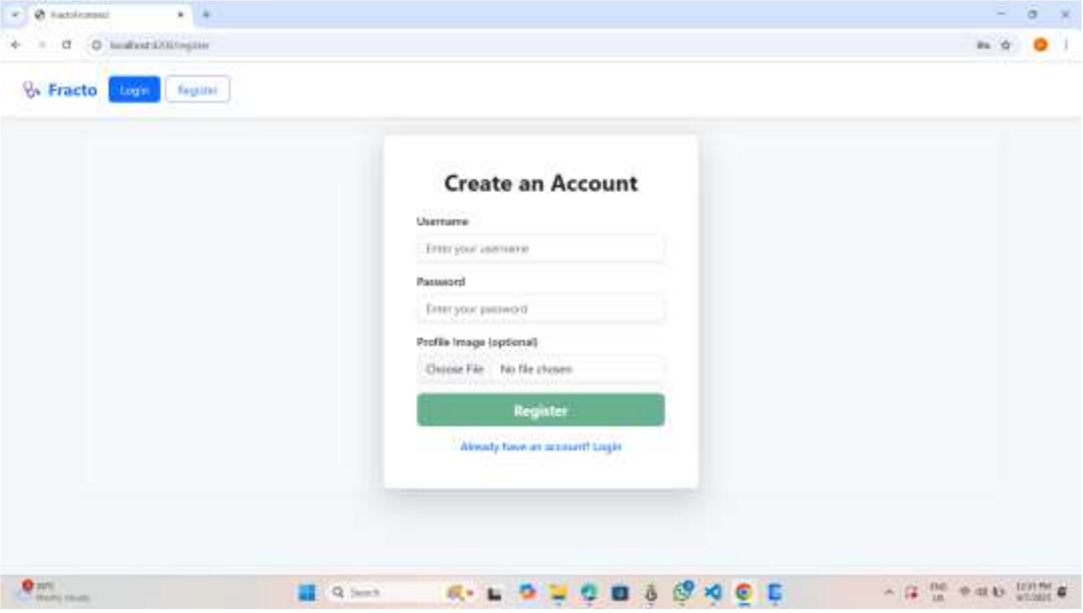
200 OK · 178 ms · 2.48 KB

JSON

```
[
    {
        "id": 1,
        "name": "Dr. John Smith",
        "city": "New York",
        "rating": 4.5,
        "profileImagePath": "uploads/doctors/[illegible].png",
        "specializationId": 1,
        "specializationName": "Cardiologist"
    },
    {
        "id": 2,
        "name": "Dr. Alice Johnson",
        "city": "Los Angeles",
        "rating": 4,
        "profileImagePath": "uploads/doctors/[illegible].png",
        "specializationId": 2,
        "specializationName": "Dermatologist"
```

8. Implementation Screenshots

User Register:



Create an Account

Username
Enter your username

Password
Enter your password

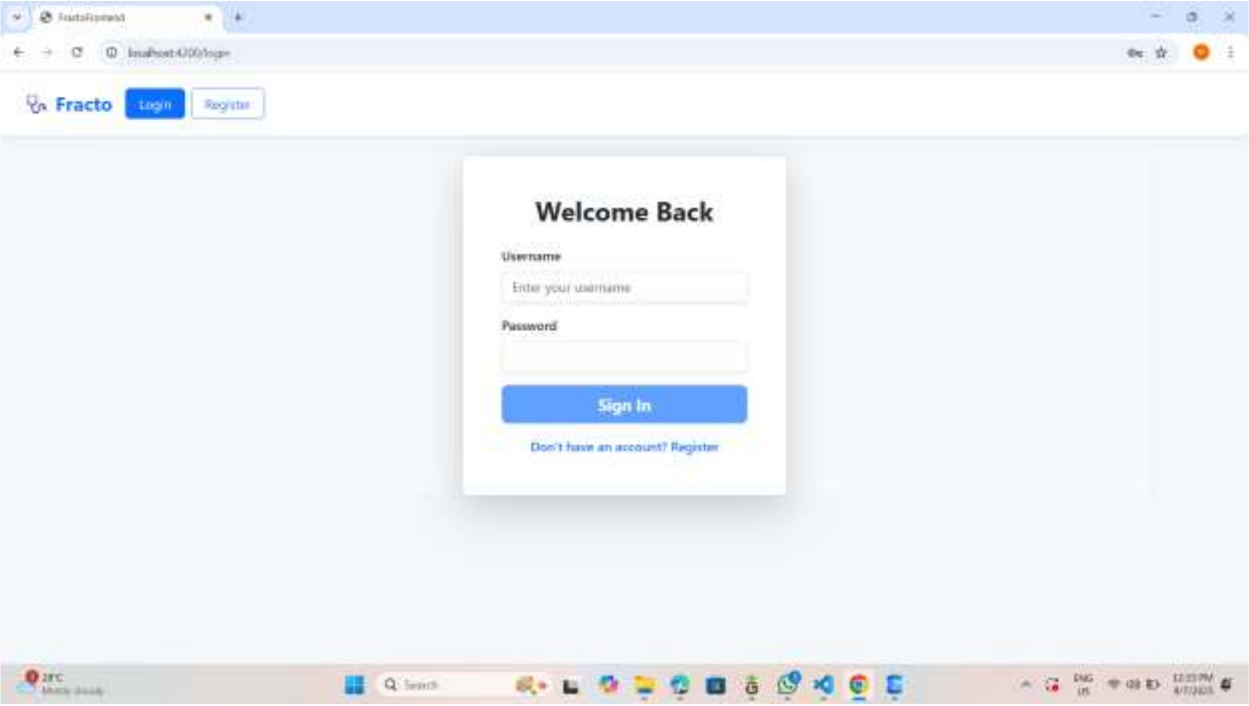
Profile Image (optional)
Choose File No file chosen

Register

Already have an account? Login

:

User Login:



Welcome Back

Username

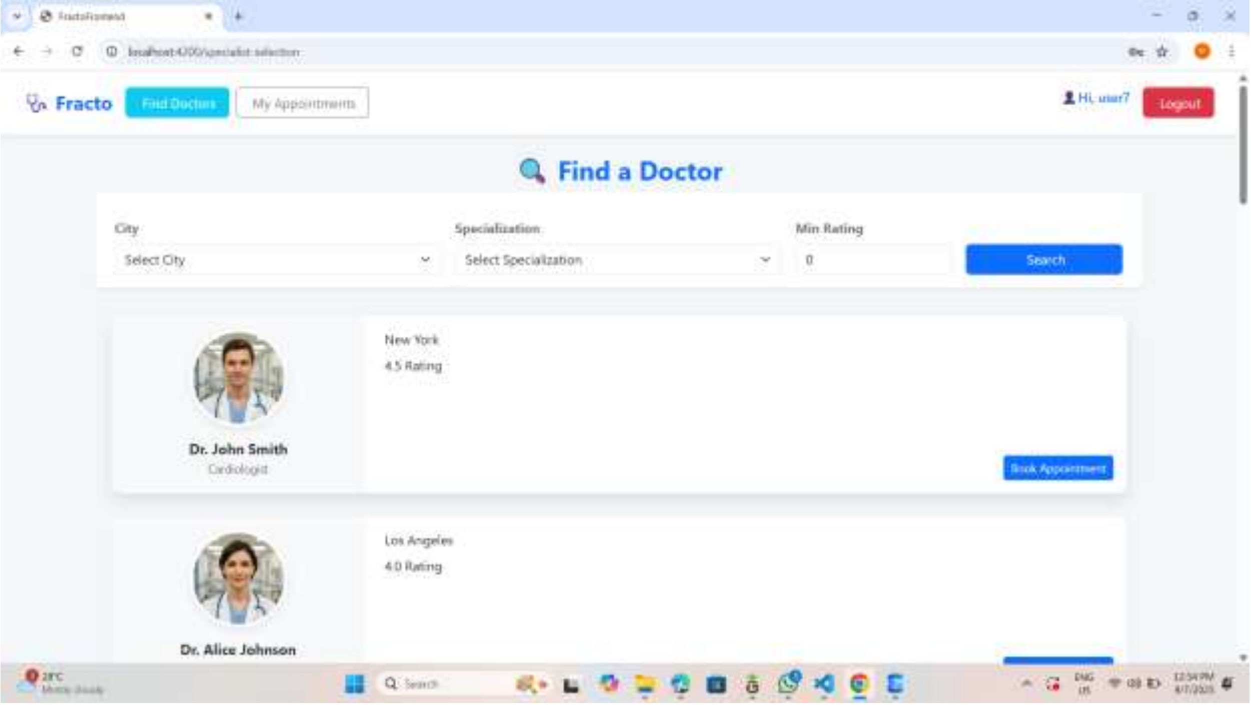
Enter your username

Password

Sign In

Don't have an account? Register

Doctor’s List:



Find a Doctor

City: Select City

Specialization: Select Specialization

Min Rating: 0

Search

Dr. John Smith

Cardiologist

New York

4.5 Rating

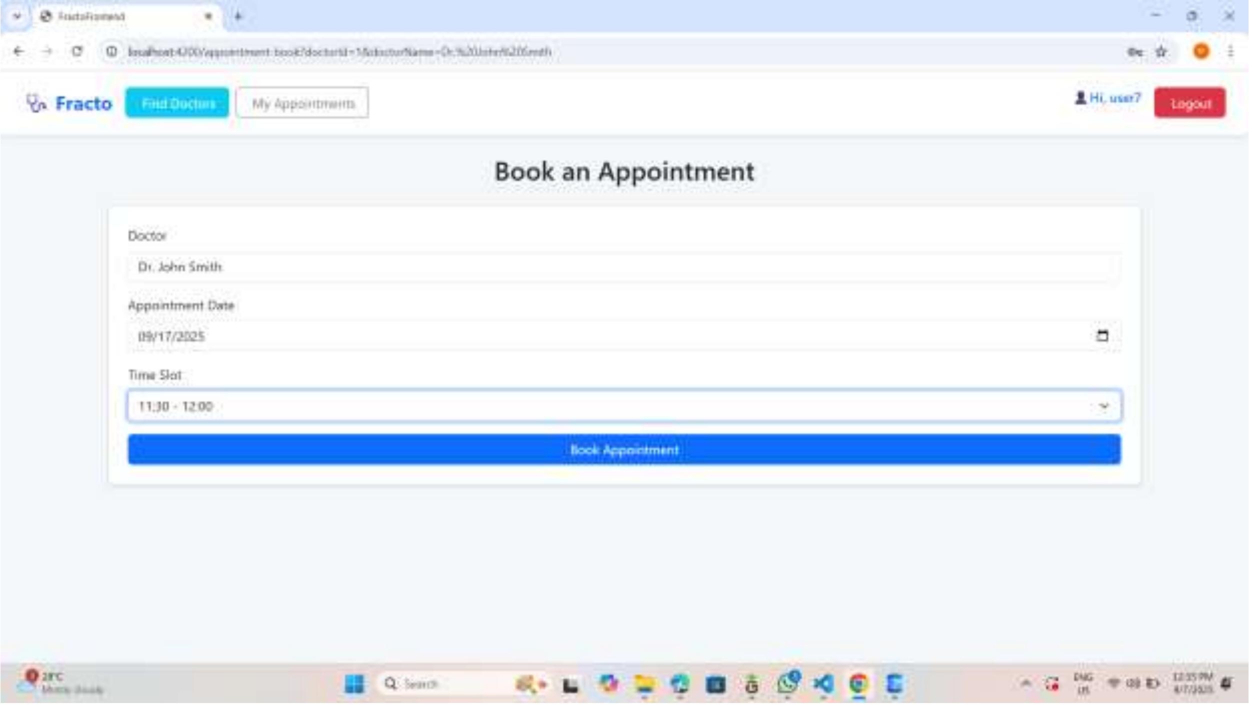
Book Appointment

Dr. Alice Johnson

Los Angeles

4.0 Rating

Booking Appointment On Basis of Slot:


FractoFrontend
localhost:4200/appointment-book?doctorId=1&doctorName=Dr.%20John%20Smith
Fracto
Find Doctors
My Appointments
Hi, user7
Logout

Book an Appointment

Doctor

Dr. John Smith

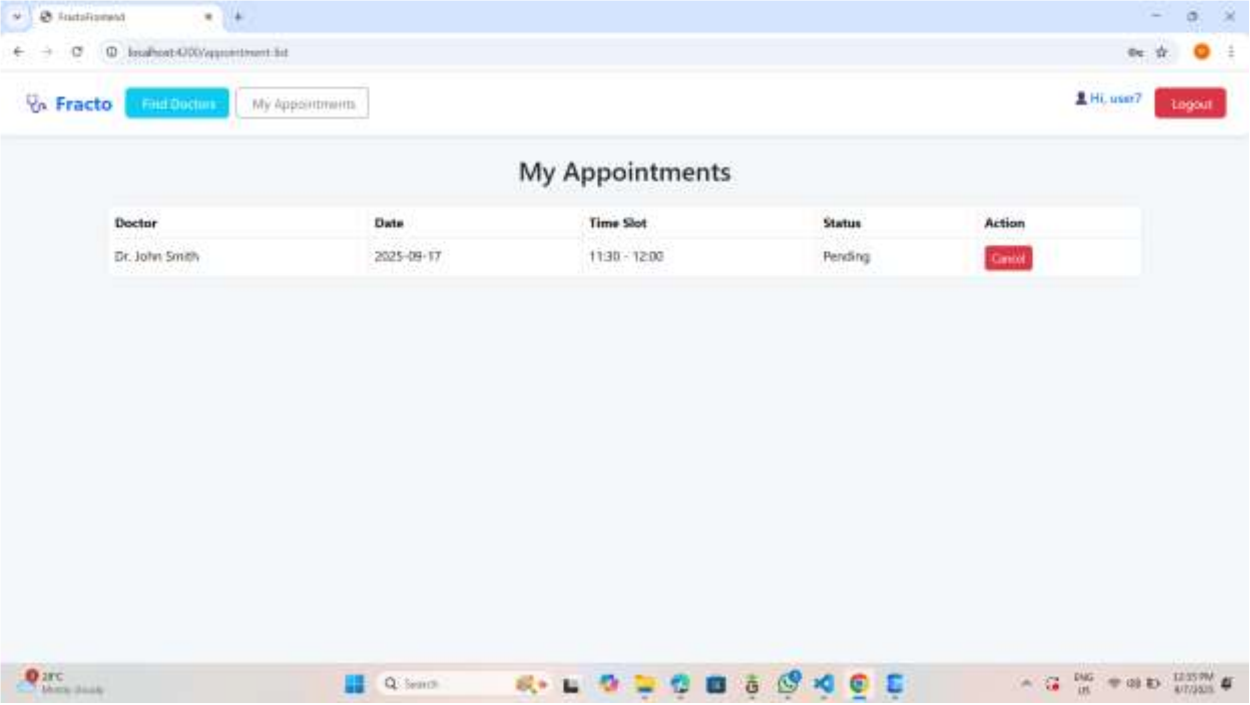
Appointment Date

09/17/2025

Time Slot

11:30 - 12:00

Checking Appointments:

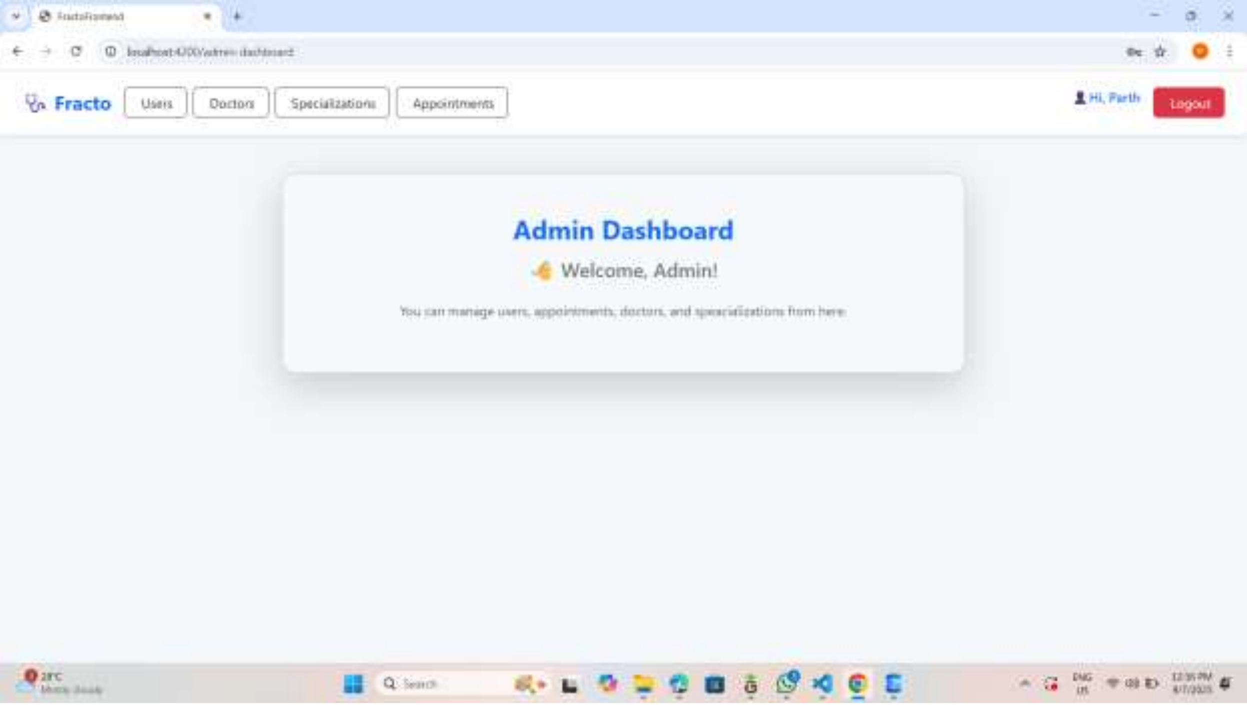


My Appointments

Doctor	Date	Time Slot	Status	Action
Dr. John Smith	2025-09-17	11:30 - 12:00	Pending	Cancel

Admin Side

Admins Dashboard:

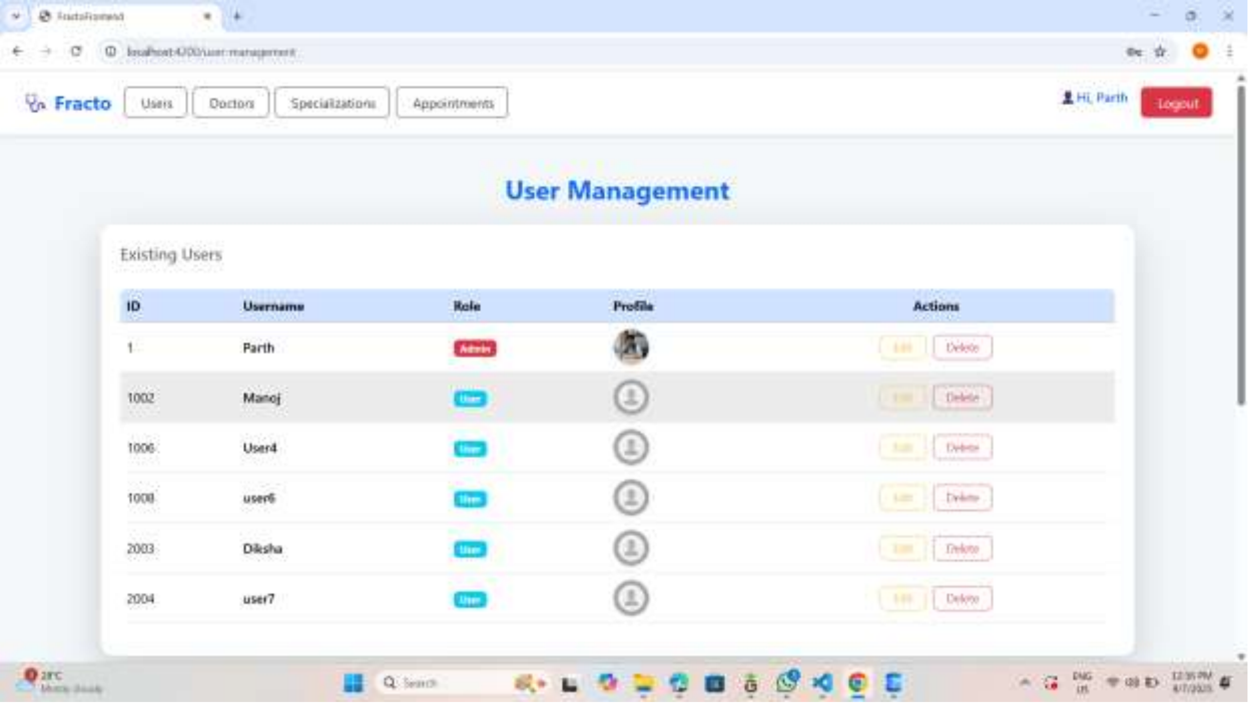


Admin Dashboard

👋 Welcome, Admin!

You can manage users, appointments, doctors, and specializations from here.

Admin User Management:

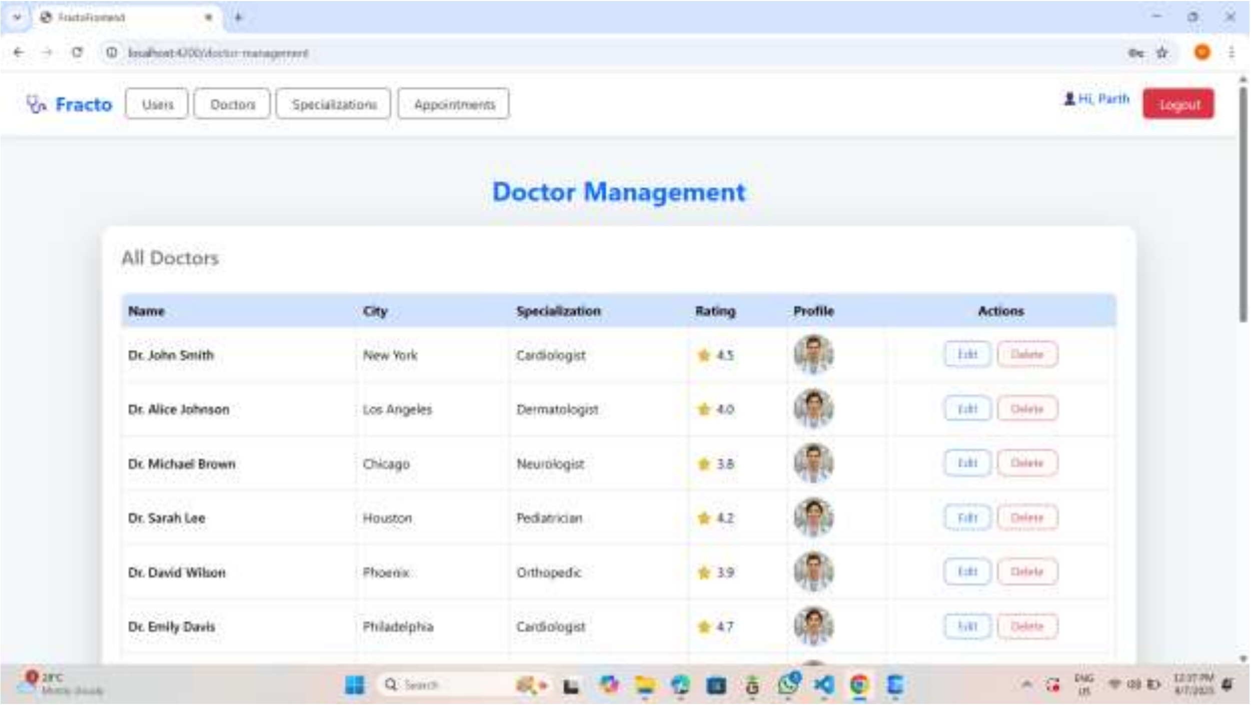


User Management

Existing Users

ID	Username	Role	Profile	Actions
1	Parth	Admin		Edit Delete
1002	Manoj	User		Edit Delete
1006	User4	User		Edit Delete
1008	user6	User		Edit Delete
2003	Diksha	User		Edit Delete
2004	user7	User		Edit Delete

Admin Doctors Management:

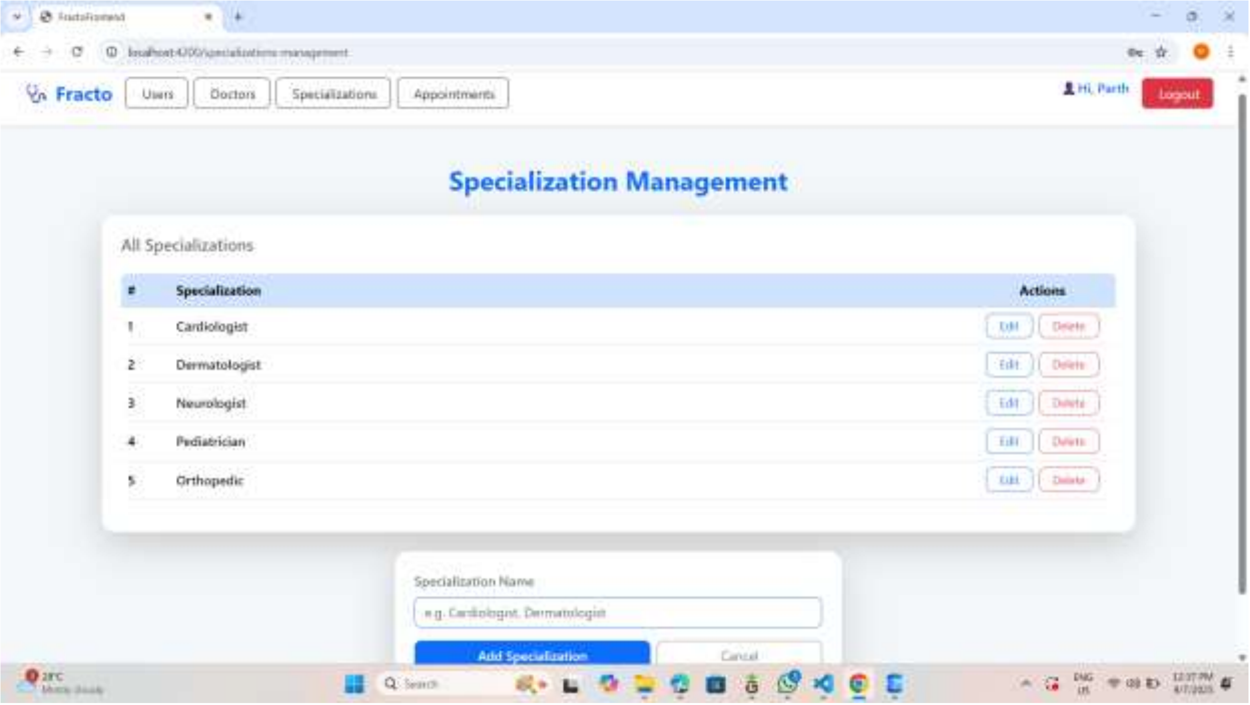


Doctor Management

All Doctors

Name	City	Specialization	Rating	Profile	Actions
Dr. John Smith	New York	Cardiologist	★ 4.5		Edit Delete
Dr. Alice Johnson	Los Angeles	Dermatologist	★ 4.0		Edit Delete
Dr. Michael Brown	Chicago	Neurologist	★ 3.8		Edit Delete
Dr. Sarah Lee	Houston	Pediatrician	★ 4.2		Edit Delete
Dr. David Wilson	Phoenix	Orthopedic	★ 3.9		Edit Delete
Dr. Emily Davis	Philadelphia	Cardiologist	★ 4.7		Edit Delete

Admin Specializations Management



Specialization Management

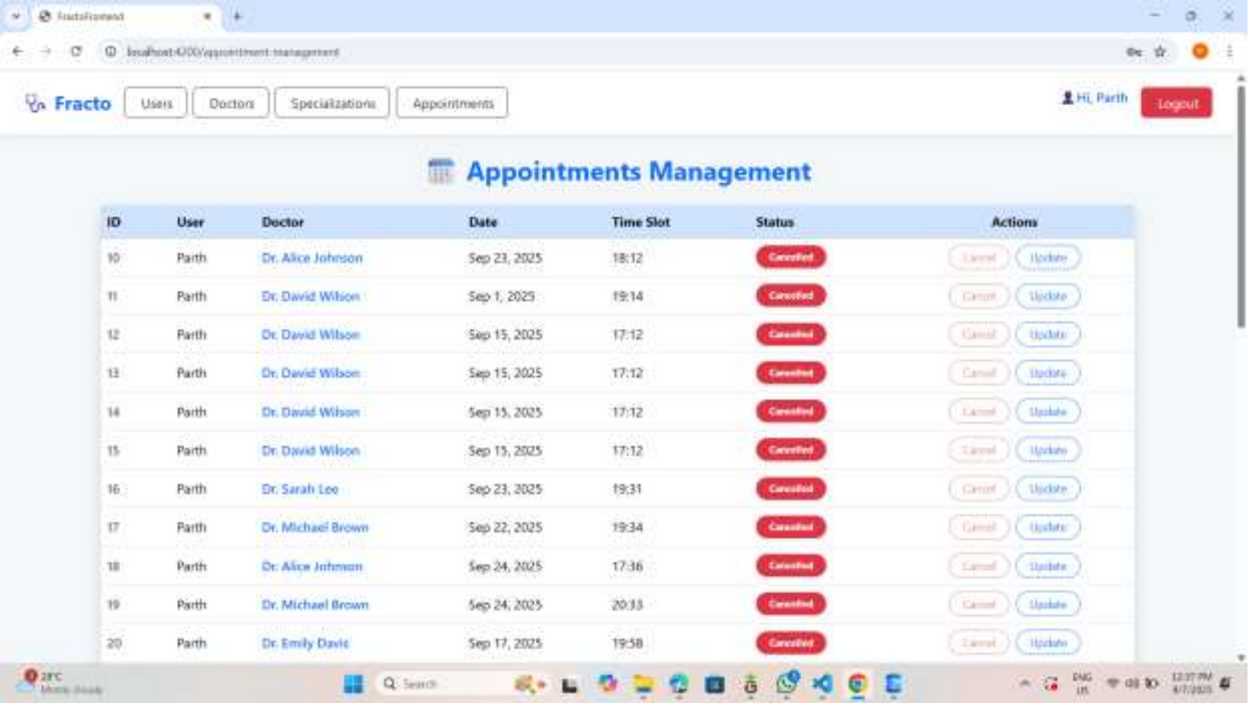
All Specializations

#	Specialization	Actions
1	Cardiologist	Edit Delete
2	Dermatologist	Edit Delete
3	Neurologist	Edit Delete
4	Pediatrician	Edit Delete
5	Orthopedic	Edit Delete

Specialization Name

e.g. Cardiologist, Dermatologist

Admin Appointment Management:



Appointments Management

ID	User	Doctor	Date	Time Slot	Status	Actions
10	Parth	Dr. Alice Johnson	Sep 23, 2025	18:12	Cancelled	Cancel Update
11	Parth	Dr. David Wilson	Sep 1, 2025	19:14	Cancelled	Cancel Update
12	Parth	Dr. David Wilson	Sep 15, 2025	17:12	Cancelled	Cancel Update
13	Parth	Dr. David Wilson	Sep 15, 2025	17:12	Cancelled	Cancel Update
14	Parth	Dr. David Wilson	Sep 15, 2025	17:12	Cancelled	Cancel Update
15	Parth	Dr. David Wilson	Sep 15, 2025	17:12	Cancelled	Cancel Update
16	Parth	Dr. Sarah Lee	Sep 23, 2025	19:31	Cancelled	Cancel Update
17	Parth	Dr. Michael Brown	Sep 22, 2025	19:34	Cancelled	Cancel Update
18	Parth	Dr. Alice Johnson	Sep 24, 2025	17:36	Cancelled	Cancel Update
19	Parth	Dr. Michael Brown	Sep 24, 2025	20:33	Cancelled	Cancel Update
20	Parth	Dr. Emily Davis	Sep 17, 2025	19:58	Cancelled	Cancel Update

9. Conclusion

Fracto has been built to streamline healthcare appointment management through a digital-first solution. It bridges the gap between patients and healthcare providers with **real-time visibility, secure access, and user-friendly interfaces**.

The project successfully delivers:

- **Ease for patients** in booking and managing appointments.
- **Efficiency for administrators** in overseeing users, doctors, and schedules.
- **Security** with robust authentication and role-based permissions.
- **Engagement** through ratings and instant notifications.