**Project Idea: Doctor Appointment System**

**Features:**

1. **User Registration and Authentication**:
   * Separate roles for **Patients** and **Doctors**.
   * Patients can register, log in, and manage their profiles.
   * Doctors can register, log in, and manage their schedules and profiles.
2. **Appointment Booking**:
   * Patients can browse available doctors by specialty, location, or ratings and book appointments.
   * Doctors can view and approve/reject appointments.
3. **Doctor Profile**:
   * Doctors can list their specialties, availability, clinic address, and consultation fees.
   * Patients can view doctor profiles and book appointments based on availability.
4. **Appointment Management**:
   * **Patients**: Can view, modify, or cancel their booked appointments.
   * **Doctors**: Can manage appointment requests, update availability, and view past appointments.
5. **Notifications**:
   * Email or SMS notifications for booking confirmations, reminders, and cancellations.
6. **Reviews and Ratings**:
   * After the consultation, patients can leave reviews and ratings for doctors.
7. **Admin Panel** (Optional):
   * Admins can manage the overall system, doctors, patients, and appointments.

**Tech Stack:**

* **Frontend (ReactJS)**:
  + Create forms for booking appointments, doctor profiles, and registration.
  + Use axios or fetch to connect with the backend APIs.
  + Use Bootstrap to create a responsive UI for doctors and patients.
  + Build dashboards for doctors (appointments) and patients (appointment history).
* **Backend (Spring Boot)**:
  + Create RESTful APIs for managing user authentication, doctor profiles, appointments, and reviews.
  + Implement different user roles (Patients, Doctors, Admins).
  + Handle email notifications for appointment confirmations and reminders using Spring Boot mail services.
  + Use JPA for MySQL database interactions.
* **Database (MySQL)**:
  + **Users Table**: Stores patient and doctor information.
  + **Doctors Table**: Stores doctor-specific information like specialty, availability, and clinic details.
  + **Appointments Table**: Stores appointment details with foreign keys referencing patients and doctors.
  + **Reviews Table**: Stores patient reviews and ratings for doctors.

**Additional Features:**

* **Search Filters**: Allow patients to search for doctors by location, specialization, and availability.
* **Doctor Availability**: Doctors can update their available time slots, and patients can only book within those time slots.
* **Appointment History**: Patients and doctors can view past appointments and their status.

**Project Flow:**

1. **Patient Side**:
   * Register/Login → View Doctor Profiles → Book Appointment → Get Notification → View Appointment History.
2. **Doctor Side**:
   * Register/Login → Update Availability → View Appointment Requests → Approve/Reject Appointments → View Patient Details.
3. **Admin Side** (Optional):
   * Manage Users, Doctors, and Appointments.

**Potential Extensions:**

* **Payment Integration**: Allow patients to pay consultation fees online.
* **Video Consultation**: Enable video consultations through integration with video call APIs (e.g., Zoom, Twilio).
* **Prescription Management**: Doctors can upload digital prescriptions after consultations.

This project covers real-world features and is an excellent opportunity to deepen your knowledge of full-stack development with MySQL, Spring Boot, and ReactJS.