To test the Patient Management System with role-based access control using Postman, you’ll follow these steps:

1. **Register and Login** each type of user (Patient, Doctor, Admin).
2. Use the **JWT token** from each login to test the access to endpoints according to the **access control grid**.
3. Test the **Patient Records** and **Appointments** endpoints with different roles to confirm that only the allowed actions are permitted.

Below is a guide on how to set up and test each part of the system in Postman.

**Step 1: User Registration and Login**

1. **Register a User (Patient, Doctor, Admin)**
   * **Method**: POST
   * **URL**: http://localhost:5000/api/auth/register
   * **Body (JSON)**:

{

"email": "patient@example.com",

"password": "password123",

"role": "Patient"

}

* + Repeat this step for roles Doctor and Admin by changing the email and role in the request body.

1. **Login as Each User**
   * **Method**: POST
   * **URL**: http://localhost:5000/api/auth/login
   * **Body (JSON)**:

{

"email": "patient@example.com",

"password": "password123"

}

* + Copy the token from each login response. This token will be used in the **Authorization** header for subsequent requests.

**Step 2: Test Patient Record Management**

Using the JWT tokens obtained in Step 1, test the patient record endpoints according to the role-based access grid.

**a. Create Patient Record (Doctors and Admins Only)**

* **Method**: POST
* **URL**: http://localhost:5000/api/patient-records
* **Headers**:
  + Authorization: Bearer <TOKEN\_FROM\_DOCTOR\_OR\_ADMIN\_LOGIN>
* **Body (JSON)**:

{

"name": "John Doe",

"age": 30,

"doctorId": 2

}

* **Expected Result**:
  + **Doctor** or **Admin**: Success with a response like { "message": "Patient record created successfully" }
  + **Patient**: Unauthorized response, e.g., { "error": "Access denied" }

**b. Read Patient Records**

1. **Doctor**: Should only read assigned patient records.
   * **Method**: GET
   * **URL**: http://localhost:5000/api/patient-records
   * **Headers**:
     + Authorization: Bearer <TOKEN\_FROM\_DOCTOR\_LOGIN>
   * **Expected Result**: List of patient records assigned to this doctor.
2. **Patient**: Should only read their own records.
   * **Method**: GET
   * **URL**: http://localhost:5000/api/patient-records
   * **Headers**:
     + Authorization: Bearer <TOKEN\_FROM\_PATIENT\_LOGIN>
   * **Expected Result**: The patient's own record.
3. **Admin**: Should access all patient records.
   * **Method**: GET
   * **URL**: http://localhost:5000/api/patient-records
   * **Headers**:
     + Authorization: Bearer <TOKEN\_FROM\_ADMIN\_LOGIN>
   * **Expected Result**: List of all patient records.

**c. Update Patient Record (Doctors and Admins Only)**

* **Method**: PUT
* **URL**: http://localhost:5000/api/patient-records/:recordId
* **Headers**:
  + Authorization: Bearer <TOKEN\_FROM\_DOCTOR\_OR\_ADMIN\_LOGIN>
* **Body (JSON)**:

{

"name": "Updated Name",

"age": 35

}

* **Expected Result**:
  + **Doctor** (if record is assigned to them) or **Admin**: Success with response { "message": "Patient record updated successfully" }
  + **Patient**: Unauthorized response, e.g., { "error": "Access denied" }

**d. Delete Patient Record (Admin Only)**

* **Method**: DELETE
* **URL**: http://localhost:5000/api/patient-records/:recordId
* **Headers**:
  + Authorization: Bearer <TOKEN\_FROM\_ADMIN\_LOGIN>
* **Expected Result**:
  + **Admin**: Success with response { "message": "Patient record deleted successfully" }
  + **Doctor** or **Patient**: Unauthorized response, e.g., { "error": "Access denied" }

**Step 3: Test Appointment Management**

Follow the steps below to test appointment functionality with each role.

**a. Create Appointment (Patients and Admins Only)**

* **Method**: POST
* **URL**: http://localhost:5000/api/appointments
* **Headers**:
  + Authorization: Bearer <TOKEN\_FROM\_PATIENT\_OR\_ADMIN\_LOGIN>
* **Body (JSON)**:

{

"doctorId": 2,

"patientId": 1,

"date": "2024-12-01 10:00:00",

"status": "scheduled"

}

* **Expected Result**:
  + **Patient** or **Admin**: Success with response { "message": "Appointment created successfully" }
  + **Doctor**: Unauthorized response, e.g., { "error": "Access denied" }

**b. Read Appointments**

1. **Doctor**: Should only read appointments assigned to them.
   * **Method**: GET
   * **URL**: http://localhost:5000/api/appointments
   * **Headers**:
     + Authorization: Bearer <TOKEN\_FROM\_DOCTOR\_LOGIN>
   * **Expected Result**: List of appointments assigned to this doctor.
2. **Patient**: Should only read their own appointments.
   * **Method**: GET
   * **URL**: http://localhost:5000/api/appointments
   * **Headers**:
     + Authorization: Bearer <TOKEN\_FROM\_PATIENT\_LOGIN>
   * **Expected Result**: List of the patient's own appointments.
3. **Admin**: Should access all appointments.
   * **Method**: GET
   * **URL**: http://localhost:5000/api/appointments
   * **Headers**:
     + Authorization: Bearer <TOKEN\_FROM\_ADMIN\_LOGIN>
   * **Expected Result**: List of all appointments.

**c. Update Appointment (Patients and Doctors Only)**

* **Method**: PUT
* **URL**: http://localhost:5000/api/appointments/:appointmentId
* **Headers**:
  + Authorization: Bearer <TOKEN\_FROM\_PATIENT\_OR\_DOCTOR\_LOGIN>
* **Body (JSON)**:

{

"date": "2024-12-10 11:00:00",

"status": "completed"

}

* **Expected Result**:
  + **Patient** (if appointment is their own) or **Doctor** (if appointment is assigned to them): Success with response { "message": "Appointment updated successfully" }
  + **Admin**: Unauthorized response, e.g., { "error": "Access denied" }

**d. Delete Appointment (Admins Only)**

* **Method**: DELETE
* **URL**: http://localhost:5000/api/appointments/:appointmentId
* **Headers**:
  + Authorization: Bearer <TOKEN\_FROM\_ADMIN\_LOGIN>
* **Expected Result**:
  + **Admin**: Success with response { "message": "Appointment deleted successfully" }
  + **Doctor** or **Patient**: Unauthorized response, e.g., { "error": "Access denied" }