Hospital Appointment Management System - Project Report.

1. Application Overview: What's the Deal?

This app is a simple hospital management system. The core goal is to simplify appointment scheduling between patients and doctors, and also to manage user info (patients, doctors, admins). Basically, it helps patients book appointments easily, lets doctors see their schedules, and gives admins the tools to manage everything. The aim is to provide a system that makes hospital admin more efficient and organized.

2. GUI Elements: Buttons and Stuff

When you fire up the program, you'll see a bunch of interface elements that let you interact with it. Here's a rundown of the main ones:

- ***Login/Registration Window (LoginForm.java):***
  - ***Text Fields (loginEmail, loginPassword):*** These are where you type in your email/phone number and password to log in.
  - ***"Login" Button (loginBtn):*** Click this after entering your details to access your account.
  - ***Text Fields (signName, signEmailPhone, signPass, signPhone):*** Use these to create a new account (name, email/phone, password, phone).
  - ***Dropdown Menu (signRole):*** Choose your user type here (patient, doctor, or admin).
  - ***"Register" Button (registerBtn):*** Click this after filling in your details to create a new account.
- ***Patient Dashboard (PatientDashboard.java):***
  - ***Text Area (appointmentsArea):*** This displays the appointments you've booked.
  - ***"View Appointments" Button (viewAppointmentsBtn):*** Click this to see your appointments.
  - ***"Book Appointment" Button (bookAppointmentBtn):*** Click this to schedule a new appointment.

    After booking an appointment, a small pop-up shows up to tell the user that the booking was successful.

  - ***"Cancel Appointment" Button (cancelAppointmentBtn):*** Click this to cancel an appointment.
  - ***Text Field (searchField) and "Search" Button (searchDoctorBtn):*** Use these to search for a specific doctor.
- ***Doctor Dashboard (DoctorDashboard.java):***
  - ***Text Area (appointmentsArea):*** This shows your appointments with patients.
  - ***"View Appointments" Button (viewAppointmentsBtn):*** Click this to view your schedule.
  - ***"Cancel Appointment" Button (cancelBtn):*** Click this to cancel an appointment.

- o ***"Reschedule Appointment" Button (rescheduleBtn):*** Click this to change the date/time of an appointment.

Admin Dashboard (AdminDashboard.java):

- o ***Text Area (reportArea):*** This displays reports on appointments and user info.
- o ***"View Reports" Button (viewReportsBtn):*** Click this to view the reports.
- o ***"Add Doctor" Button (addDoctorBtn):*** Click this to add a new doctor.
- o ***"Delete Doctor" Button (deleteDoctorBtn):*** Click this to remove a doctor.
- o ***"Add Patient" Button (addPatientBtn):*** Click this to add a new patient.
- o ***"Delete Patient" Button (deletePatientBtn):*** Click this to remove a patient.
- o ***"Update Appointment Status" Button (updateStatusBtn):*** Click this to change the status of an appointment (e.g., from "pending" to "confirmed").

3. Database Connection: Storing the Data

To store appointment and user information, the app needs to talk to a database. I used SQLite (the database file is "hospital.db"). I connected to it using the JDBC library in Java.

Think of the database like a big, organized notebook, where each page (table) stores specific types of info. For example:

"users" table: This table stores user info (ID, name, email, password, phone, user role).
"appointments" table: This table stores appointment details (appointment ID, patient ID, doctor ID, date, time, status).

4. Exception Handling: Dealing with Errors

It's normal for errors or small issues to pop up while using an app. So, I tried to handle as many of these situations as possible to prevent crashes or a bad user experience. Here are some errors I considered and how I dealt with them:

Login Errors: If you enter the wrong email/phone or password, you'll get an error message telling you the login details are incorrect.
Registration Errors: If you try to register with an email that already exists or use a weak password, you'll get a message explaining the issue.
Appointment Booking Errors: If you try to book an appointment at a time that's already taken, you'll be asked to choose another time.
Database Connection Errors: If there's a problem connecting to the database, an error message will appear.
General Errors (Exception): I used try-catch blocks to catch any unexpected errors and display a helpful message to the user.

5. Testing and Validation: Making Sure It Works

After coding the app, I put it through its paces to make sure everything works correctly. I did manual testing to check:

Login and registration work properly.

Each dashboard (patient, doctor, admin) displays the right information and lets users perform their intended actions.
Booking, modifying, and canceling appointments work smoothly.
Searching for doctors works as expected.
The app handles errors gracefully and doesn't crash.