Hospital Management System Project

Group members:

Name	Roll No
Ruthwik K	AM.EN.U4EAC21037
Srikar Nadupalle	AM.EN.U4EAC21050
Lokesh Vadakoppula	AM.EN.U4ECE21128
Tejasvin V	AM.EN.U4ECE21156

Project Report: Hospital Management System

Project Overview:

The Hospital Management System is a comprehensive web application designed to streamline hospital operations. Developed using Angular for the frontend and Spring Boot for the backend, the system provides functionality for managing patients, appointments, and medicines efficiently. It aims to serve as an intuitive platform for doctors and administrators.

Objective:

To build a reliable and user-friendly hospital management application that:

1. Enables doctors to manage patient information and medicine inventory seamlessly.
2. Provides administrators with tools to manage appointments and track hospital activities.

Core Features

Doctor Module

1. Dashboard:
    a. Displays a list of current patients and their diagnostic details.
2. CRUD Operations:
    a. Add, update, view, and delete patient records.
3. Medicine Management:
    a. Maintain and update a list of medicines.

Admin Module

1. Dashboard:
    a. Lists all patients (excluding sensitive diagnostic information).
2. Appointment Management:
    a. CRUD operations for hospital appointments.

Authentication

- Role-based login for doctors and administrators to ensure secure access.

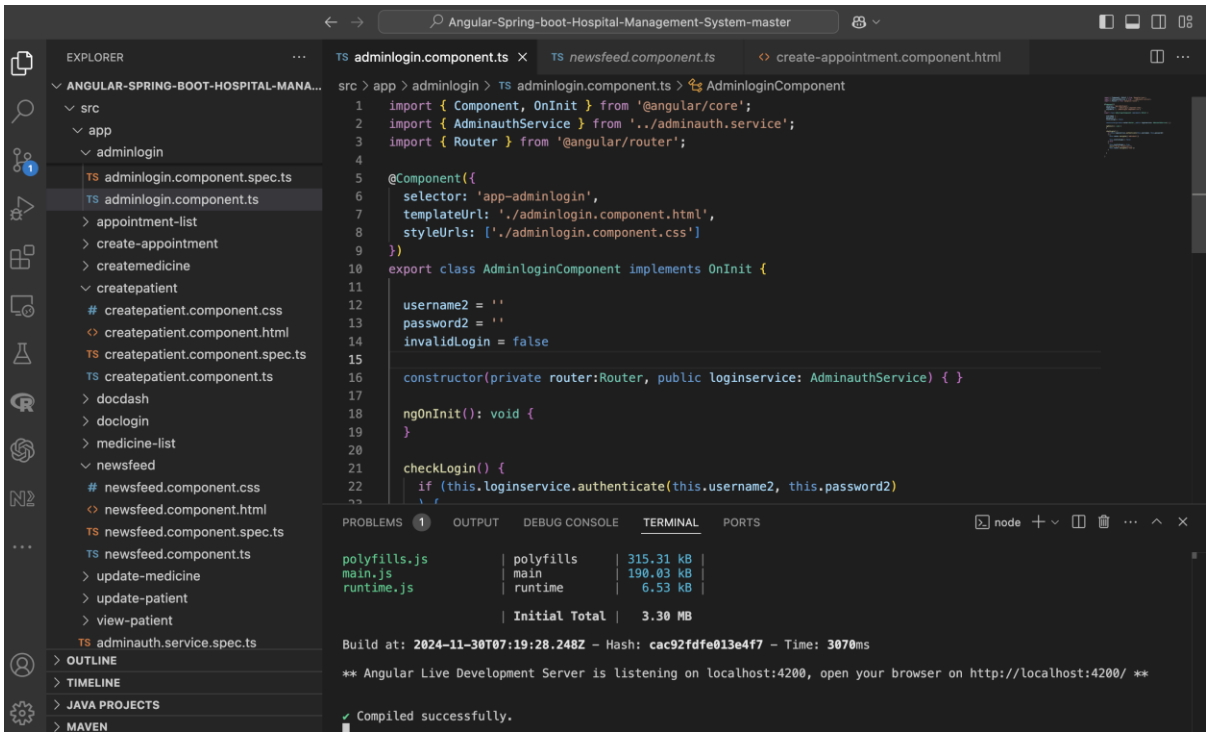
Search Functionality

- Search for patients by name to quickly locate records.

Technology Stack

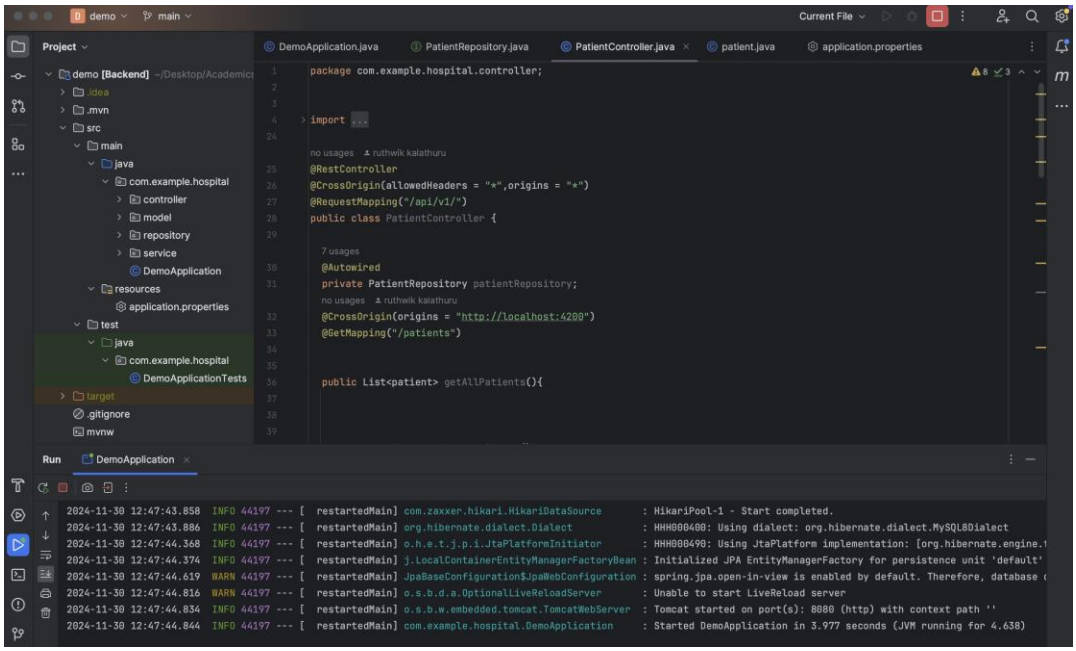
Frontend

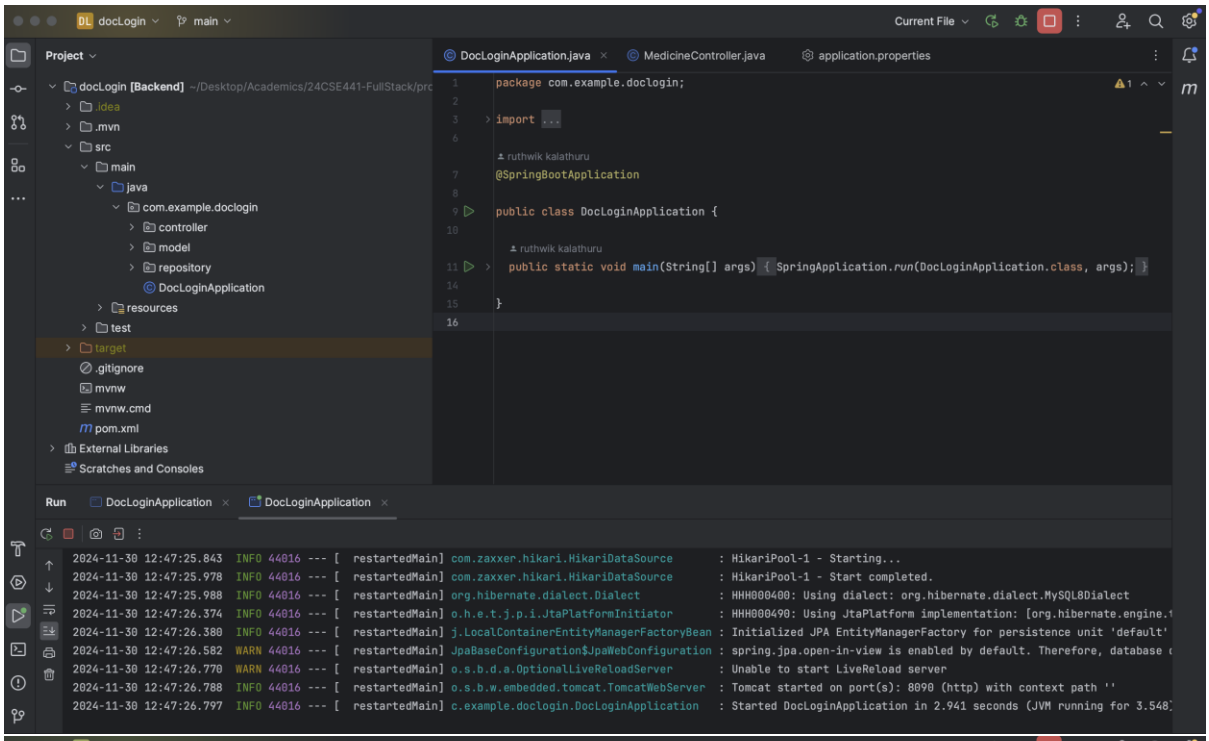
- **Framework**: Angular 14
- **Languages**: HTML, CSS, TypeScript, JavaScript
- **Tools**: Angular CLI for component generation, routing, state management, and API integration.



Backend

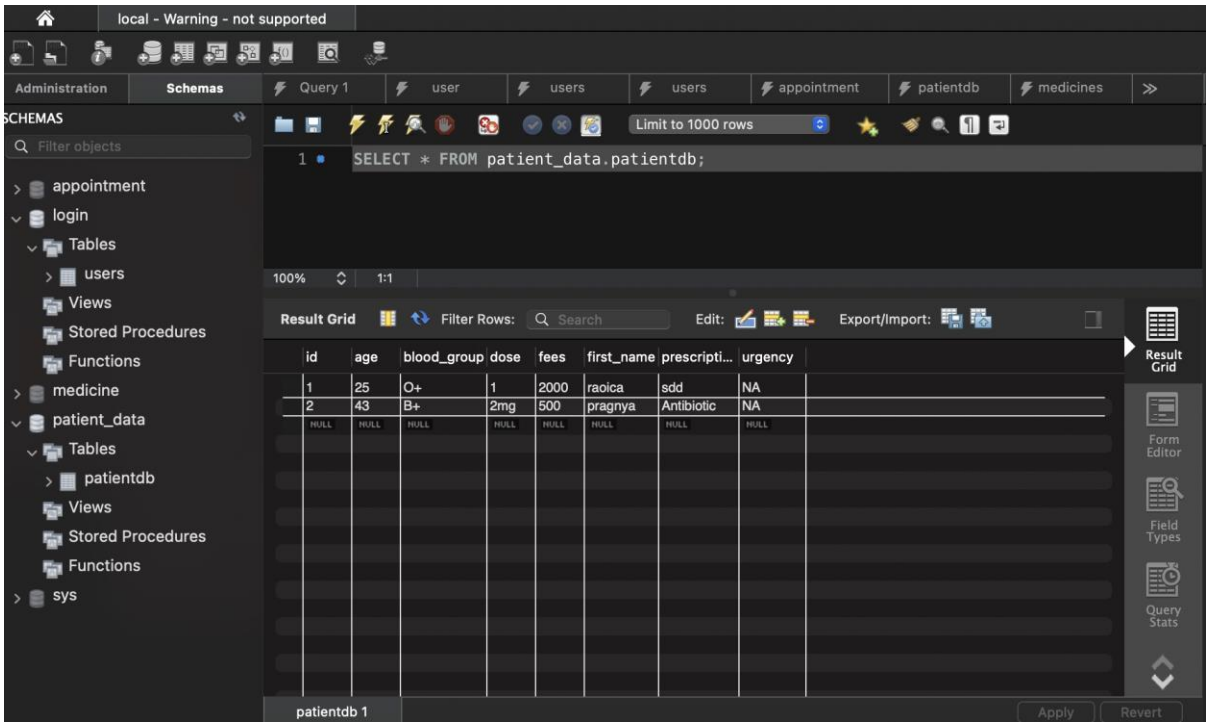
- **Framework**: Spring Boot
- **Language**: Java
- **Middleware Role**: Provides APIs for data transactions and database connectivity.





Database

- **Tool**: MySQL 8.0 with Workbench CE
- **Function**: Stores patient, appointment, and medicine data.



System Architecture

The application is based on a **Client-Server Model**, with Angular managing the client-side logic and Spring Boot acting as the middleware for database interactions. This architecture ensures modularity, scalability, and efficient communication between components.

Development Process

Frontend Development

- Built with reusable Angular components.
- API calls implemented for dynamic data rendering.
- Responsive design for better user experience.

Backend Development

- Developed RESTful APIs using Spring Boot.
- Secure database connectivity with MySQL 8.0.
- Integration of role-based authentication.

Deployment

1. **Local Development**:
    a. Run the frontend using ng serve at http://localhost:4200/.
    b. Backend server powered by Spring Boot (http://localhost:8080/ by default).
2. **Build and Production**:
    a. Use ng build to prepare production-ready frontend artifacts.
    b. Deploy the backend as a standalone Spring Boot JAR or WAR file.

Testing

1. **Unit Testing**:
    a. Execute ng test to run frontend unit tests with Karma.
    b. Backend tests can be implemented using JUnit.
2. **End-to-End Testing**:
    a. Run ng e2e for end-to-end testing after adding an appropriate test package.

Features Showcase

1. **Home Page**: A simple interface for navigation.
2. **Doctor Login**: Secure authentication to access dashboards.
3. **Doctor Dashboard**: Tools for managing patient and medicine records.
4. **Admin Dashboard**: Tools for appointment management and data tracking.
5. **Search Functionality**: Locate patient records by name.
6. **CRUD Components**: Simplified interfaces for adding and managing data.

Future Scope

1. **Enhanced Features**:
    a. Integration of email/SMS notifications for appointments.
    b. Advanced patient analytics with reporting tools.
2. **Scalability**:
    a. Cloud deployment for large-scale hospital networks.
    b. Enhanced database optimization for faster queries.
3. **Security**:

a. Implement OAuth for authentication.
b. Secure sensitive patient data with encryption.

Conclusion

The Hospital Management System successfully combines Angular's dynamic frontend capabilities with Spring Boot's robust backend features to deliver a powerful application. With scope for further enhancements, this project is well-suited for modern hospital management needs.

GitHub Repository: Hospital Management System