APP DEV PROJECT REPORT

Student Details

Name: Anushka Garg

Roll Number: 24f2007187

Email: 24f2007187@ds.study.iitm.ac.in

About Me: I am a student at IIT Madras BS Degree program with a deep interest in web application development and data-driven technologies. I am learning to build meaningful applications that combine learning, analytics, and user experience.

Project Details

Project Title: Hospital Management System

Problem Statement: To develop a secure, role-based Hospital Management System (HMS) that efficiently manages doctors, patient records, and appointment scheduling, for a hospital to replace slow, manual processes like paper records and phone scheduling and implementing distinct workflows for Administrators, Doctors, and Patients.

Approach: We started by creating a blueprint of all the information (the ER Diagram) to show exactly how patient data, doctor availability, and appointment records are connected. Then I built the login screen and added role-based security checks to every page. Then focused on building the Admin tools first, as the system can't run without doctors and their specialties being set up. Built other role tools and then connected the pieces.

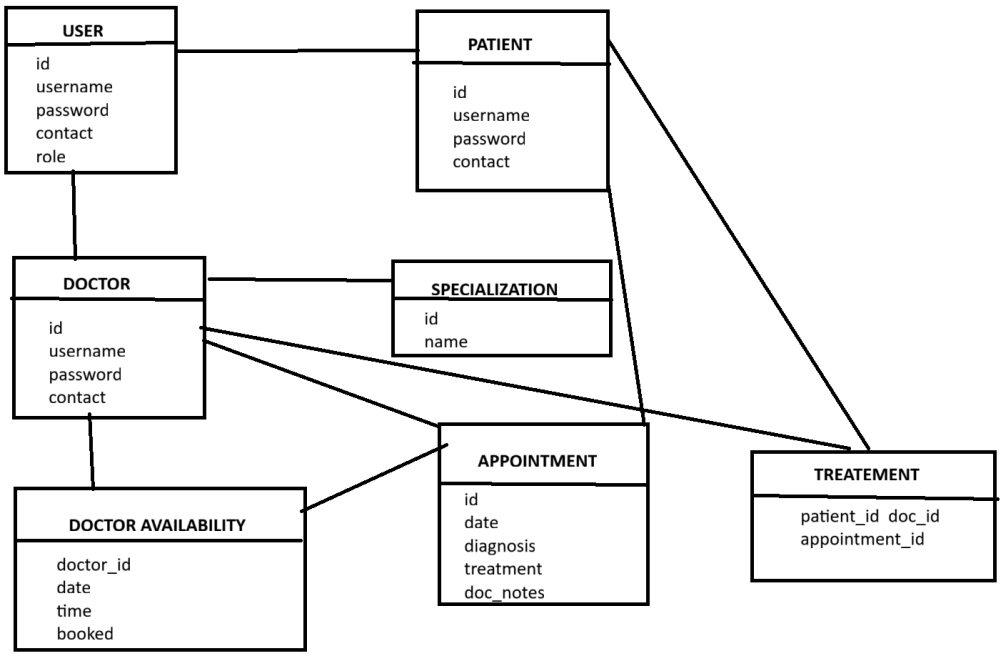
AI/LLM Declaration

This Hospital Management System project was developed with assistance from Gemini in writing the initial Python/Flask route logic, and identifying logical flaws in routing and database operations, and suggesting necessary security improvements.

Technologies and Frameworks Used

Category	Component	Purpose
Web Framework	**Flask**	Lightweight Python framework used for handling HTTP requests, routing, and structuring the application.
Database	**SQLite**	A simple, file-based relational database used for storing all application data (Users, Doctors, Appointments, etc.).
Database Connector	sqlite3 (Python standard library)	Used to connect to and execute SQL queries against the SQLite database.
Security	hashlib	Python standard library used for one-way password hashing (SHA-256) to secure user credentials.
Utilities	datetime	Used for managing dates, times, and performing calculations (e.g., checking if a slot is in the past, determining missed appointments).
Templating	**Jinja2**	Integrated with Flask for generating HTML pages dynamically based on server-side data.

ER DIAGRAM



API RESOURCE ENDPOINTS

Endpoint	Method	Purpose
/login	GET, POST	Handles user login authentication.
/logout	GET	Clears session and redirects to login.
/dashboard	GET	Redirects to the specific role dashboard.

/patient/register	GET, POST	Allows new users to register as a Patient.

Architecture and Features

Architecture Overview

- **app.py** - main Flask application entry point
- **database.py -** Contains all the database schema definition logic.
- **/templates** - Jinja2 HTML templates
- **/static** - CSS

Features

Admin

- Manages Doctors: Adds new doctors and assigns them specialties.
- Quality Control: Can temporarily "blacklist" a doctor, instantly preventing patients from booking them.
- Reporting: Sees big-picture reports, like a list of all appointments and all registered patients.

Doctor

- Sets Own Hours: Easily enters the dates and times they are available to see patients.
- Sees Their Day: Views a live list of their upcoming booked appointments.
- Records Treatment: When a consultation is finished, the doctor enters the Diagnosis and Prescription, marking the appointment as "Completed".
- Views History: Instantly pulls up a patient's full past treatment history during a consultation.

Patient

- Self-Registration: Creates their own secure account.
- Easy Booking: Searches for doctors by specialty and date, and books an available time slot immediately.
- Appointment Control: Views their upcoming bookings and can cancel a scheduled appointment if needed, freeing up the slot for others.
- Accesses Records: After a visit, they can log in to view their own prescriptions and diagnoses.

Video Presentation

Drive Link:

https://drive.google.com/file/d/10Dw9fDS-_ghDCN0_67i7gKF5SYmbnhzE/view?usp=sharing