Hospital Management System

---

1. Student Details

Name: KHAN ISHAAN SAHIL

Roll Number:24F2002035
Email: 24f2002035@ds.study.iitm.ac.in

About Me:I am a BS Data Science student interested in backend development, UI/UX workflows, and modular application design. I enjoy building structured systems that solve practical problems.

2. Project Details

Project Title: Hospital Management System (HMS)

Problem Statement

To build a web application that manages hospital workflows such as patient details, doctor profiles, appointments, and role-based dashboards with secure authentication.

Approach

The project uses Flask with a modular blueprint structure.
Key components include:

- **Admin, Doctor, and Patient dashboards**
- **Appointment booking and management**
- **Patient medical history**
- **Doctor specialization list**
- **Secure login/session handling**

All pages are rendered using Jinja2 templates and Bootstrap UI.
Optional API endpoints, CSV export, and Chart.js analytics were not included as they were recommended but not mandatory.

---

3. AI/LLM Declaration

I used ChatGPT (GPT-5) for limited guidance during development.
This included:

- **Suggestions for UI layout improvements**
- **Help structuring SQLAlchemy relationships**
- **Minor assistance with database setup and naming consistency**

Total AI/LLM involvement: 12–15%.
All routing, logic, authentication, debugging, and full implementation were written manually.

---

4. Technologies and Frameworks Used

Technology	Purpose
Flask	Core backend framework
SQLite	Project database
SQLAlchemy	ORM for models and relationships
Jinja2	Template rendering
Bootstrap 5	UI and layout
Flask-Login	Authentication & session handling

Flask Blueprints	**Modular route structure**
Custom Validation	**Used instead of WTForms**

---

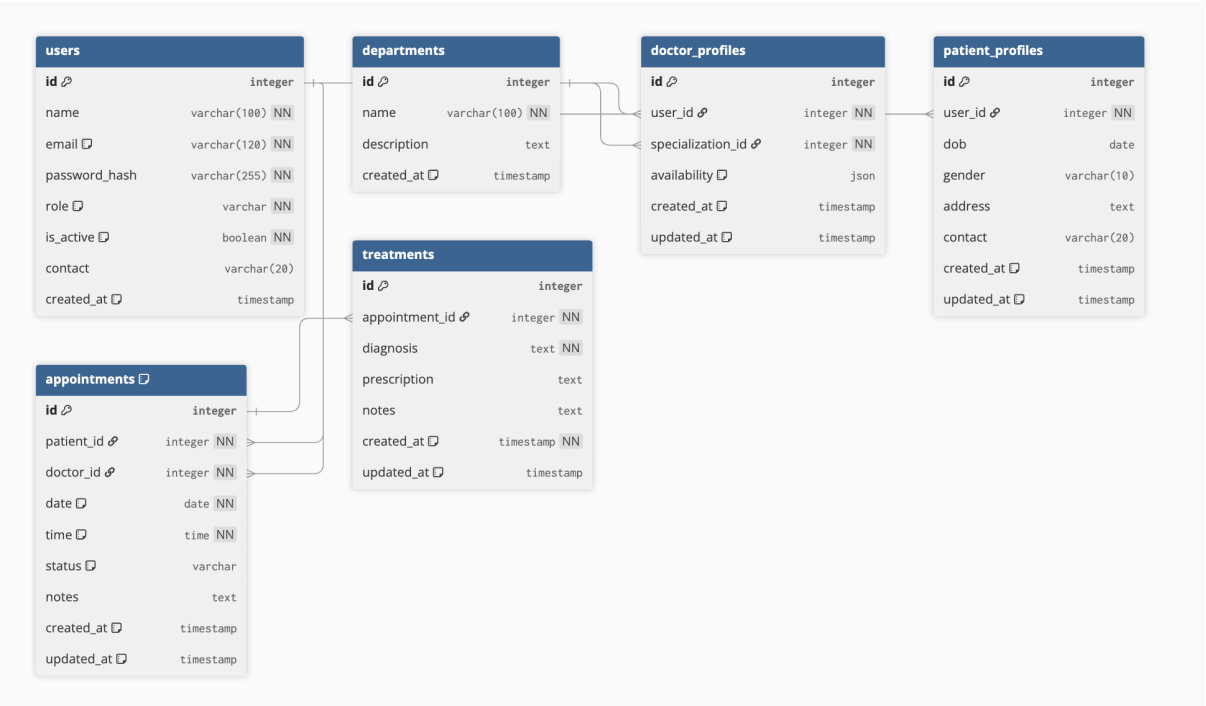
5. Database Schema / ER Diagram

Tables

User – id, name, email, password, role
Doctor – id, user_id, specialization, experience
Patient – id, user_id, age, gender, medical_history
Appointment – id, doctor_id, patient_id, date, time, status

Relationships

- **User → Doctor (1–1)**
- **User → Patient (1–1)**
- **Doctor → Appointments (1–Many)**
- **Patient → Appointments (1–Many)**



---

6. API Resource Endpoints (Not Implemented)

The project uses server-rendered HTML pages.
Recommended but not implemented API endpoints:

- **POST /api/login**
- **GET /api/doctors**
- **GET /api/patient/history**
- **POST /api/appointments/create**
- **POST /api/doctor/update**

Reason: These were optional in MAD-1 and not required for our workflow.

---

7. Architecture and Features

Project Structure

```
hospital-management-system/
|
├── run.py
├── setup_db.py
├── requirements.txt
├── README.md
|
└── app/
    ├── __init__.py
    ├── config.py
    ├── extensions.py
```

```
├── models.py
├── seed.py
│
├── routes/
│   ├── admin.py
│   ├── doctor.py
│   ├── patient.py
│   └── auth.py
│
├── static/
└── templates/
```

Implemented Features

- **Login, logout, and role-based dashboards**
- **Admin access to doctors, patients, and appointments**
- **Doctor profile and appointment management**
- **Patient history and appointment interface**
- **Secure session management**
- **Responsive Bootstrap UI**
- **Modular Flask Blueprints**

8. Video Presentation

DRIVE LINK- mad1