

# HOSPITAL MANAGEMENT SYSTEM

Hari Om

24f3005068

24f3005068@ds.study.iitm.ac.in

I am Hari Om currently studying in diploma level of the IITM BSDS course. I am passionate about learning new technologies and techniques that make my life easier and improve my capability as a creator.

## Description

This project is a multi-user Hospital Management System where users can register as Admin, Doctor, or Patient. Patients can search for doctors, book appointments, and view their medical history. Doctors can manage their availability, view assigned appointments, and add treatment details. Admins can manage doctors, patients, and view all appointments.

## Technologies used

### Database:

SQLite

It is used for the Database to store all the information about Users, Doctors, Patients, Appointments, Departments, and Treatments.

### Backend Technologies:

Flask

It is used for building the application and to start the app server.

## Flask-SQLAlchemy

It is used for Database management and CRUD operations.

## Flask-Login

It is used for user authentication and to keep the user logged in using the session and to log out.

## Werkzeug.security

It is used for securely hashing and verifying passwords during login and registration.

## **Frontend Technologies:**

### Jinja2

It is used for generating dynamic HTML templates using the data received from flask server.

### CSS

It is used to apply custom styling to some components.

### Bootstrap 5

It is used to make the frontend responsive and use predefined bootstrap components for forms, buttons, cards, and navigation.

## **DB Schema Design**

### 1. Users

- id: INTEGER [PRIMARY KEY] NOT NULL
- username: VARCHAR(80) NOT NULL [UNIQUE]
- email: VARCHAR(120) NOT NULL [UNIQUE]
- password\_hash: VARCHAR(255) NOT NULL
- role: VARCHAR(20) NOT NULL

- created\_at: DATETIME
- is\_active: BOOLEAN

## 2. Departments

- id: INTEGER [PRIMARY KEY] NOT NULL
- name: VARCHAR(100) NOT NULL [UNIQUE]
- description: TEXT
- created\_at: DATETIME

## 3. Doctors

- id: INTEGER [PRIMARY KEY] NOT NULL
- user\_id: INTEGER NOT NULL [FOREIGN KEY → Users.id]
- department\_id: INTEGER NOT NULL [FOREIGN KEY → Departments.id]
- full\_name: VARCHAR(100) NOT NULL
- specialization: VARCHAR(100) NOT NULL
- phone: VARCHAR(15)
- qualification: VARCHAR(200)
- experience\_years: INTEGER
- consultation\_fee: FLOAT
- created\_at: DATETIME

## 4. Patients

- id: INTEGER [PRIMARY KEY] NOT NULL

- user\_id: INTEGER NOT NULL [FOREIGN KEY → Users.id]
- full\_name: VARCHAR(100) NOT NULL
- date\_of\_birth: DATE
- gender: VARCHAR(10)
- phone: VARCHAR(15) NOT NULL
- address: TEXT
- blood\_group: VARCHAR(5)
- emergency\_contact: VARCHAR(15)
- medical\_history: TEXT
- created\_at: DATETIME

## 5. Doctor Availability

- id: INTEGER [PRIMARY KEY] NOT NULL
- doctor\_id: INTEGER NOT NULL [FOREIGN KEY → Doctors.id]
- date: DATE NOT NULL
- start\_time: TIME NOT NULL
- end\_time: TIME NOT NULL
- is\_available: BOOLEAN

## 6. Appointments

- id: INTEGER [PRIMARY KEY] NOT NULL
- patient\_id: INTEGER NOT NULL [FOREIGN KEY → Patients.id]
- doctor\_id: INTEGER NOT NULL [FOREIGN KEY → Doctors.id]

- appointment\_date: DATE NOT NULL
- appointment\_time: TIME NOT NULL
- status: VARCHAR(20) [DEFAULT: 'Booked']
- reason: TEXT
- created\_at: DATETIME
- updated\_at: DATETIME

## 7. Treatments

- id: INTEGER [PRIMARY KEY] NOT NULL
- appointment\_id: INTEGER NOT NULL [FOREIGN KEY → Appointments.id]
- diagnosis: TEXT NOT NULL
- prescription: TEXT
- notes: TEXT
- follow\_up\_date: DATE
- created\_at: DATETIME
- updated\_at: DATETIME

## Architecture and Features

/MAD1Project\_24f3005068 (This is root folder)

```
├── /routes
│   ├── admin.py
│   ├── doctor.py
│   ├── patient.py
│   ├── auth.py
│   └── api.py
├── /templates
│   ├── /admin
│   ├── /doctor
│   ├── /patient
│   ├── /auth
│   ├── base.html
│   └── index.html
├── /instance
│   └── hospital.db
├── app.py
├── models.py
├── extensions.py
├── utils.py
├── config.py
└── README.md
└── requirements.txt
```

## Features

### User Registration and Login

Flask-Login is used to keep the user logged in using sessions and Werkzeug.security is used to hash the password during registration to securely store them in the database. Users can register as Admin, Doctor, or Patient etc.

## Admin Dashboard and Management

The admin can view dashboard with statistics on total doctors, patients, and appointments. They can add new doctors with department assignments, edit doctor profiles, and manage patient information. Admins can view and manage all appointments in the system and search for specific patients or doctors.

## Doctor Dashboard and Appointment Management

Doctors can view their dashboard showing today's and week's appointments. They can mark appointments as completed and add diagnosis, prescriptions, and treatment notes. Doctors can manage their availability for the next 7 days and view patient medical history.

## Patient Dashboard and Appointment Booking

Patients can search for doctors by specialization and view available doctors with their profiles. They can book appointments with available doctors on specific dates and times. Patients can view their appointment history, cancel booked appointments, and access their complete medical history with diagnoses and prescriptions.

## RESTful API

API endpoints are available for all major operations including appointments, doctors, and patients management, allowing integration with other systems.

## Styling & Responsive Design

Bootstrap 5's predefined responsive elements are used to ensure better viewing experience across all device sizes with modern UI components.

## **VIDEO**

### [Project Video URL](#)

