App Development Project Report

App Dev Project Report

1. Student Details

Name: Pranatheesh S

Roll Number: 24F2007139
Email: 24f2007139@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 enjoy building meaningful applications that combine learning, analytics, and user experience.

---

2. Project Details

Project Title: Hospital Management System

Problem Statement:

To design and develop a centralized web-based application that replaces manual registers and disconnected software, enabling Admins, Doctors, and Patients to efficiently manage appointments, track medical history, and coordinate hospital operations without scheduling conflicts or data redundancy.

Approach:

The application will be built using **Flask** as the backend framework with a relational database to ensure data consistency. It utilizes a Role-Based Access Control (RBAC) architecture to provide secure, distinct interfaces for users: allowing patients to book appointments online, doctors to update digital medical records, and admins to manage staff and system data efficiently.

---

3. AI/LLM Declaration

This project utilized Google Gemini as an auxiliary tool to accelerate development. AI assistance was strictly limited to **10–15%** of the workflow, focusing primarily on drafting

SQLAlchemy model definitions, scaffolding frontend templates, and formatting API documentation. All core business logic, complex problem-solving, debugging, and system integration were executed manually to ensure architectural integrity and reliability.

---

4. Technologies and Frameworks Used

Technology / Library	Purpose
Flask	Core backend web framework
SQLAlchemy	Object Relational Mapper for SQLite database
Jinja2	Template engine for rendering dynamic HTML pages
Bootstrap 5	Frontend styling and responsive design
Flask-Login	User authentication and session management
SQLite	Lightweight local database for storing user data

---

5. Database Schema and ER Diagram

Tables:

1. **Department** — stores hospital department categories (`id`, `name`, `description`).
2. **Doctor** — stores medical staff profiles and credentials (`id`, `name`, `email`, `specialization`, `department_id`, `bio`, `experience`, `is_blacklisted`, `password_hash`).
3. **DoctorAvailability** — tracks specific dates and shift availability for doctors (`id`, `doctor_id`, `date`, `morning_available`, `evening_available`).
4. **Patient** — stores patient personal and login details (`id`, `name`, `email`, `password_hash`, `age`, `gender`, `is_blacklisted`).
5. **Appointment** — links patients and doctors for specific times (`id`, `patient_id`, `doctor_id`, `date`, `time`, `status`).

6. **Treatment** — stores medical diagnosis and prescriptions for a completed appointment (`id`, `appointment_id`, `visit_type`, `test_done`, `diagnosis`, `prescription`, `medicines`, `created_at`).

Relationships:

One-to-Many → Department → Doctor

- *(One department contains many doctors)*

One-to-Many → Doctor → Appointment

- *(One doctor handles many appointments)*

One-to-Many → Patient → Appointment

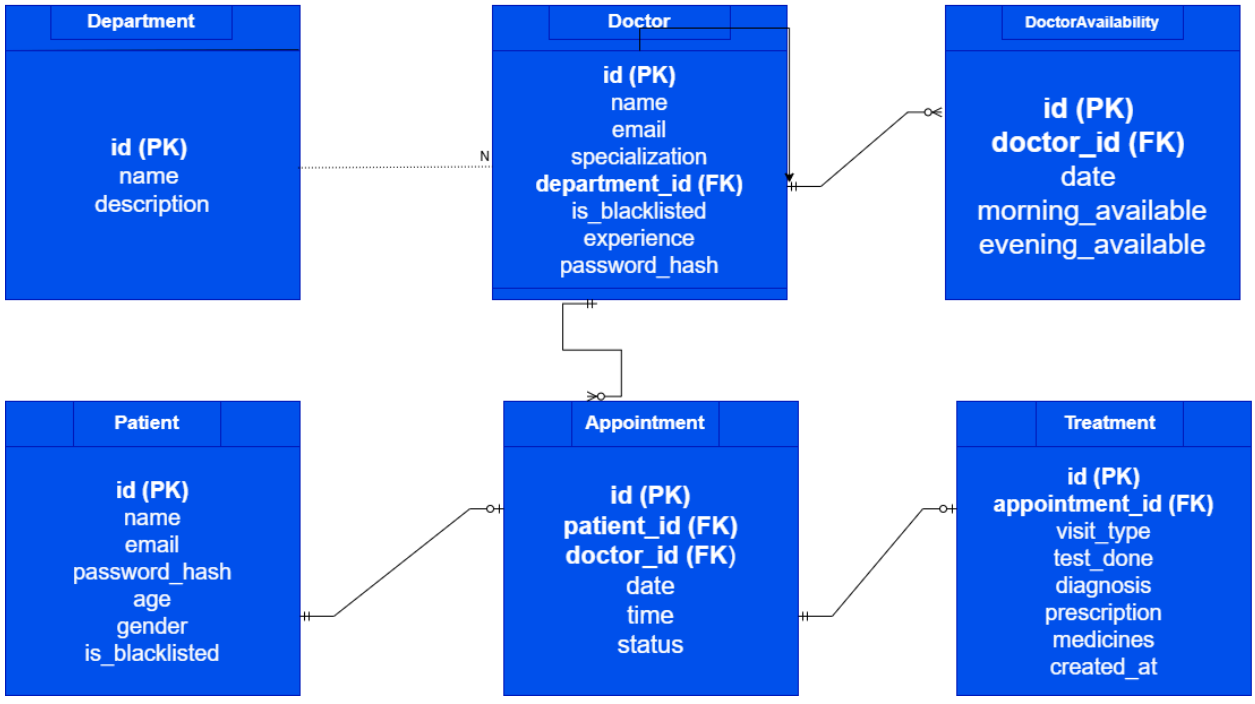
- *(One patient can book many appointments)*

One-to-Many → Doctor → DoctorAvailability

- *(One doctor has many availability records for different dates)*

One-to-One → Appointment → Treatment

(One appointment results in exactly one treatment record)



6. API Resource Endpoints

Endpoint	Method	Description
`/login`	POST	Authenticate patient and generate session
`/dashboard`	GET	Fetch patient dashboard with departments and upcoming appointments
`/book/<int:doctor_id>`	POST	Book a new appointment slot with a specific doctor
`/doctor/dashboard`	GET	Retrieve doctor's upcoming schedule and assigned patients
`/appointment/update/<int:appt_id>`	POST	Update medical treatment details, diagnosis, and prescription

---

7. Video Presentation

Drive Link:
https://drive.google.com/file/d/1WkV7TRENqdYdK5ezG4ztSiVkQjsj0qzU/view?usp=sharing