MAD II Project Report

1. Student Details

Name: Md Ehtesham Ansari
Roll Number: 23f2005532
Email: 23f2005532@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:

Hospitals need efficient systems to manage patients, doctors, appointments, and treatments. Currently, many hospitals use manual registers or disconnected software, which makes it difficult to manage records, avoid scheduling conflicts, and track patient history.

Approach:

The app was built using Flask as the backend framework with a modular structure, and Vue as the frontend framework. It allows users to register as patients and book appointments with any of the available doctors in different departments and then they can track the status and check their medical history with us, the doctors can check the booked appointments and address them and update the medical history. And the admin can manage everything in the application.

---

3. Objectives

Keep it simple but purposeful:
• Build a portal with separate roles: Admin, Doctor, Patient.
• Provide secure authentication and session handling.
• Allow patients to browse doctors and book 30-minute slots.

• Let doctors manage availability, appointments, and profiles.
• Give admins control over departments, users, and overall system data.

---

4. Technologies and Frameworks Used

Frontend: Vue 3 (Composition API), Pinia store, Vue Router

Backend: Flask (REST APIs), Flask-JWT-Extended

Database: SQLite

Design Pattern: REST API + SPA workflow

Authentication: JWT-based login

---

5. Database Schema / ER Diagram

Tables:

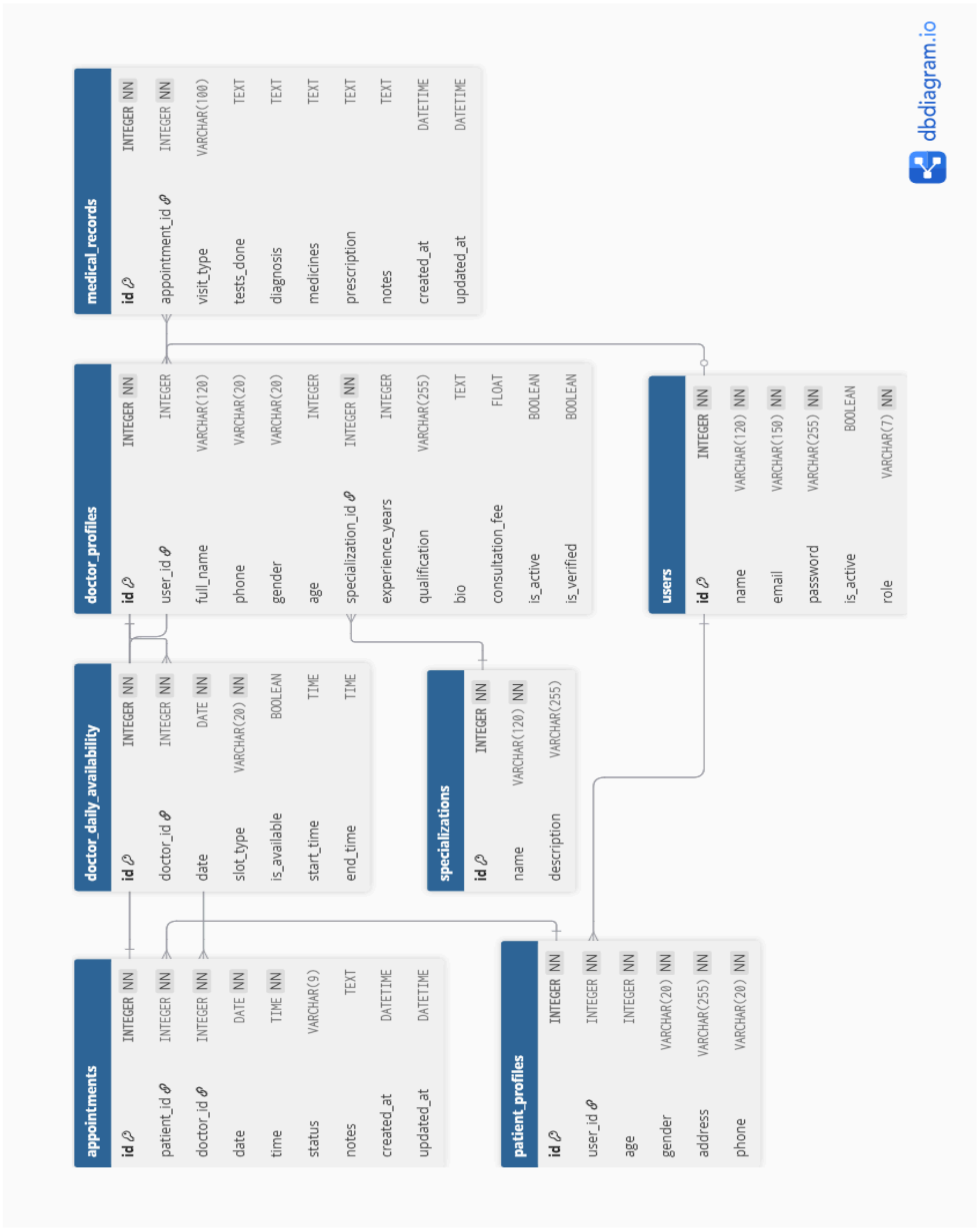
1. **User** — Stores the core authentication and identity information for every account in the system (patients, doctors, admins).
2. **PatientProfiles —** Holds the extended demographic and contact information for users who have the *patient* role.
3. **DoctorProfiles** — Contains professional and personal information for users with the *doctor* role.
4. **Specializations —** Defines medical departments or specialties that doctors belong to.
5. **DoctorDailyAvailability** — Represents the daily schedule blocks for each doctor.
6. **Appointments —** Stores confirmed bookings between patients and doctors.
7. **MedicalRecords** — Holds diagnostic and treatment data connected to completed appointments.

Relationships:

- One-to-Many → Users → PatientProfiles
- One-to-Many → Users → DoctorProfiles
- One-to-Many → Specializations → DoctorProfiles
- One-to-Many → DoctorProfiles → DoctorDailyAvailability
- One-to-Many → DoctorProfiles → Appointments

- One-to-Many → PatientProfiles → Appointments
- One-to-One / One-to-Many → Appointments → MedicalRecords

DB DIAGRAM:



medical_records		
id	INTEGER	NN
appointment_id	INTEGER	NN
visit_type	VARCHAR(100)	
tests_done	TEXT	
diagnosis	TEXT	
medicines	TEXT	
prescription	TEXT	
notes	TEXT	
created_at	DATETIME	
updated_at	DATETIME	

doctor_profiles		
id	INTEGER	NN
user_id	INTEGER	
full_name	VARCHAR(120)	
phone	VARCHAR(20)	
gender	VARCHAR(20)	
age	INTEGER	
specialization_id	INTEGER	NN
experience_years	INTEGER	
qualification	VARCHAR(255)	
bio	TEXT	
consultation_fee	FLOAT	
is_active	BOOLEAN	
is_verified	BOOLEAN	

users		
id	INTEGER	NN
name	VARCHAR(120)	NN
email	VARCHAR(150)	NN
password	VARCHAR(255)	NN
is_active	BOOLEAN	
role	VARCHAR(7)	NN

doctor_daily_availability		
id	INTEGER	NN
doctor_id	INTEGER	NN
date	DATE	NN
slot_type	VARCHAR(20)	NN
is_available	BOOLEAN	
start_time	TIME	
end_time	TIME	

specializations		
id	INTEGER	NN
name	VARCHAR(120)	NN
description	VARCHAR(255)	

appointments		
id	INTEGER	NN
patient_id	INTEGER	NN
doctor_id	INTEGER	NN
date	DATE	NN
time	TIME	NN
status	VARCHAR(9)	
notes	TEXT	
created_at	DATETIME	
updated_at	DATETIME	

patient_profiles		
id	INTEGER	NN
user_id	INTEGER	NN
age	INTEGER	NN
gender	VARCHAR(20)	NN
address	VARCHAR(255)	NN
phone	VARCHAR(20)	NN

6. API Resource Endpoints YAML API Definition File:

Endpoint	Method	Description
/auth/register	POST	Create a new user account.
/auth/login	POST	Authenticate user and return JWT token.
/admin/dashboard	GET	Fetch admin dashboard statistics.
/admin/tasks/<task_id>	GET	Retrieve async task logs/results.
/admin/specializations	GET / POST	Fetch or create medical specializations.
/admin/specializations/<spec_id>	GET / PUT / DELETE	View, update, or delete specialization.
/admin/doctors	GET	Retrieve list of all doctors.
/admin/doctors/<doc_id>	GET / PUT	View or update doctor profile.
/admin/doctors/<doc_id>/verify	PUT	Verify a doctor's credentials.
/admin/doctors/<doc_id>/toggle-status	PUT	Activate or deactivate doctor account.
/admin/doctors/<doc_id>/availability	GET / POST	View or create doctor availability slots.
/admin/doctors/<doc_id>/availability/<slot_id>	GET / PUT / DELETE	Manage specific availability slot.
/admin/appointments	GET	Fetch all appointments.

/admin/appointments/<appt_id>	GET	View appointment details.
/admin/appointments/<appt_id>/status	PUT	Update appointment status.
/admin/patients	GET	Retrieve list of all patients.
/admin/patients/<patient_id>	GET / PUT	View or update patient details.
/admin/patients/<patient_id>/toggle	PUT	Activate or deactivate patient account.
/admin/patients/<patient_id>/records	GET	Fetch patient medical history.
/doctor/dashboard	GET	Fetch doctor’s dashboard data.
/doctor/profile	GET / PUT	View or update own profile.
/doctor/availability	GET / POST	Get or create availability slots.
/doctor/availability/<slot_id>	GET / PUT / DELETE	Manage a specific availability slot.
/doctor/appointments	GET	Retrieve doctor’s appointments.
/doctor/appointments/<appointment_id>	GET / PUT	View or update appointment details.
/doctor/patients	GET	View doctor’s patient list.
/doctor/records	GET / POST	Fetch or create medical records.
/doctor/records/<appointment_id>	GET	View records for a specific appointment.
/doctor/patients/<patient_id>/records	GET	Fetch history for a specific patient.

/patient/dashboard	GET	Fetch patient dashboard.
/patient/profile	GET / PUT	View or update patient profile.
/patient/records	GET	View patient medical records.
/patient/export-treatments	GET	Export treatment history file.
/patient/doctors	GET	Retrieve list of available doctors.
/patient/doctors/<doctor_id>	GET	View doctor profile.
/patient/doctors/<doctor_id>/availability	GET	View doctor's available slots.
/patient/appointments	GET / POST	View or book appointments.
/patient/appointments/<appointment_id>	GET	View appointment details.
/patient/appointments/<appointment_id>/cancel	PUT	Cancel appointment.

---

7. Video Presentation

Drive Link:
https://drive.google.com/file/d/1W-jzAwE8S2FXW45oeS6rwz4PTb5GntYd/view?usp=sharing
(Accessible to all with "View" permission.)
