Modern Application Development Project Report

1. Student Details

Name: Krishna Vallabha Goswami
Roll Number: 23f3002697
Email: 23f3002697@ds.study.iitm.ac.in
About Me: Data Science undergrad passionate about machine learning, Bayesian inference and quantitative finance. Focused on building systems that think, adapt and solve real world problems.

---

2. Project Details

Project Title: Hospital Management System - V2

Problem Statement:

Manual appointment scheduling leads to double bookings and scheduling conflicts. Patients struggle to find doctors, book appointments, and access medical history. Administrative overhead increases due to fragmented systems and lack of centralized management.

Approach:

Multi-role web application (Admin, Doctor, Patient) using Flask backend and Vue.js frontend. Real-time appointment booking with conflict prevention and treatment history tracking. Performance optimization through Redis caching and database indexing, with JWT-based security and role-based access control.

---

3. AI/LLM Declaration

I have used ChatGPT to assist in writing Charts / Data Visualization code using Chart.js. The extent of AI/LLM usage is around 2%, limited to code suggestions and code refinement. All final implementation logic, debugging, and integration were done manually.

---

4. Technologies and Frameworks Used

Backend (Python)

- Flask — REST API framework
- Flask-SQLAlchemy — Database ORM
- Flask-JWT-Extended — Authentication tokens
- Flask-CORS — Cross-origin requests
- Redis — Caching layer
- Celery — Background tasks
- ReportLab — PDF generation
- Twilio — SMS notifications

Frontend (JavaScript)

- Vue.js 3 — UI framework

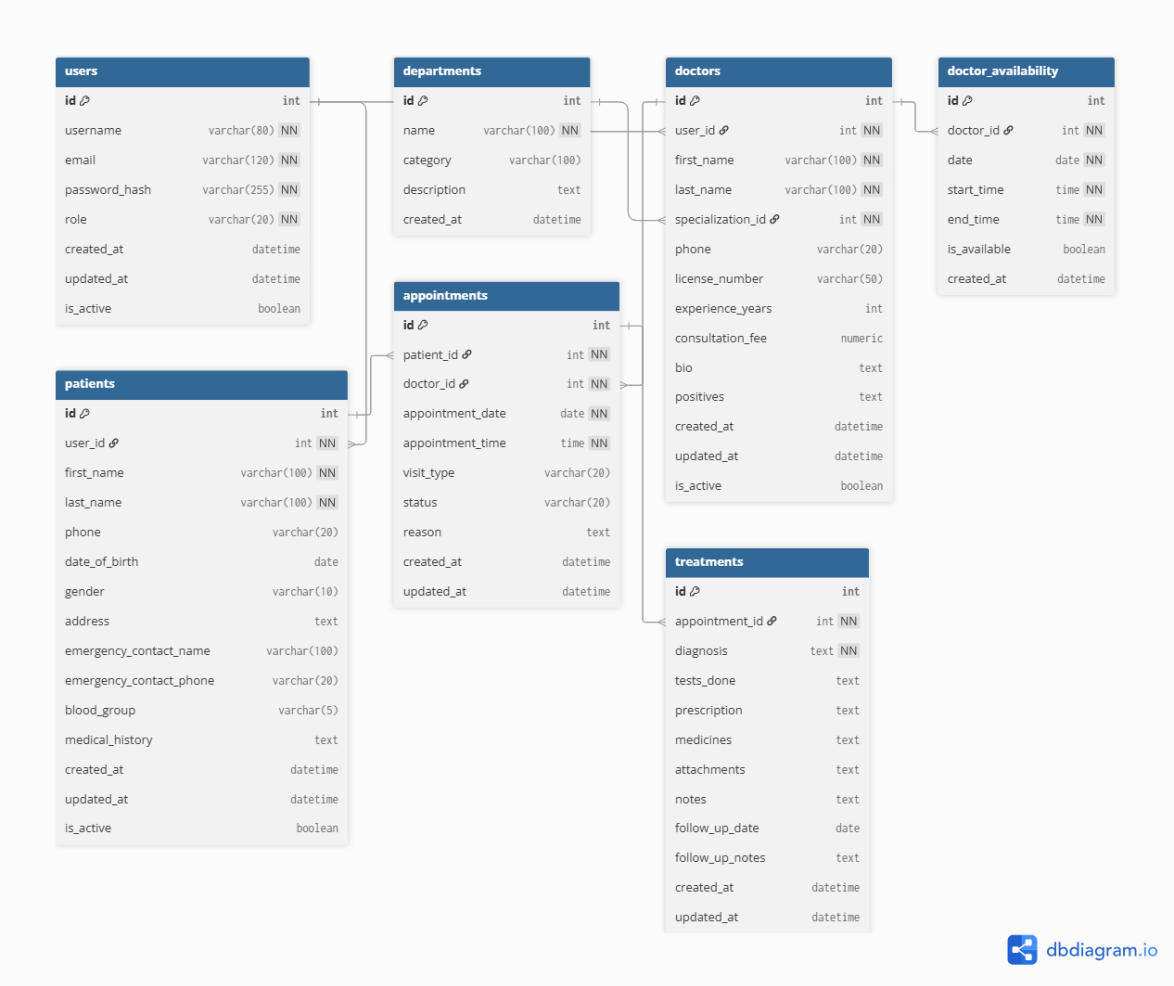
- Vite — Build tool
- Bootstrap 5 — UI styling
- Chart.js — Analytics graphs
- Axios — HTTP client

Database

- SQLite — Database storage

---

5. Database Schema / ER Diagram



Relationships

- **User → Doctor:** One-to-One
  Each User (with role = doctor) has one Doctor profile.
- **User → Patient:** One-to-One
  Each User (with role = patient) has one Patient profile.
- **Department → Doctor:** One-to-Many
  A department can have multiple doctors.
- **Doctor → Doctor Availability:** One-to-Many
  A doctor can have many availability slots.
- **Patient → Appointment:** One-to-Many
  A patient can book many appointments.
- **Doctor → Appointment:** One-to-Many
  A doctor can have many appointments.
- **Appointment → Treatment:** One-to-One
  Each appointment has exactly one treatment record.

6. API Resource Endpoints

1. Authentication

Endpoint	Method	Purpose
/api/auth/register	POST	Create patient account
/api/auth/login	POST	Login (Admin/Doctor/Patient)

2. Core Dashboards

Endpoint	Method	Purpose
/api/dashboard/admin	GET	Admin overview
/api/dashboard/doctor	GET	Doctor overview
/api/dashboard/patient	GET	Patient overview

3. Appointments (Central Feature)

Endpoint	Method	Purpose
/api/patient/appointments	POST	Book appointment
/api/doctor/appointments	GET	Doctor's scheduled appointments
/api/doctor/appointments/{id}/complete	PUT	Mark appointment completed
/api/patient/appointments/{id}	GET	View appointment & treatment

4. Treatment Management

Endpoint	Method	Purpose
/api/doctor/appointments/{id}/treatment	POST	Add treatment details
/api/doctor/appointments/{id}/treatment	PUT	Update treatment

5. Doctor & Patient Access

Endpoint	Method	Purpose
/api/patient/doctors	GET	Search doctors
/api/doctor/patients/{id}	GET	View patient medical history

6. Key Admin Operations

Endpoint	Method	Purpose
/api/admin/doctors	POST	Add doctor
/api/admin/patients	GET	View all patients
/api/admin/departments	POST	Add specialization

---

7. Architecture and Features

Architecture

- **Three-tier:** Vue.js frontend → Flask REST API → SQLite database

- **Key components:** 7 route modules, 7 database models, JWT auth, Redis caching, Celery background jobs
- **Communication:** RESTful API with JSON, CORS-enabled

Implemented features

1. Authentication & Authorization - Multi-role login, JWT tokens
2. Admin Features - Dashboard, CRUD operations, analytics
3. Doctor Features - Appointment management, treatment records, availability
4. Patient Features - Registration, booking, history access
5. Appointment Management - Conflict prevention, status tracking
6. History & Reporting - Complete history with filters, exports
7. Performance - Redis caching, database indexing
8. PWA Support - Offline functionality, installable
9. Security - Password hashing, role-based access
10. Data Export - PDF/CSV generation

---

8. Video Presentation

Drive Link:
https://drive.google.com/drive/folders/1FzOCXhjwjOcclmHxpE36Lg64lGjaYTgC?usp=sharing