Project Proposal: MediTrack – Patient Health Record System

Submitted By: Manjeet
Course: B.Tech CS & AI
Date: 30 October 2025

---

1. Project Title

MediTrack – Patient Health Record System

---

2. Problem Statement

In today's healthcare system, managing patient data can be difficult and time-consuming. Many hospitals and clinics still depend on manual methods or multiple disconnected tools to store and share health information. This often leads to missing reports, delayed communication, and inefficient record handling.

MediTrack aims to solve this problem by providing a single, secure online platform where patients, doctors, and administrators can access and update medical records anytime. It will help in keeping health data organized, easy to track, and accessible to the right people when needed.

Additionally, the system will dynamically fetch and display real-time data from the backend, ensuring that users always view the latest records without needing to refresh pages manually.

---

3. System Architecture

Structure:
Frontend → Backend (API) → Database

Stack:

- **Frontend:** React.js with React Router for smooth page navigation and dynamic data rendering

- **Backend:** Node.js + Express.js to create RESTful APIs
- **Database:** MongoDB (non-relational)
- **Authentication:** JWT-based login/signup with role-based access (Patient, Doctor, Admin)

Dynamic Data Flow:

- The frontend dynamically fetches and updates data using **Fetch API**.
- **React Query** or custom hooks will manage API calls, caching, and background updates.
- When users perform any operation (add/update/delete), the UI automatically re-fetches data from the backend to display updated results instantly.

Example Data Flow:

- When a doctor logs in, the dashboard dynamically fetches patients and appointments using `/api/patients` and `/api/appointments`.
- When a patient uploads a report, the upload triggers an API call to `/api/reports/upload`, and the updated list is fetched automatically.
- Admin dashboards display live, searchable, and filterable tables using dynamic API queries.

Hosting:

- **Frontend** → Vercel
- **Backend** → Render
- **Database** → MongoDB Atlas

---

4. Key Features

Category	Features

Authentication & Authorization	Secure signup, login, and logout with roles for Patients, Doctors, and Admins
Dynamic Data Rendering	Pages dynamically fetch and display live data from the backend using Axios or Fetch API
CRUD Operations	Add, view, update, and delete patient records, appointments, and prescriptions
Search, Filter, Sort & Pagination	Real-time search, dynamic filters, and paginated results fetched through APIs
Appointment Management	Book, update, or cancel appointments and track upcoming visits dynamically
Health Report Upload	Patients can upload lab results, test reports, and prescriptions that update the dashboard instantly
Data Visualization	Charts showing trends such as blood pressure or sugar levels, powered by dynamic API data
Frontend Routing	Separate pages for Home, Login, Dashboard, Patient Records, Appointments, and Profile
Real-Time Updates	Components re-fetch updated data automatically after any CRUD operation
Hosting	Both frontend and backend will be deployed online for easy user access

---

5. Tech Stack

Layer	Technologies
Frontend	React.js, React Router
Backend	Node.js, Express.js
Database	MongoDB (MongoDB Atlas)
Authentication	JSON Web Token (JWT)
Hosting	Vercel (Frontend), Render (Backend), MongoDB Atlas (Database)

---

6. API Overview

Endpoint	Method	Description	Access
`/api/auth/signup`	POST	Register a new user (Patient/Doctor/Admin)	Public
`/api/auth/login`	POST	Login and generate authentication token	Public
`/api/patients`	GET	Fetch all patient records dynamically from the database	Authenticated
`/api/patients/:id`	PUT	Update patient details	Doctor/Admin
`/api/patients/:id`	DELETE	Delete a patient record	Admin only
`/api/appointments`	POST	Create a new appointment	Authenticated
`/api/appointments`	GET	Retrieve all appointments dynamically	Authenticated
`/api/reports/upload`	POST	Upload medical reports	Patient only
/api/patients/:id	GET	To get all the details	Doctor/Admin