Phase 1: Foundational Development (User & Core Entities)

This phase is the absolute starting point, as it builds the core backend logic for all users and the main entities (doctors and clinics).

- **Functions:**
    - **User Management:** Build API endpoints for user registration, secure login, password resets, and email confirmation.
    - **Core Entities:** Create the backend services and API endpoints to manage doctors, patients, and clinics. This includes creating, retrieving, updating, and deleting these records.
    - **Associations:** Implement the logic to create relationships between doctors and clinics (doctor_clinic_associations) and to assign user roles (user_role).
- **Workflow:**
    - The team will first set up the backend framework and database.
    - They will then build the user authentication and authorization services.
    - Finally, they'll create CRUD operations for all the core tables: users, patients, doctors, clinics, and their association tables.
- **Output:** A functional backend with secure user authentication and API endpoints to manage patient, doctor, and clinic profiles.
- **Time Required:** Approximately 2-3 weeks.

Phase 2: Booking and Scheduling

This phase focuses on the core functionality of the platform: making appointments.

- **Functions:**
    - **Schedule Management:** Create API endpoints for doctors to add, edit, or remove their available time slots.
    - **Appointment Booking:** Implement the logic for patients to search for available slots, select one, and create a new appointment record. The backend must handle a "booked" status to prevent double-booking.
    - **Appointment Management:** Build endpoints for patients to view their upcoming and past appointments and to cancel a booking.
    - **Reviews & Ratings:** Create the API for patients to submit reviews and for the backend to calculate and update a doctor's average_rating.
- **Workflow:**
    - The team will begin by designing the schedule logic to handle time slot availability.

    - They will then create the appointments table and build the booking API, ensuring it links patients and doctors to a specific schedule slot.
    - Finally, they will implement the API for patients to view their own bookings and the cancellation logic.
- **Output:** A fully functional booking system that allows patients to find and reserve time slots and for doctors to manage their schedules.
- **Time Required:** Approximately 3-4 weeks.

Phase 3: Medical and Financial Services

This phase is dedicated to the more sensitive and complex parts of the system: medical history and the points-based loyalty program.

- **Functions:**
    - **Medical Records:** Build secure API endpoints for doctors to add new entries to a patient's history, including diagnoses, treatments, and prescriptions.
    - **Points System:** Implement the logic to award points to a patient's account after a completed appointment. Create the logic for a patient to check their points balance and to redeem points for a free booking.
    - **Financials:** Develop the backend to handle the consultation_fee and transaction_fee for each appointment and to update the payment status.
- **Workflow:**
    - The team will start with the patient_history_entries table, ensuring strict security protocols.
    - Finally, they will integrate this logic into the appointments API to handle payment and points redemption.
- **Output:** A secure system for managing patient medical records and a fully functional points-based loyalty program.
- **Time Required:** Approximately 4-5 weeks.

Phase 4: Administrative & Reporting Services

The final development phase focuses on the tools that manage the entire platform.

- **Functions:**
  - **Admin Tools:** Build secure APIs for administrators and clinic admins to manage user accounts (confirming new doctors, activating/deactivating accounts) and clinic profiles.
  - **Reporting:** Create backend services that can generate aggregated data. This involves writing complex database queries to produce reports on total bookings, revenue, and overall user activity.
- **Workflow:**
  - The team will develop the logic for the various admin dashboards, which will require building secure, private API endpoints that only admin roles can access.
  - They will then focus on the reporting aspect, creating services that can pull and aggregate data from all the other tables.
- **Output:** The full backend for the admin dashboards, providing robust tools for managing the entire platform and generating key business reports.
- **Time Required:** Approximately 2-3 weeks.