Build a Simple Patient Management System

- This task covers back-end, front-end, and database skills.
- Do not use any AI
- Try to use third-party packages only if necessary.
- Consider that you are building an enterprise-level application while designing the system, so include your advanced skills but keep the code simple.
- Submission deadline: **Sunday 07 July, 2024 – 12:00 AM (PST)**
- Send a GitHub public repository link with your name on WhatsApp +1 (646) 493-7098

Requirements

1. User Authentication
    - Implement user authentication (login and registration) using Laravel's built-in authentication system.
    - Users should be able to reset their passwords.
2. Patient Records
    - Create a system to manage patient records with the following fields:
        - Name
        - Date of Birth
        - Gender
        - Country & City
        - Contact Information
        - Medical History
        - Current Medications
3. Frontend Interface
    - Use Laravel Blade templates for the front end, try to use livewire AJAX, AXIOS, or whatever is convenient for you.
    - Implement a responsive design using any CSS framework like Bootstrap or Tailwind CSS *(you can also use prebuilt templates)*.
    - Front-end should be attractive and user-friendly.
    - Create forms for adding, editing, and viewing patient records.
4. Database
    - Use Laravel migrations to create the necessary database tables and relations
    - Seed the database with some initial data for testing.
5. Patient List and Search
    - Implement a paginated list of patients.
    - Include a search functionality to filter patients.
    - List and search should be server-side rendering (can use data tables or any alternative)
6. Validation and Error Handling
    - Add server-side validation for all forms.
    - Display appropriate error messages on the front end.
7. Security
    - Ensure proper security measures like input sanitization and CSRF protection are in place.
    - User roles & permission (you can use build-in or any third-party package)
8. Code Quality
    - Follow Laravel best practices and coding standards.
    - Use meaningful variable names and write clean, readable code.

   - Include comments where necessary to explain the code.

9. Deliverables
   - Source Code
     - A link to a GitHub public repository containing the working source code.
   - Database Schema
     - A SQL file or migration files for setting up the database.
   - Documentation
     - A brief document explaining the design and implementation of the system.

Evaluation Criteria

1. Functionality
   - Completeness of the task as per the requirements.
   - Correctness and efficiency of the implemented features.
2. Code Quality
   - Adherence to Laravel coding standards and best practices.
   - Cleanliness and readability of the code.
3. User Interface
   - Usability and responsiveness of the frontend.
   - Visual appeal and use of CSS framework.
4. Database Design
   - Proper use of migrations and seeding.
   - Efficient schema design and data handling.
5. Security
   - Implementation of security measures like input validation and CSRF protection.