Full Stack Development with MERN

Project Documentation

1. Introduction

- **Project Title:** Book a Doctor
- **Team Members:**

  Swathie P (Project Manager)

  V Kavya Sahithi (Frontend Developer)

  Jaiganesh S (Backend Developer)

  Maturi Nikhitha (Database Administrator)

2. Project Overview

- **Purpose:** The purpose of the "Book a Doctor" project is to provide users with an online platform to book appointments with doctors, manage their health records, and get reminders for their appointments.
- **Features:**

  · User registration and login
  · Doctor search and profile viewing
  · Appointment booking and management
  · User profile management
  · Notifications and reminders for appointments

3. Architecture

- **Frontend:** The frontend is built using React.js. It follows a component-based architecture, utilizing Redux for state management and Material-UI for the user interface components. React Router is used for client-side routing.
- **Backend:** The backend is developed using Node.js and Express.js. It provides RESTful APIs for the frontend to interact with. The backend handles user authentication, data validation, and business logic.
- **Database:** The database schema is designed using MongoDB. Mongoose is used as the ODM to define schemas and interact with the MongoDB database. The main collections are users, posts, and comments.

4. Setup Instructions

- **Prerequisites:**

  · Node.js (v14 or later)
  · MongoDB (v4.4 or later)
  · Git

- **Installation:**

  Clone the repository:

  ```
  git clone https://github.com/yourusername/book-a-doctor.git
  cd book-a-doctor
  ```

  Install dependencies:
  For frontend:

  ```
  cd client
  npm install
  ```

  For backend:

  ```
  cd server
  npm install
  ```

  Set up environment variables:
  Create a `.env` file in the `server` directory and add the following:

  ```
  MONGO_URI=mongodb+srv://<bookadoc >:<bookadoctor
  >@cluster0.mongodb.net/book_a_doctor?retryWrites=true&w=majority
  JWT_SECRET=your_jwt_secret
  ```

5. Folder Structure

- **Client:**

  `src/`
  `components/`: Reusable UI components

  `pages/`: Different pages of the application

  `redux/`: Redux store and slices

  `App.js`: Main application component

  `index.js`: Entry point

- **Server:**

  `controllers/`: Controller functions for handling requests
  `models/`: Mongoose schemas and models
  `routes/`: API routes
  `middlewares/`: Custom middleware
  `server.js`: Main server file

6. Running the Application

- Commands to start the frontend and backend servers locally.
  - **Frontend:**

    ```
    cd client

    npm start
    ```

  - **Backend:**

    ```
    cd server

    npm start
    ```

7. API Documentation

Endpoints:

- **User Management**:
  - `POST /api/users/register`: Register a new user
  - `POST /api/users/login`: Authenticate a user
- **Post Management**:
  - `GET /api/posts`: Get all posts
  - `POST /api/posts`: Create a new post
- **Comment Management**:
  - `GET /api/comments`: Get all comments
  - `POST /api/comments`: Create a new comment

Example Request/Response:

- `POST /api/users/login`:
  - Request:

```
{
  "email": "user@example.com",
  "password": "password123"
}
```

- Response:

```
{
  "token": "jwt_token",
  "user": {
    "_id": "user_id",
    "name": "User Name",
    "email": "user@example.com"
  }
}
```

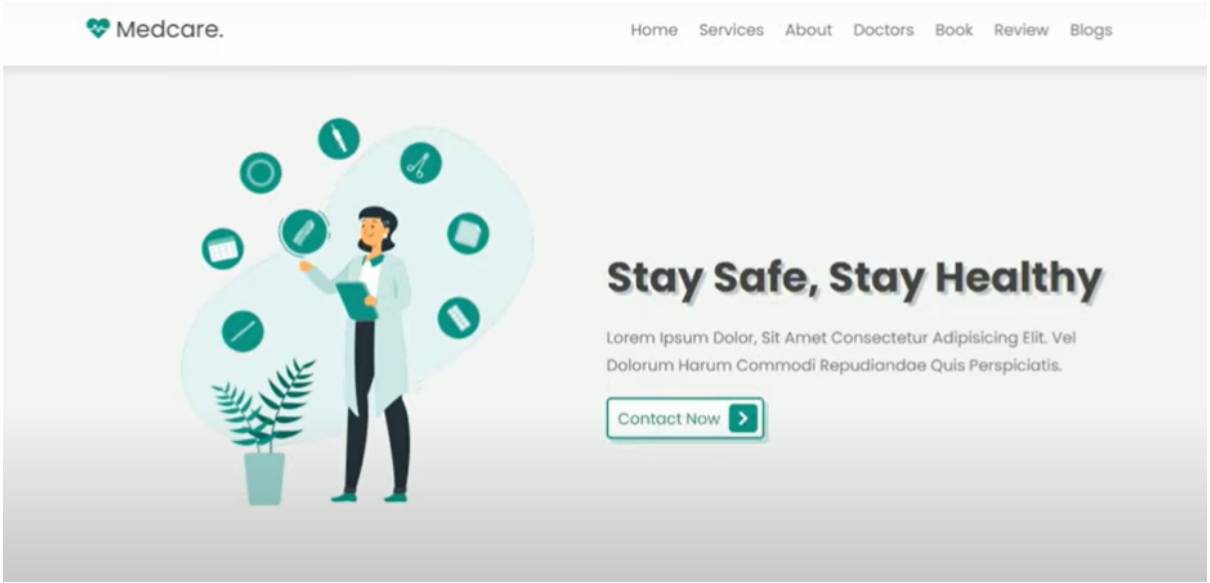
8. Authentication

Method: JWT (JSON Web Token)

- Upon successful login, a JWT token is issued and must be included in the `Authorization` header for protected routes.
- Example:

```
Authorization: Bearer jwt_token
```

9. User Interface



10. Testing

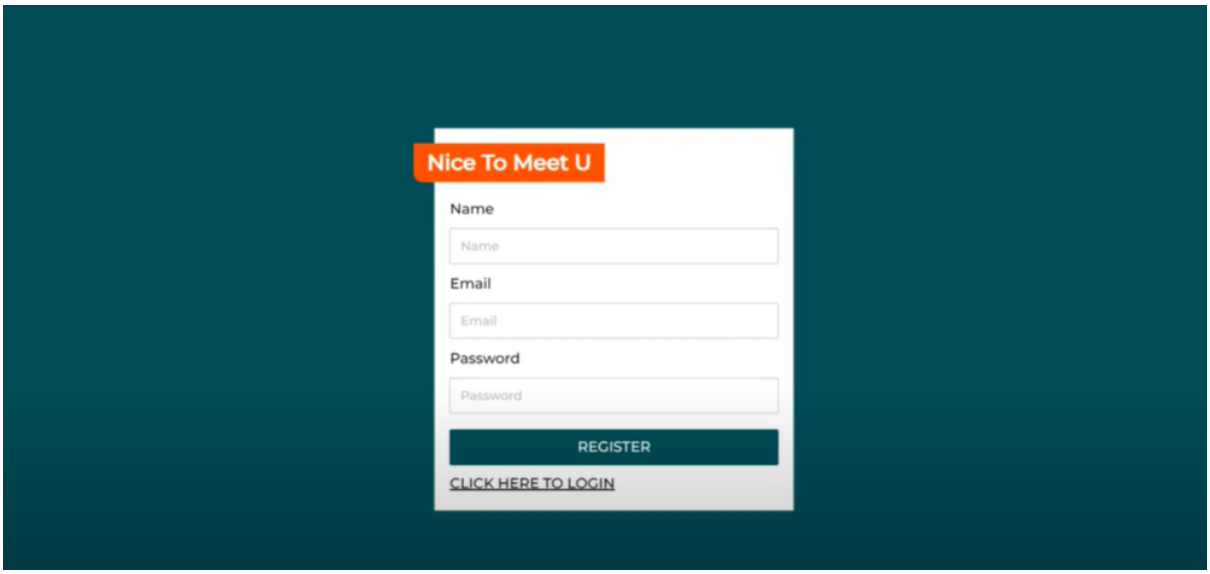
· **Strategy**:

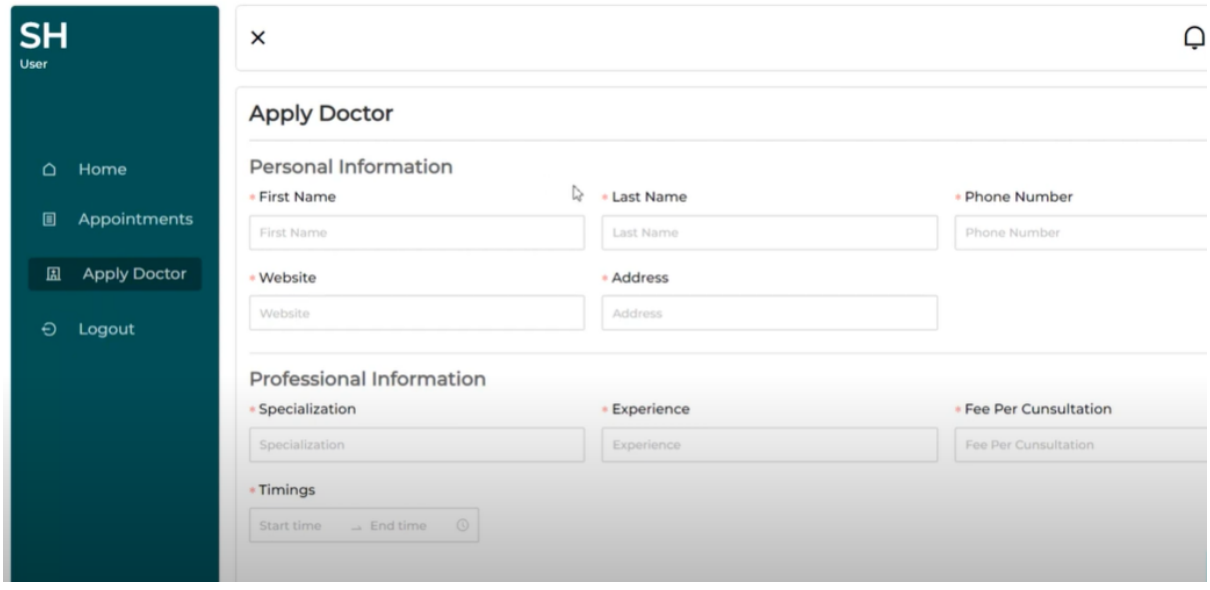
- Unit tests for individual components and functions

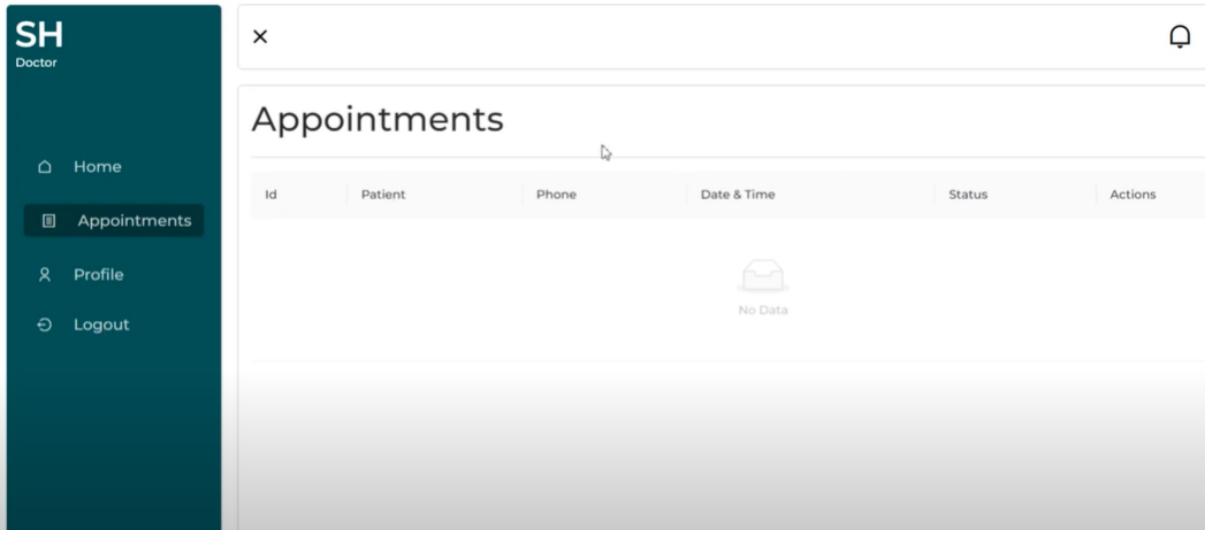
- Integration tests for API endpoints
- End-to-end tests for user flows

· **Tools**: Jest, Mocha, Chai, Supertest

11. Screenshots or Demo







12. Known Issues

- Slow loading times on the dashboard page.
- Minor UI glitches on mobile devices.

13. Future Enhancements

- Implementing video consultation feature.
- Adding more detailed analytics for users and doctors.
- Improving the UI/UX for mobile responsiveness.
- Enhancing the search functionality with more filters.