Final Document for Docspot: Seamless Appointment Booking for Health

Date: 22-June-2025

Team ID: LTVIP2025TMID30950

Project Name: Docspot: Seamless Appointment Booking for Health

Maximum Marks:

1. Project Overview

Docspot is a comprehensive application designed to streamline the appointment booking process for patients and healthcare providers. The application aims to enhance user experience by providing features such as real-time appointment scheduling, automated notifications, and telehealth options.

2. Repository Structure

The project repository contains the following key files and directories:
.git/: Contains all the version control information.
client/: The frontend application built using React.js.
public/: Contains static files such as images and the main HTML file.
src/: Contains the source code for the React application, including components, hooks, and Redux slices.
server/: The backend application built using Node.js and Express.js.
controllers/: Contains the logic for handling requests and responses.
models/: Contains the database models for users, doctors, and appointments.
routes/: Defines the API endpoints for the application.
utils/: Contains utility functions and error handling.
README.md: Documentation for the project, including setup instructions and usage.
package.json: Lists the dependencies and scripts for both the client and server applications.

3. Technology Stack

Frontend:
Framework: React.js
State Management: Redux

Styling: CSS, Bootstrap, or Material-UI

Backend:
Framework: Node.js with Express.js
Database: MongoDB for data storage
Authentication: JSON Web Tokens (JWT)

Deployment:
Cloud Hosting: AWS or Heroku
Containerization: Docker for consistent deployment

4. **Solution Requirements**

Functional Requirements:

- User Registration & Login: Users can register and log in using email and password.
- Appointment Booking: Users can view available doctors and book appointments.
- Appointment Management: Users can view, edit, and cancel their appointments.
- Real-Time Availability: The system displays real-time availability of doctors.
- Automated Notifications: Users receive SMS and email notifications for upcoming appointments.
- Telehealth Options: Users can book telehealth appointments.
- User Feedback System: Users can provide feedback on their healthcare providers.
- Admin Dashboard: Admins can manage user accounts and view appointment statistics.

Non-Functional Requirements:

- Usability: The application should have an intuitive interface.
- Security: User data must be secured using encryption.
- Reliability: The application should ensure consistent performance.
- Performance: The application should respond to user interactions within 2 seconds.
- Availability: The system should maintain an uptime of at least 99.9%.
- Scalability: The application should handle increasing user loads efficiently.

5. **User Acceptance Testing (UAT)**

Testing Objectives:

- Validate that the application meets specified requirements.
- Ensure all functionalities work as intended.
- Identify defects before the application goes live.

Key Test Cases:

- Verify user registration with valid credentials.
- Validate login with incorrect email or password.
- Verify appointment booking functionality.
- Validate appointment management features.
- Check real-time availability of doctors.
- Verify automated notifications.
- Validate telehealth consultation options.
- Check user feedback submission.

Bug Tracking:

Document any issues found during testing, including severity and status.