Full Stack Development with MERN

Project Documentation format

1. INTRODUCTION

1.1 Project Overview

The SmartDoc Appointment System is a web-based healthcare appointment booking platform developed using the MERN stack (MongoDB, Express.js, React.js, Node.js). The system enables patients to book doctor appointments online, allows doctors to manage schedules, and provides administrators with system monitoring capabilities.

1.2 Purpose

The purpose of this project is to digitize the traditional manual appointment booking system and provide a secure, efficient, and user-friendly online solution to reduce waiting times and improve healthcare service management.

2. IDEATION PHASE

2.1 Problem Statement

Patients experience long waiting hours, difficulty checking doctor availability, and inefficiencies in manual appointment booking systems. There is no centralized digital system to manage appointments effectively.

2.2 Empathy Map Canvas

The empathy map identifies:

- Users: Patients, Doctors, Administrators
- Pains: Long waiting time, uncertainty, manual errors
- Gains: Quick booking, reminders, organized scheduling

2.3 Brainstorming

Different solutions were considered:

- Manual digitization system
- Mobile-only application
- Web-based MERN system (Selected Solution)

The web-based MERN system was chosen due to scalability and flexibility.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Patient Journey:

1. Register/Login
2. Search Doctor
3. Check Availability
4. Book Appointment
5. Receive Confirmation
6. Visit Doctor

3.2 Solution Requirements

Functional Requirements

- User Registration & Login
- Doctor Registration & Approval
- Search Doctor
- Book/Cancel/Reschedule Appointment
- Dashboard Management
- Email Notifications

Non-Functional Requirements

- Security (JWT Authentication)
- Performance (<3 sec response time)
- Scalability (Cloud Database)
- Availability (24/7 system uptime)

3.3 Data Flow Diagram

The DFD consists of:

- External Entities: Patient, Doctor, Admin
- Processes: Authentication, Appointment Booking
- Data Store: MongoDB Atlas

3.4 Technology Stack

- Frontend: React.js
- Backend: Node.js, Express.js
- Database: MongoDB Atlas
- Authentication: JWT, Bcrypt
- Email Service: Nodemailer

4. PROJECT DESIGN

4.1 Problem – Solution Fit

The system directly addresses:

- Manual booking inefficiencies
- Scheduling conflicts
- Communication gaps

By providing:

- Centralized digital platform
- Real-time availability updates
- Automated notifications

4.2 Proposed Solution

SmartDoc provides:

- Role-based dashboards
- Secure login system
- Real-time booking management
- Admin monitoring

4.3 Solution Architecture

The system follows a 3-tier architecture:

1. Presentation Layer (React Frontend)
2. Application Layer (Node.js + Express Backend)
3. Database Layer (MongoDB Atlas Cloud)

Communication occurs via REST APIs.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Total Story Points: 45
Number of Sprints: 3
Velocity: 15 Story Points per Sprint

Sprint Distribution:

- Sprint 1: Authentication & Doctor Approval
- Sprint 2: Appointment Management
- Sprint 3: Dashboard & Notifications

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Functional Testing

- Registration validation
- Login authentication
- Appointment booking
- Admin approval
- Dashboard display

All major test cases passed successfully.

6.2 Performance Testing

- API response time: ~2.1 seconds
- Concurrent users handled successfully
- No data loss during load testing

7. RESULTS

7.1 Output Screenshots

Include screenshots of:

- Registration Page
- Login Page
- Doctor Dashboard
- Patient Dashboard
- Appointment Booking Page
- Admin Panel

8. ADVANTAGES & DISADVANTAGES

Advantages

- Reduces hospital crowding
- Saves time
- Secure authentication
- Scalable architecture
- Cloud-based database

Disadvantages

- Requires internet access
- Basic version does not include payment integration
- Dependent on server availability

9. CONCLUSION

The SmartDoc Appointment System successfully digitizes the healthcare appointment booking process. The MERN-based architecture ensures scalability, security, and performance. The project meets all functional and non-functional requirements and is ready for deployment.

10. FUTURE SCOPE

- Mobile application integration
- Online payment gateway
- Telemedicine video consultation
- AI-based doctor recommendation
- SMS notification system

11. APPENDIX

Source Code

Available in GitHub repository.

GitHub & Project Demo Link

(Add your GitHub link here)

Dataset

Not applicable (Live user database via MongoDB Atlas)