Full Stack Development with MERN - Project Documentation

1. Introduction

Project Title: DocSpot

Team Members: Yelamarthi Satya Vasavi - Full Stack Developer (Solo Project)

2. Project Overview

Purpose:

DocSpot is a web-based platform for seamless appointment booking between patients and doctors. It reduces wait times, simplifies scheduling, and provides an intuitive interface.

Features:

- User authentication
- Doctor and patient profiles
- Real-time appointment booking
- Admin panel for managing users
- Search/filter doctors by specialty
- Dashboards for both patients and doctors

3. Architecture

Frontend:

Built using React.js with dynamic routing, component-based design, and responsive UI.

Backend:

Developed using Node.js and Express.js for API handling and business logic.

Database:

MongoDB with Mongoose ORM for schema and data management.

4. Setup Instructions

Full Stack Development with MERN - Project Documentation

Prerequisites:

- Node.js
- MongoDB

Installation:

1. Clone the repository
2. Install dependencies using npm
3. Set up environment variables in a `.env` file

Environment Variables:

MONGO_URI=your_mongo_connection

JWT_SECRET=your_secret

PORT=5000

5. Folder Structure

Client:

- src/components: Reusable components
- src/pages: Views like Login, Dashboard
- App.js: Routing

Server:

- routes/: API endpoints
- controllers/: Logic for routes
- models/: MongoDB schemas
- server.js: Entry point

6. Running the Application

Frontend:

$ cd client && npm start

Backend:

$ cd server && npm start

7. API Documentation

- POST /api/register: Register new user
- POST /api/login: Login and receive JWT
- GET /api/doctors: List doctors
- POST /api/appointments: Book appointment
- GET /api/appointments/:id: Get user appointments

8. Authentication

JWT (JSON Web Token)-based authentication.

- Token generation at login
- Middleware checks token validity
- Routes protected for doctor/patient/admin

9. User Interface

Note: No screenshots provided as per user's request.

10. Testing

Manual testing with Postman and browser.

Basic form and route validations verified.

11. Demo Link

View project demo:

https://drive.google.com/drive/folders/1pteT8STdObONWwELNDHRK9biItLuiJ-1

Full Stack Development with MERN - Project Documentation

12. Known Issues

- Appointment cancel feature not implemented
- Role-based access needs more restrictions
- UI not fully responsive on small screens

13. Future Enhancements

- Payment integration
- Feedback/rating system
- Video consultation module
- Email/SMS notifications

14. Sample Code: Appointment Booking

```
// Sample booking logic (Node.js/Express)
router.post('/book', async (req, res) => {
  const { userId, doctorId, date, time } = req.body;
  try {
    const appointment = new Appointment({ userId, doctorId, date, time });
    await appointment.save();
    res.status(201).send('Appointment booked successfully');
  } catch (err) {
    res.status(500).send('Error booking appointment');
  }
});
```

15. Sample Code: MongoDB Schema

```
// Appointment Schema (Mongoose)
const mongoose = require('mongoose');
const appointmentSchema = new mongoose.Schema({
```

```
  userId: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
  doctorId: { type: mongoose.Schema.Types.ObjectId, ref: 'Doctor' },
  date: String,
  time: String,
  status: { type: String, default: 'Pending' }
});
module.exports = mongoose.model('Appointment', appointmentSchema);
```