Back end

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const app = express();

app.use(cors());
app.use(express.json());

// MongoDB Connection
mongoose.connect('mongodb://localhost:27017/health_app', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

// Appointment Model
const Appointment = mongoose.model('Appointment', {
  patientName: String,
  doctorName: String,
  date: String,
  time: String,
});

// Routes
app.post('/api/appointments', async (req, res) => {
  const { patientName, doctorName, date, time } = req.body;
  if (!patientName || !doctorName || !date || !time) {
    return res.status(400).json({ message: 'All fields are required' });
  }

  const appointment = new Appointment({ patientName, doctorName, date, time });
  await appointment.save();
  res.status(201).json({ message: 'Appointment booked successfully', appointment });
});

app.get('/api/appointments', async (req, res) => {
  const appointments = await Appointment.find();
  res.json(appointments);
});

// Server Start
const PORT = 5000;
app.listen(PORT, () => console.log(`Server running on http://localhost:${PORT}`));
```

Front end

```
import React, { useState, useEffect } from 'react';
import axios from 'axios';

function App() {
  const [form, setForm] = useState({
    patientName: '',
    doctorName: '',
    date: '',
    time: '',
  });

  const [appointments, setAppointments] = useState([]);

const handleChange = (e) => {
  setForm({ ...form, [e.target.name]: e.target.value });
};

const handleSubmit = async (e) => {
  e.preventDefault();
  await axios.post('http://localhost:5000/api/appointments', form);
  setForm({ patientName: '', doctorName: '', date: '', time: '' });
  fetchAppointments();
};

const fetchAppointments = async () => {
  const res = await axios.get('http://localhost:5000/api/appointments');
  setAppointments(res.data);
};

useEffect(() => {
  fetchAppointments();
}, []);

return (
  <div className="p-6 max-w-xl mx-auto">
    <h1 className="text-2xl font-bold mb-4">Health Appointment Booking</h1>
    <form onSubmit={handleSubmit} className="space-y-3">
      <input
        type="text"
        name="patientName"
        placeholder="Patient Name"
        value={form.patientName}
```

```
          onChange={handleChange}
          className="w-full p-2 border rounded"
          required
        />
        <input
          type="text"
          name="doctorName"
          placeholder="Doctor Name"
          value={form.doctorName}
          onChange={handleChange}
          className="w-full p-2 border rounded"
          required
        />
        <input
          type="date"
          name="date"
          value={form.date}
          onChange={handleChange}
          className="w-full p-2 border rounded"
          required
        />
        <input
          type="time"
          name="time"
          value={form.time}
          onChange={handleChange}
          className="w-full p-2 border rounded"
          required
        />
        <button type="submit" className="w-full bg-blue-600 text-white p-2 rounded">
          Book Appointment
        </button>
      </form>

      <div className="mt-6">
        <h2 className="text-xl font-semibold">Upcoming Appointments</h2>
        <ul className="mt-2">
          {appointments.map((apt, idx) => (
            <li key={idx} className="border p-2 rounded mb-2">
              <strong>{apt.patientName}</strong> with Dr. {apt.doctorName} on {apt.date} at
{apt.time}
            </li>
          ))}
        </ul>
```

```
      </div>
    </div>
  );
}

export default App;
```

Mango DB model

```
const mongoose = require('mongoose');

const appointmentSchema = new mongoose.Schema({
  patientName: String,
  doctorName: String,
  date: String,
  time: String,
});

module.exports = mongoose.model('Appointment', appointmentSchema);
```