#Solution   
  
from flask import Flask, request, jsonify

from flask\_sqlalchemy import SQLAlchemy

app = Flask(\_\_name\_\_)

app.config['SQLALCHEMY\_DATABASE\_URI'] = r'sqlite:///C:/Users/harsh/Desktop/harsha\_wipro/final/app.db'

app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS'] = False

db = SQLAlchemy(app)

# Doctor Model

class Doctor(db.Model):

id = db.Column(db.String, primary\_key=True)

name = db.Column(db.String, nullable=False)

specialty = db.Column(db.String, nullable=False)

available\_slots = db.Column(db.PickleType, nullable=False)

on\_leave = db.Column(db.Boolean, default=False)

# Patient Model

class Patient(db.Model):

id = db.Column(db.String, primary\_key=True)

name = db.Column(db.String, nullable=False)

age = db.Column(db.Integer, nullable=False)

contact = db.Column(db.String, nullable=False)

# Appointment Model

class Appointment(db.Model):

id = db.Column(db.Integer, primary\_key=True, autoincrement=True)

patient\_id = db.Column(db.String, db.ForeignKey('patient.id'), nullable=False)

doctor\_id = db.Column(db.String, db.ForeignKey('doctor.id'), nullable=False)

time\_slot = db.Column(db.String, nullable=False)

# Home Route (Fixes 404 Issue)

@app.route('/')

def home():

return "Doctor Appointment API is running!"

# Register Doctor

@app.route('/register\_doctor', methods=['POST'])

def register\_doctor():

data = request.json

if Doctor.query.get(data['doctor\_id']):

return jsonify({"success": False, "message": "Doctor ID already exists!"})

new\_doctor = Doctor(

id=data['doctor\_id'],

name=data['name'],

specialty=data['specialty'],

available\_slots=data['available\_slots']

)

db.session.add(new\_doctor)

db.session.commit()

return jsonify({"success": True, "message": "Doctor registered successfully!"})

# Register Patient

@app.route('/register\_patient', methods=['POST'])

def register\_patient():

data = request.json

if Patient.query.get(data['patient\_id']):

return jsonify({"success": False, "message": "Patient ID already exists!"})

new\_patient = Patient(

id=data['patient\_id'],

name=data['name'],

age=data['age'],

contact=data['contact']

)

db.session.add(new\_patient)

db.session.commit()

return jsonify({"success": True, "message": "Patient registered successfully!"})

#Book Appointment

@app.route('/book\_appointment', methods=['POST'])

def book\_appointment():

data = request.json

doctor = Doctor.query.get(data['doctor\_id'])

if not doctor:

return jsonify({"success": False, "message": "Doctor not found!"})

if data['time\_slot'] not in doctor.available\_slots:

return jsonify({"success": False, "message": "Time slot not available!"})

existing\_appointment = Appointment.query.filter\_by(doctor\_id=data['doctor\_id'], time\_slot=data['time\_slot']).first()

if existing\_appointment:

return jsonify({"success": False, "message": "Time slot already booked!"})

new\_appointment = Appointment(

patient\_id=data['patient\_id'],

doctor\_id=data['doctor\_id'],

time\_slot=data['time\_slot']

)

db.session.add(new\_appointment)

db.session.commit()

return jsonify({"success": True, "message": "Appointment booked successfully!"})

#Get Doctor's Appointments

@app.route('/get\_appointments/<doctor\_id>', methods=['GET'])

def get\_appointments(doctor\_id):

appointments = Appointment.query.filter\_by(doctor\_id=doctor\_id).all()

return jsonify([{"patient\_id": a.patient\_id, "doctor\_id": a.doctor\_id, "time\_slot": a.time\_slot} for a in appointments])

# Cancel Appointment

@app.route('/cancel\_appointment', methods=['POST'])

def cancel\_appointment():

data = request.json

appointment = Appointment.query.filter\_by(patient\_id=data['patient\_id'], doctor\_id=data['doctor\_id'], time\_slot=data['time\_slot']).first()

if not appointment:

return jsonify({"success": False, "message": "Appointment not found!"})

db.session.delete(appointment)

db.session.commit()

return jsonify({"success": True, "message": "Appointment cancelled successfully!"})

# Run App & Create Tables

if \_\_name\_\_ == '\_\_main\_\_':

with app.app\_context():

db.create\_all()

app.run(debug=True)  
  
  
