**LUMA health project**

**Problem Definition**

A busy hospital has a list of dates that a doctor is available to see patients. Their process is manual and error prone leading to overbooking. They also have a hard time visualizing all of the available time for a doctor for specific dates.

**Interview Task**

Create a REST API that enables a simple scheduling system that manages doctor availabilities and allows patients to book appointments.

**System Dependencies:**

* Requires node.js with sequelize library and mocha test framework
* PostgreSQL.
* Postico or any DB viewer to look at the records in the tables.
* Postman app to test the APIs

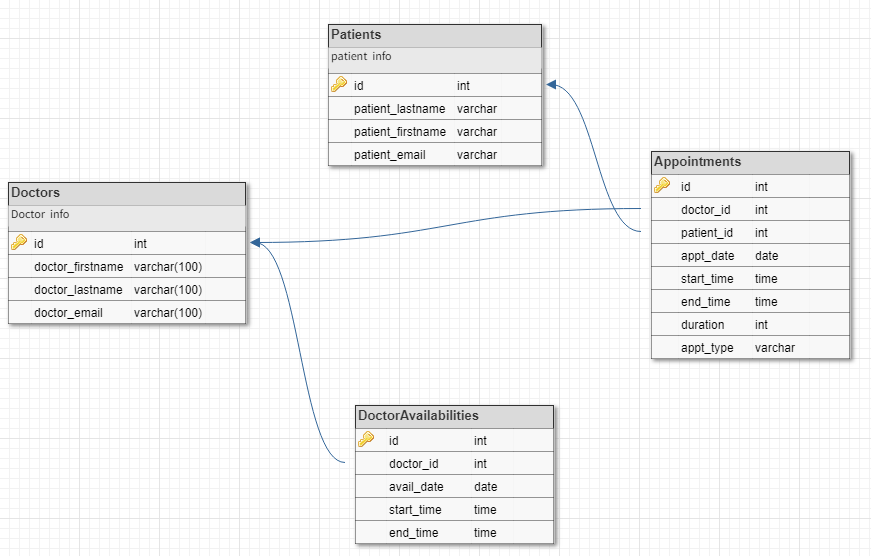
**Setup and Instructions to run:**

* Install node.js and postgresSQL.
* Download the test zip file and extract the directory that contains the scripts.
* Start the postgresSQL server and create a database called ‘**luma’**.
* Open a command prompt window/terminal.
* Navigate to the directory where the files are extracted.
* Run **npm install**
* Run the **sequelize** **db:migrate** command to create the tables where sequelize is an ORM.
* Start the server using **npm start** command to start the server.
* Download and install postman app using which we can test the APIs.
* Open postman and submit the api’s that you want to test from the “APIs in the system” list.
* The results should be displayed in the postman and can also be verified using postico.

**Assumptions:**

* The REST APIs have to be called with the necessary params mentioned in the readme for each API.
* On a particular date, there will be only one entry for the doctor in the availability table.

**Data Models:**



**Patient:**

The patient model contains all the necessary information about the patients such as the patient id (primary key), name, email etc. It can be joined with the appointments to retrieve the appointments made by the particular patient.

This model can be used to track the patients.

**Doctor:**

Contains all necessary information about the doctors in the hospital I.e doctorId, name and other details and it can be joined with the availability model to check the doctor’s availability and appointments model to check current appointments.

This model can be used to track the doctors.

**Availability:**

Describes the availability of the doctors in the hospital. Contains fields such as DoctorId, date, start and end time available.

This can be used to track the doctor’s working hours and days.

**Appointments:**

To track the currently booked appointments. Each appointment entry has the fields appointmentID, patientID, doctorID, appointmentDate, startTime, endTime and duration of the appointment.

**REST API**

To run and test the APIs, type the corresponding URLs in postman and click submit.

Before running the APIs, please create a doctor, patient and availability entry.

**Required APIs:**

* **Find a doctor's working hours**

**To get the doctor’s working hours for a particular date:**

/api/doctor/getworkinghours/:id/:date

**To get the doctor’s working hours for all dates:**

/api/doctor/getallworkinghours/:id

* **Book a doctor opening:**

/api/appointment/bookappointment', Appointment.bookAppointment

request type: post

params required:

patientId

doctorId

apptDate (YYYY-MM-DD)

stime (HH:MM)

etime (HH:MM)

* **Create and update the list of doctor's working hours**

**To create an availability entry for a particular doctor.**

/api/doctor/createavailability

Params:

id

date

stime

etime

**To create a list of doctors’ availability:**

/api/doctor/createlist

Array of Params:

id

date

stime

etime

**To update the doctor availability:**

/api/doctor/updateAvailability

Params:

id

date

stime

etime

**Other APIs:**

* **Create a patient:**

/api/patient/create

* **Create a doctor:**

/api/doctor/create

* **Retrieve list of all patients:**

/api/patient/getall

* **Retrieve a particular patient:**

/api/patient/get/:id

* **Retrieve all appointments for a patient**

/api/patient/getappts/:id

* **Retrieves all doctors:**

/api/doctor/getall

* **Retrieve a particular doctor:**

/api/doctor/get/:id

* **Retrieve all appointments for a doctor:**

/api/doctor/getappts/:id

**Future enhancements:**

* A GUI to send and receive API requests and responses.
* API to cancel appointments.