REST API DOCUMENTATION

1) API endpoints (routes)

Endpoint	**Parameters expected**	**JSON** response
/specializations (returns all available specializations in the system)	none	{'data' : list of specializations} Eg: `{ "data": [ "dentist", "dermatologist", "cardiologist" ] }`
/doctors (returns all doctors of specialization equal to the chosen one)	{'specialization: *chosen_specialization*} Eg: {'specialization: 'dentist'}	{'data' : list of doctors' JSON objects} Eg: `{ "data": [ { "id": 1, "name": "mark", "specialization": "dentist" }, // other doctors as JSON objects ] }`
/dr_appointments (returns all free appointments of the chosen dr)	{'option_id': 0 or 1 'dr_id': chosen_dr_id 'appointment_id': appointment_id} Eg: {'option': 0, 'dr_id': 1} or {'option': 1, appointment_id': 3} **Note: you either send the dr_id or the appointment_id, if you will send the appointment_id set option to 1, if otherwise set option to 0**	{'data': list of doctor's free appointments as JSON objects} Eg: `{ "data": [ { "appointment_id": 3, "end_date": "2020-12-30T04:25:47.028308", "start_date": "2020-12-30T04:25:47.028308", "status": "free" }, // another appointment ]`

		}
/normal_appointment (creates normal appointment)	{‘appointment_id’ : appointment_id, ‘patient_id’ : patient_id, ‘patient_name’ : patient_name, ‘patient_age’ : patient_age, ‘patient_email’ : patient_email, ‘patient_phone_number’ : patient_phone_num, ‘specialization’ : specialization}	{‘status’: ‘False’} or {‘status’: ‘False’} True: appointment created False: didn’t
/urgent_appointment (created urgent appointment)	{‘patient_id’ : patient_id, ‘patient_name’ : patient_name, ‘patient_age’ : patient_age, ‘patient_email’ : patient_email, ‘patient_phone_number’ : patient_phone_num, ‘specialization’ : specialization}	Same as above
/update_appointment (update current appointment with a new one)	{‘old_appointment_id’ : old_appointment_id, ‘new_appointment_id’ : new_appointment_id}	Same as above
/retrieve_patient_appointments (retrieves all this patient’s appointments)	none	{‘data’: list of patient appointments} See JSON response below the table
/cancel_appointment	{‘appointment_id’ : appointment_id} Eg: {‘appointment_id’ : 3}	Same as the appointment creation response


```
{
  "data": [
    {
      "appointment_type": "normal",
      "patient_age": 30,
      "patient_email": "x@domain.com",
      "patient_name": "another name",
      "patient_phone_number": null,
      "specialization": "dentist"
    },
    {
      "appointment_type": "normal",
      "patient_age": 30,
      "patient_email": "ashraf@domain.com",
      "patient_name": "ashraf",
      "patient_phone_number": null,
      "specialization": "dentist"
    }
  ]
}
```

2) Expected parameters data types

specialization, patient_name, patient_email ---> string

dr_id, appointment_id, patient_id, patient_age, patient_phone_number ---> integers

3) Notes for frontend developers

3.1) create normal appointment:

1) on load: /specialization -> populate the retrieved specializations list in the menu

2) When a specialization is selected, (/doctors) set params to {'specialization: *chosen_specialization*}

3) Populate the returned list of doctors in the doctors menu

4) When a doctor is selected, (/dr_appointments) set params to {'option': **0**, 'dr_id': selected dr id}

5) Populate the returned list of appointments in the appointments menu

5) When an appointment is selected (/normal_appointment) set params to:

{'appointment_id' : appointment_id,
'patient_id' : patient_id,
'patient_name' : patient_name,
'patient_age' : patient_age,
'patient_email' : patient_email,
'patient_phone_number' : patient_phone_num,
'specialization' : specialization}

3.2) create urgent appoint:

(/urgent_appointment)

Same as the create normal appointment flow, but skip step 3, and don't pass 'appointment_id' in the params

3.3) cancel appointment:

1) Retrieve all this patient's appointments -> (/retrieve_patient_appointments)

2) Populate the retrieved patient appointments in the HTML menu (select tag)

3) When the user selects an appointment to cancel, store the appointment_id

4) (/cancel_appointment) ---> set params to {‘appointment_id’ : appointment_id}

3.4) update the appointment’s date only:

1) Retrieve all this patient’s appointments -> (/retrieve_patient_appointments)

2) Populate the retrieved patient appointments in the HTML menu (select tag)

3) When the user selects an appointment to update, store the appointment’s id as old_appointment_id

4) (/dr_appointments) set the request’s paramers to {‘option’: **1**, ‘old_appointment_id’: old_appointment_id}

5) After the above request receive from server {‘data’ : list of free appointments}

6) (/update_appointment) set params to {‘old_appointment_id’ : old_appointment_id,

‘new_appointment_id’ :the selected appointment from the above list}

3.5) The other update scenarios:

1) Retrieve all this patient’s appointments ---> (/retrieve_patient_appointments)

2) Populate the retrieved patient appointments in the HTML menu (select tag)

3) When the user selects an appointment to update, store the appointment’s id as old_appointment_id

4) Maybe move to a new update menu similar to the create appointment page, but **without** filling user details (email, age, email, number, name)

5) (/specialization) ---> populate the retrieved specializations list in the menu

6) (/doctors) set params to {‘specialization: *chosen_specialization*}

7) Populate the returned list of doctors in the doctors menu

8) When a doctor is selected, (/dr_appointments) set params to {‘option’: **1**, ‘dr_id’: selected dr id}

9) Populate the returned list of appointments in the appointments menu

10) (/update_appointment) set params to {‘old_appointment_id’ : old_appointment_id,

‘new_appointment_id’ :the selected appointment from the above list}