1. Health Care Service Backend

You have to create an API service for Health Care Service.

Below are the details of the API endpoints.

Backend:

These are the API endpoints

login
register

addPatient
fetchSinglePatient
editProfile
diseases

bookAppointment

fetchAppointment

deleteAppointment

singlePatientAppointment

getProfile

Functionalities of endpoint are given below:

login:

In this endpoint, you have to validate the login details of the user with received parameters like uname, pwd and respond with JSON data object as below

Key	Value
status	boolean
message	string
token	string
uid	string
mobile	number
email	string
location	string
userName	string

Details about the token in the data is explained at the end of question in middleware section.

register:

In this endpoint, you will receive request parameters like email, pwd, uname, mobile,location.

you should get these parameters and insert into the database as an object which will be in the below structure.

Key	Value
_id	mongoose.Types.ObjectId()
email	string
pwd	string
userName	string
mobile	Number
location	string

Note: Storing password as plain text is not at all good practice, This is all about demonstration purpose to create an API.
The general practice could be hash the password and store in the database.

addPatient:

In this endpoint, you will receive request parameters like fname, lname, gender, dob, mobile, email, desc, userId

you should get these parameters and insert into the database as an object which will be in the below structure.

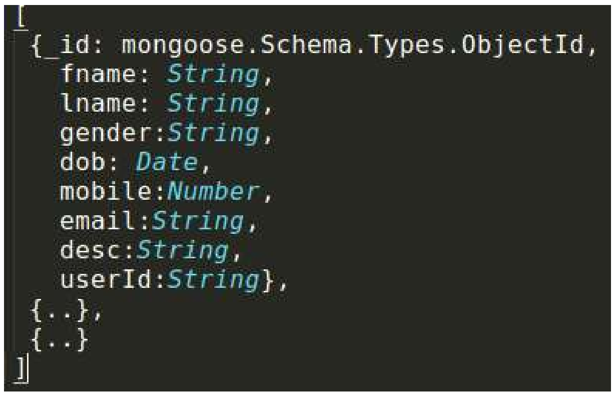
Key	Value
_id	mongoose.Types.ObjectId()
fname	string
lname	string
gender	string
dob	Date
mobile	Number
email	string
desc	string
userId	string

fetchSinglePatient:

In this endpoint, you will receive request parameter like patientId.

you should get this parameter and you should query the patient collections with received patientId and return the response which can be as below.

You use MongoDB as database and details about the required collections and their structures are the end of the question.



```
[
 {_id: mongoose.Schema.Types.ObjectId,
   fname: String,
   lname: String,
   gender:String,
   dob: Date,
   mobile:Number,
   email:String,
   desc:String,
   userId:String},
 {..},
 {..}
]
```

After querying , if no match with patientId is found in database, response can be an empty array too.

editProfile:
In this endpoint, you will receive request parameters like email, mobile, location, uid.

Fetch the object from user collection based on uid, edit the parameters of the object and save it in the database.

You should save the object as below

Key	Value
_id	mongoose.Types.ObjectId()

Key	Value
userName	string
pwd	string
mobile	Number
email	string
location	string

You need to edit only received parameter values only.

diseases:

Request method used in this API is GET.

In this endpoint, you have to query the diseases collections and return the JSON data obtained from MongoDB.

The diseases data that should be in the "diseases" collection is mentioned below:

```
{diseases:['flu','cold','fever','typhoid',.....]}
```

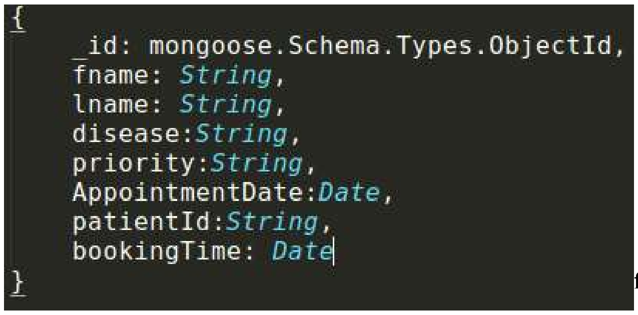
The array can contain any disease name you want.

bookAppointment:

In this endpoint, you will receive request parameters like fname, lname, disease, priority, tentativeDate, patientId, registeredTime.

You have to save these details in 'appointments' collection in MongoDB.

The JSON object that should be saved should be as below:

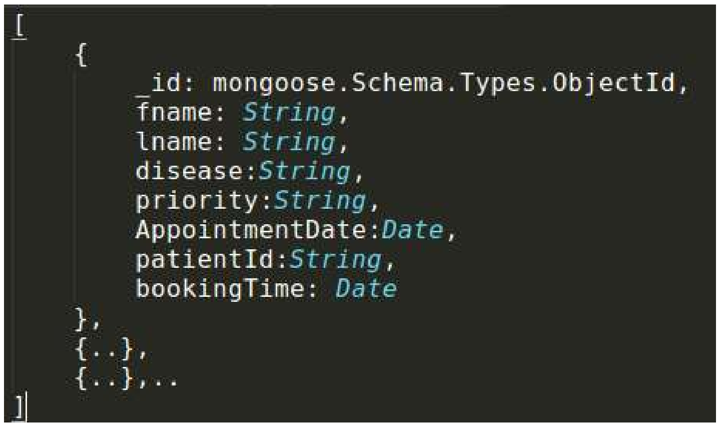


```
{
    _id: mongoose.Schema.Types.ObjectId,
    fname: String,
    lname: String,
    disease:String,
    priority:String,
    AppointmentDate:Date,
    patientId:String,
    bookingTime: Date
}
```

fetchAppointment:

In this endpoint, You have to GET all the appointments from the 'appointments' collections.

The response after querying 'appointments' collection can be as below:



```
[
    {
        _id: mongoose.Schema.Types.ObjectId,
        fname: String,
        lname: String,
        disease:String,
        priority:String,
        AppointmentDate:Date,
        patientId:String,
        bookingTime: Date
    },
    {..},
    {..},..
]
```

After querying , if there are no appointments, response can be an empty array too.

deleteAppointment:

In this endpoint, you will receive an request parameter like appointmentId,

You have to delete object with id equals to appointmentId.

On successful deletion, you have to return an JSON object with 'status' as key and "success" as value in response with status code as 200.

On deletion failed, you have to return an JSON object with 'status' as key and "failure" as value in response with status code 404.

NOTE: Every Object in MongoDB has an object id with key as '_id'.

In this API, received appointmentId is the _id of the object that you need to delete.

singlePatientAppointments:

If you remember, in the fetchAppointment API, you fetch all the appointments available in database.

But in this API you have to fetch a single patient appointment based on the patientId which we store while booking an appointment using bookAppointment API.

In this API, you receive request parameter like patientId.

You have to query the 'appointments' collection based on patientId and return the JSON response.

getProfile:

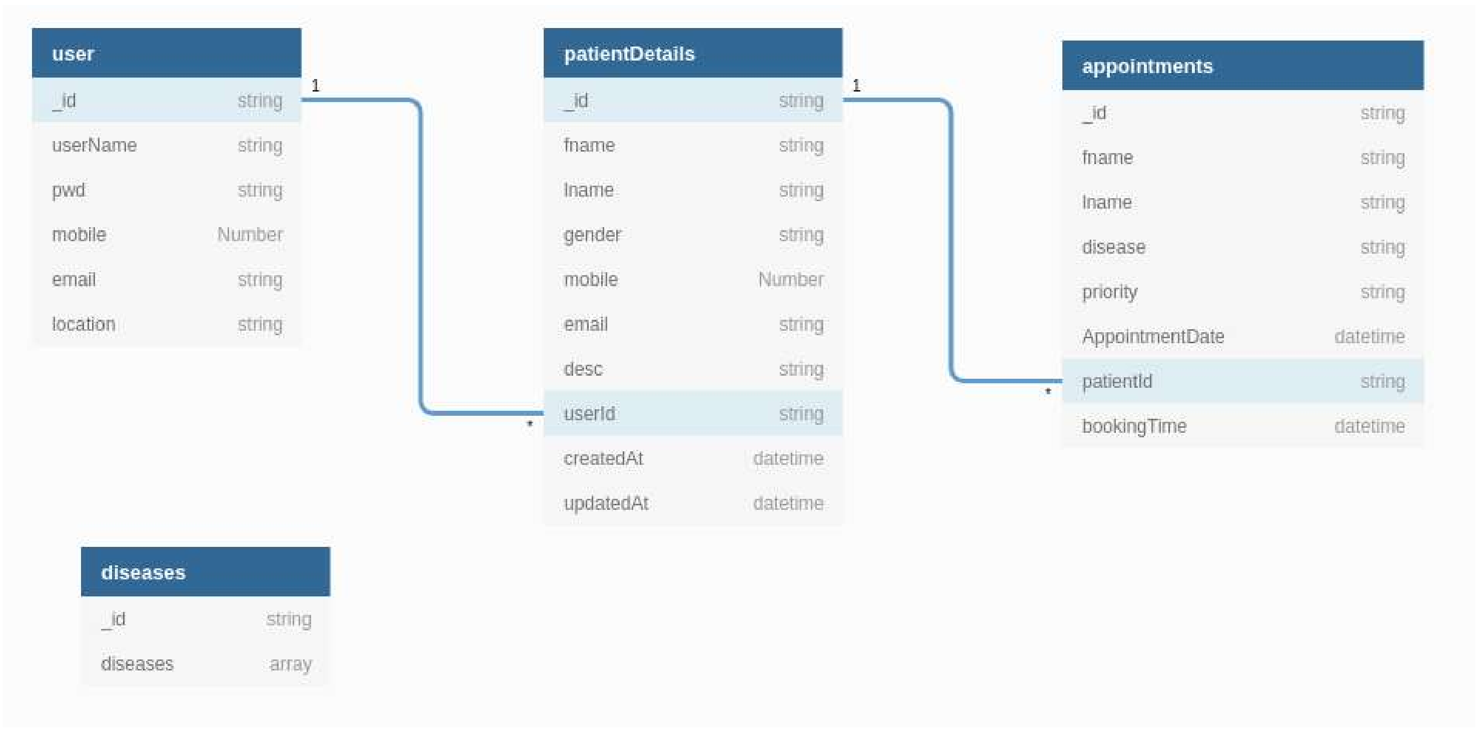
In this API, You have send the user details.

You receive request parameters like uid, with which you have to query 'user' collection and send the query results as JSON.

BUT the response object shouldn't have the 'pwd' property.

perform operation to delete the property from the response object.

Internal Structure of collection and meaning of attributes in MongoDB:



While structuring the database models, we consider below factor:

user:

_id	unique ID generated By MongoDB
userName	username of the user
pwd	Password of user
mobile	user's mobile number
email	email address of user
location	location of user

patientDetails:

_id	unique ID generated By MongoDB
fname	firstname of patient
lname	last name of patient
gender	gender of Patient
dob	Date of birth of patient
mobile	phone number of patient
email	email address of patient
desc	description provided by patient
userId	userId of user who added patient data

This model contains 'timestamps' as true, which create createAt and updateAt field in objects which indicates the time at which object is created and updated.

appointments:

_id	unique ID generated By MongoDB
fname	first name of patient
lname	last name of patient
disease	name of the disease patient has
priority	Priority of patient
AppointmentDate	date of appointment
patientId	Id of patient
bookingTime	Time at which booking is made

diseases:

_id	unique ID generated By MongoDB
diseases	Array of diseases

IMPORTANT:

Middleware:

You need to create an Middleware to validate the API calls.

For Middleware, you have to use 'json web token' (jwt) with below rules:

1) login and registration API endpoint don't need middleware. If login details are validated , you send response with json web token as token parameter.

2) With obtained token,for every request, the request should have 'x-access-token' or 'authorization' as key and token as value.

Steps to be followed:

- **Run -> install :** To install all the node modules for both frontend and backend, to install MongoDB and insert dummy data into it
- **Run -> Run:** To see running application
- Before running the testcases execute **bash test.sh** from the terminal.
- **Run -> Test:** To see results of testing the API and react behaviour as expected