Salon Booking System

- B.Sri Bhargav Ram

Generally, I have been given the task of salon booking system, to prepare the part of backend using Node.js

Tech Stack Used: I have used Tech stack of

1. Nodejs
2. Expressjs
3. MongoDB database

Task-1 Authentication:

I have implemented user authentication using JWT (JSON Web Tokens), and stored it in a database.

➢ I created a User data model, which is present in the file of User.js in the models folder in the given code.

➢ Basically, while User signing up, I am taking the name, email, password, contact number of the user.

➢ Passwords are stored by hashing.. using bcryptjs.

➢ When User logs in, it checks if a user exists in the database, and it logs in, giving jwt-token as a response.

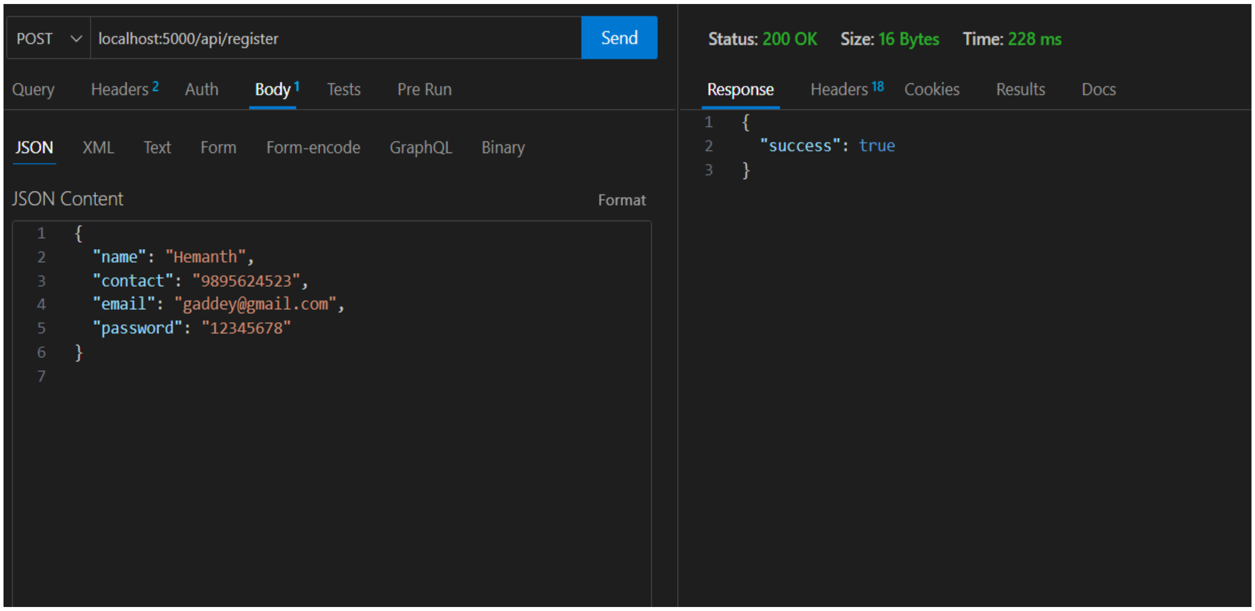
➢ **API Endpoint:** localhost:5000/api/register [POST] , for signing up.
localhost:5000/api/login [POST] , for logging in.
localhost:5000/api/logout [POST], for logout

➢ While Registering, Request should be of format:
- Name
- Email
- Password

    - Contact

- ➢ While logging in, Request should be of format:
    - Email
    - Password

Everything is of type String, and response will be: **Status-code 200, Success.**



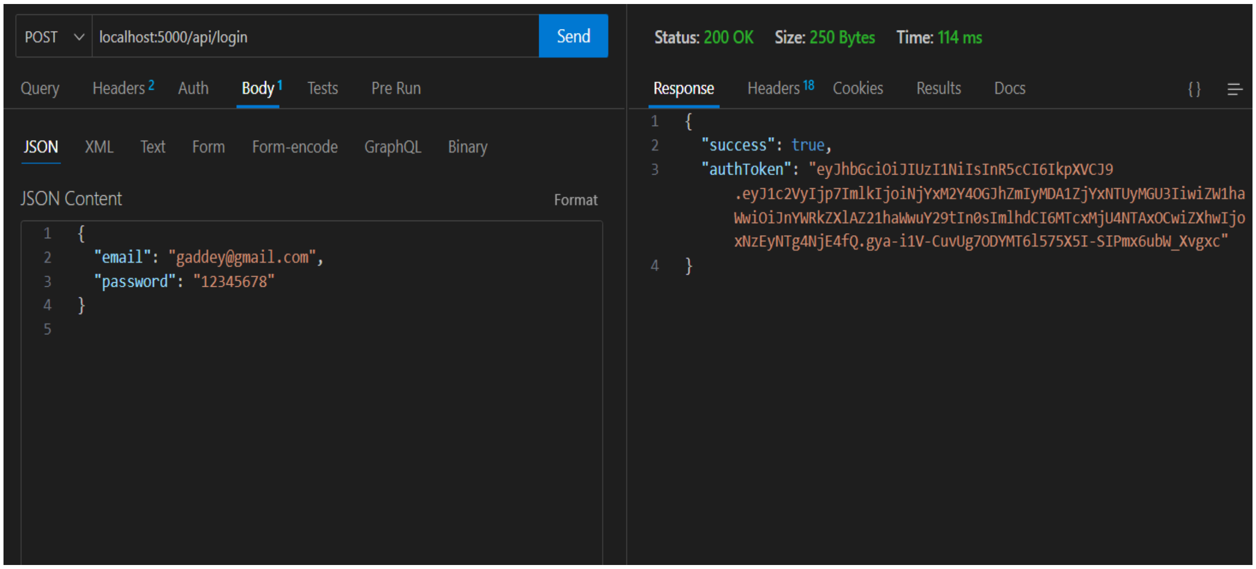
localhost:5000/api/register

JSON Content

```
{
  "name": "Hemanth",
  "contact": "9895624523",
  "email": "gaddey@gmail.com",
  "password": "12345678"
}
```

Status: 200 OK Size: 16 Bytes Time: 228 ms

```
{
  "success": true
}
```



localhost:5000/api/login

JSON Content

```
{
  "email": "gaddey@gmail.com",
  "password": "12345678"
}
```

Status: 200 OK Size: 250 Bytes Time: 114 ms

```
{
  "success": true,
  "authToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
      .eyJ1c2VyIjp7ImlkIjoiNjYxM2Y4OGJhZmIyMDA1ZjYxNTUyMGU3IiwiZW1ha
      WwiOiJnYWRkZXlAZ21haWwuY29tIn0sImlhdCI6MTcxMjU4NTAxOCwiZXhwIjo
      xNzEyNTg4NjE4fQ.gya-i1V-CuvUg7ODYMT6l575X5I-SIPmx6ubW_Xvgxc"
}
```

Task-2   Set Availability APIs:

In this part, Salon owners will give the available slots and maxCapacity of people that can occupy a particular slot.

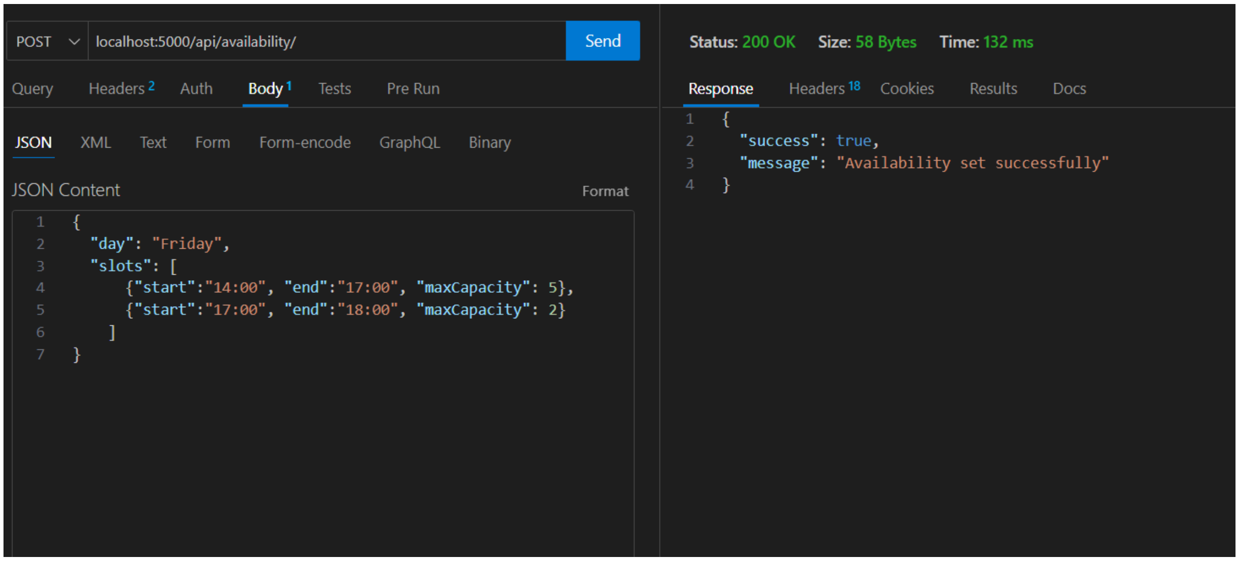
- Here, I have prepared a data model to take “day” as mentioned in sample response and slots, which owners will provide, which contains [start of slot,end of the slot, maxCapacity, currentCapacity(default:0)].

- Here, For example..if the owner edits “On Monday, (we will have some type of Slots)”...., then it will check whether Monday exists in the database. If it exists, it updates slots on monday, else it will create new data.

- If User, need to get slots, he has to select the date on which he needs slots. If he enters some date, the function that i wrote will automatically convert that date to the corresponding day and give the slot details.

- **API Endpoint:** localhost:5000/api/availability [POST] , for updating slots.
  localhost:5000/api/availability/:required_date [GET] for getting slots.

- While Updating Availability, Request should be of format:
  - day (String)
  - slots (Array) [start,end,maxCapacity]



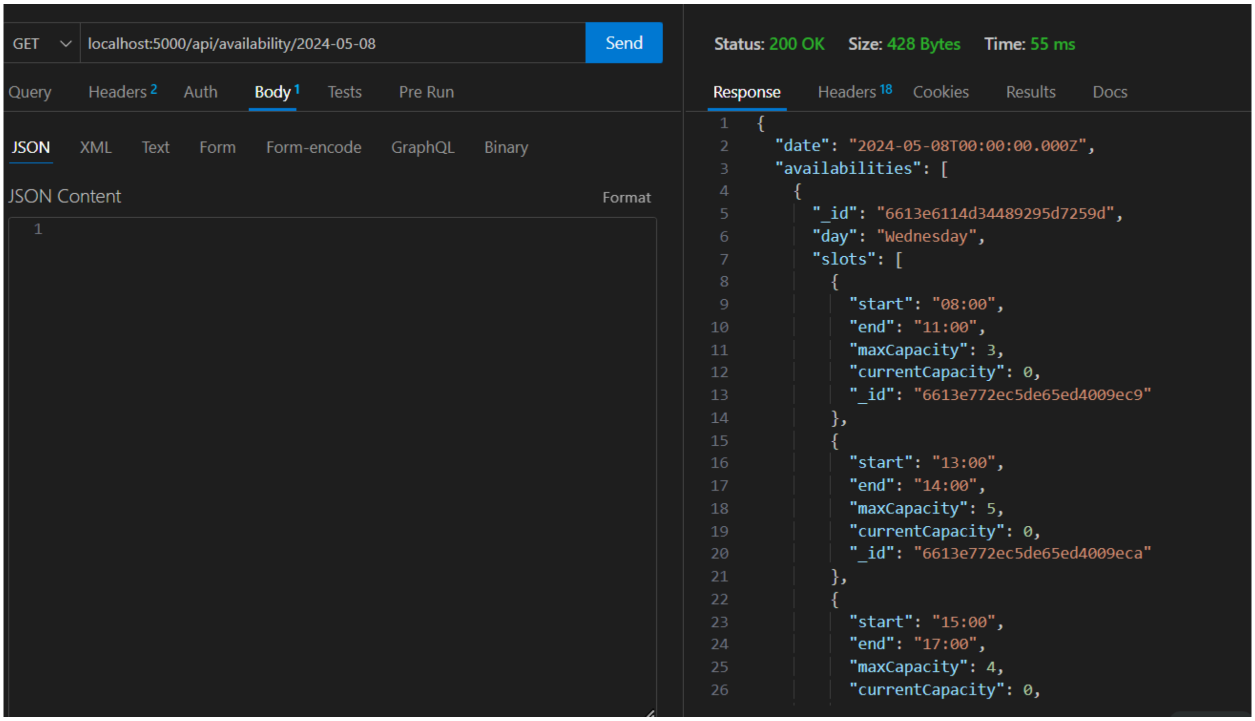
localhost:5000/api/availability/

```
{
  "day": "Friday",
  "slots": [
      {"start":"14:00", "end":"17:00", "maxCapacity": 5},
      {"start":"17:00", "end":"18:00", "maxCapacity": 2}
    ]
}
```

Status: 200 OK   Size: 58 Bytes   Time: 132 ms

```
{
  "success": true,
  "message": "Availability set successfully"
}
```

- ➢ I have ensured that, for each slot, start time will be less than the end time of the slot.



localhost:5000/api/availability/2024-05-08

Status: 200 OK Size: 428 Bytes Time: 55 ms

```
{
  "date": "2024-05-08T00:00:00.000Z",
  "availabilities": [
    {
      "_id": "6613e6114d34489295d7259d",
      "day": "Wednesday",
      "slots": [
        {
          "start": "08:00",
          "end": "11:00",
          "maxCapacity": 3,
          "currentCapacity": 0,
          "_id": "6613e772ec5de65ed4009ec9"
        },
        {
          "start": "13:00",
          "end": "14:00",
          "maxCapacity": 5,
          "currentCapacity": 0,
          "_id": "6613e772ec5de65ed4009eca"
        },
        {
          "start": "15:00",
          "end": "17:00",
          "maxCapacity": 4,
          "currentCapacity": 0,
```

Task-3 Schedule Booking APIs:

In this part, User will book the slot, and at the backend, we will ensure that the number of people booked at a particular slot do not exceed maxCapacity provided by the salon owner.

- ➢ Here, I have prepared data model as follows:
  - ○ userId
  - ○ Date (the user will select)
  - ○ slot (he will only select start slot, since i have fixed one slot time as 15 mins)

NOTE: I assumed the slot timings that the owner will give are the times that shop will open, and maxCapacity is the no. of workers/ chairs available, while one slot is fixed to 15 mins (i have assumed)

- If a user gives a date, my function will convert it to a day, and check the slots according to that day, and give it to the user. Users can book from them.

- If no.of people booked at that time, doesn’t exceed maxCapacity, we will book for that user, else we won't.

- Users can also delete the slot that they have booked.

- **API Endpoint:** localhost:5000/api/bookings [POST] , for booking slot.
  localhost:5000/api/bookings/:id [DELETE] , for deleting slot
  localhost:5000/api/bookings [GET], for getting slots

How to run it Locally?

1. Clone the project from github / download zip file
2. Execute the command: npx nodemon index.js (inside the folder -> 7webs-assignment)

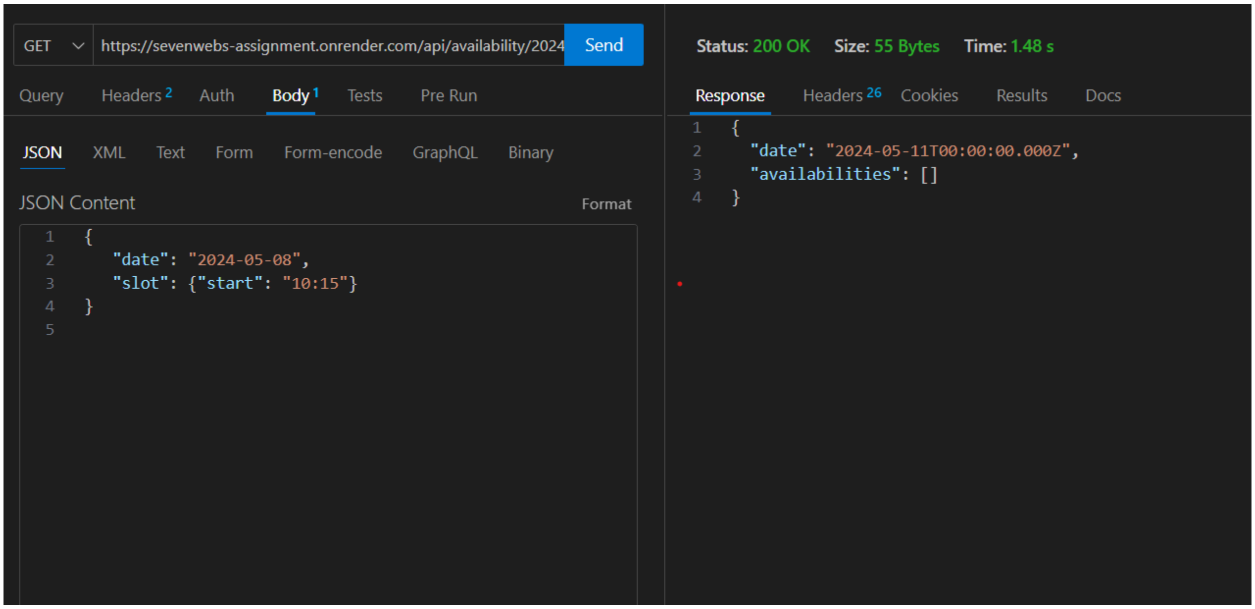
Link of Github: https://github.com/sribhargav1345/7Webs-Assignment

Link of URL: https://sevenwebs-assignment.onrender.com/

Note:

1. I have implemented middleware for request validation, error handling, etc in middleware folder
2. For security reasons, I have used a helmet. It is a middleware library, helps secure our express apps by setting various http headers, which mitigate security vulnerabilities.

After Deploying:



https://sevenwebs-assignment.onrender.com/api/availability/2024

JSON Content

```
{
    "date": "2024-05-08",
    "slot": {"start": "10:15"}
}
```

Status: 200 OK Size: 55 Bytes Time: 1.48 s

```
{
  "date": "2024-05-11T00:00:00.000Z",
  "availabilities": []
}
```