

MONGODB PROJECT

Supreme Salon – Online Salon Appointment Booking System

NAME	: JEEVITHA R
REGNO	: 23BAI1550
CAMPUS	: CHENNAI

PROJECT DESCRIPTION:

The Online Salon Appointment Booking System utilizes **MongoDB Atlas** as its cloud database, structured into three primary collections: **Users**, **Appointments**, and **Feedbacks**.

- **Users Collection:** Stores user information including name, email, password (hashed), and role (customer or admin). This collection manages authentication and role-based access.
- **Appointments Collection:** Records salon appointments booked by users. Each appointment document contains references to the user who booked it, the selected salon service (e.g., haircut, facial), the appointment date, and the chosen time slot.
- **Feedbacks Collection:** Stores customer feedback post-service. Each feedback includes a star rating represented as a numeric value (1 to 5) alongside optional textual comments. The rating is saved as a number in MongoDB for easy querying and analytics.

This full stack system, built with **Node.js** and **Express.js** for the backend, **React.js** for the frontend, and **MongoDB Atlas** as the database, offers a scalable and efficient platform for both customers and salon administrators. Customers can create accounts, book and manage appointments, and submit feedback. Admin users have privileged access to monitor all bookings and feedback through a centralized dashboard.

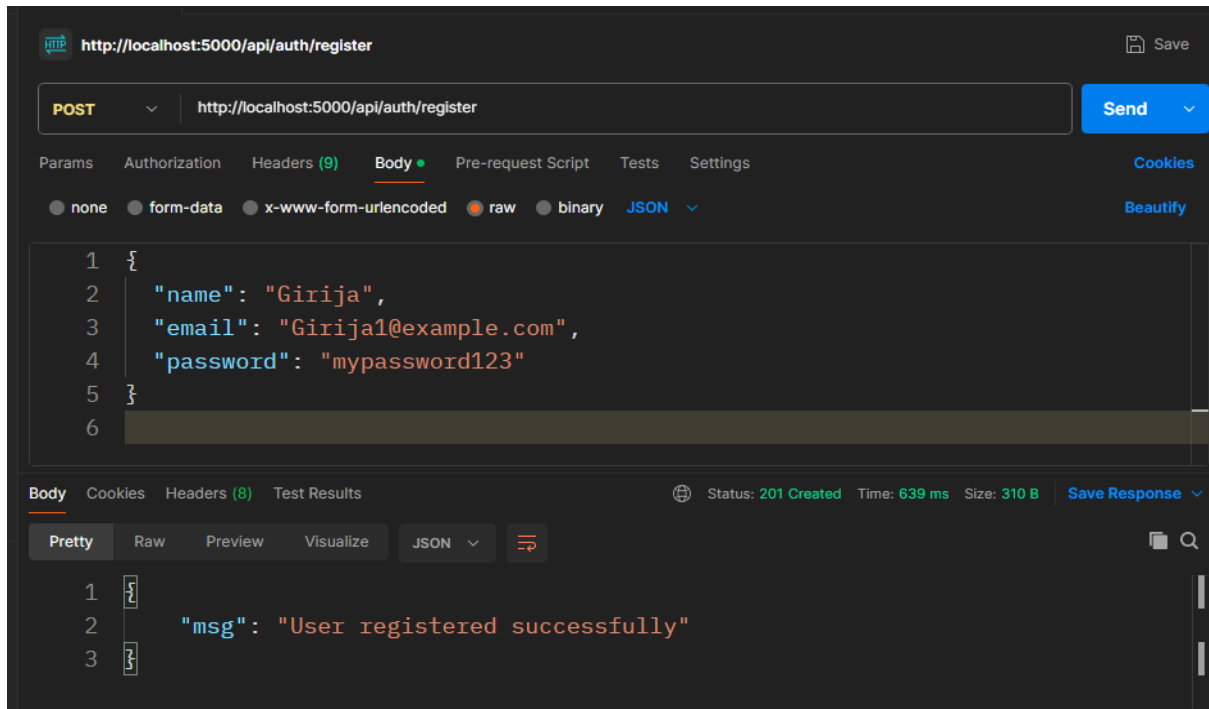
All CRUD operations for users, appointments, services, and feedback are fully supported and validated. These APIs are thoroughly tested using **Postman** through GET, POST, PUT, and DELETE requests to ensure reliability and correctness.

The system ensures secure authentication, smooth client-server communication, and role-based data access, streamlining salon operations and enhancing user experience.

BACKEND & POSTMAN SCREENSHOT

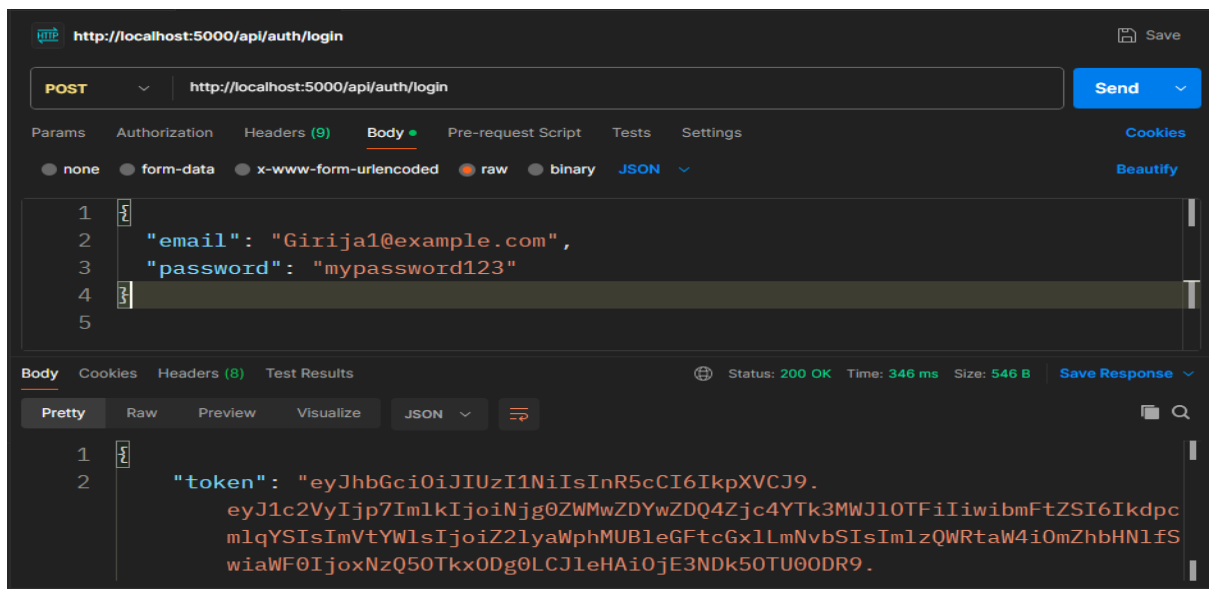
1. User Registration via Postman (POST /auth/register)

- **Type:** POST
- **URL:** `http://localhost:5000/api/auth/register`



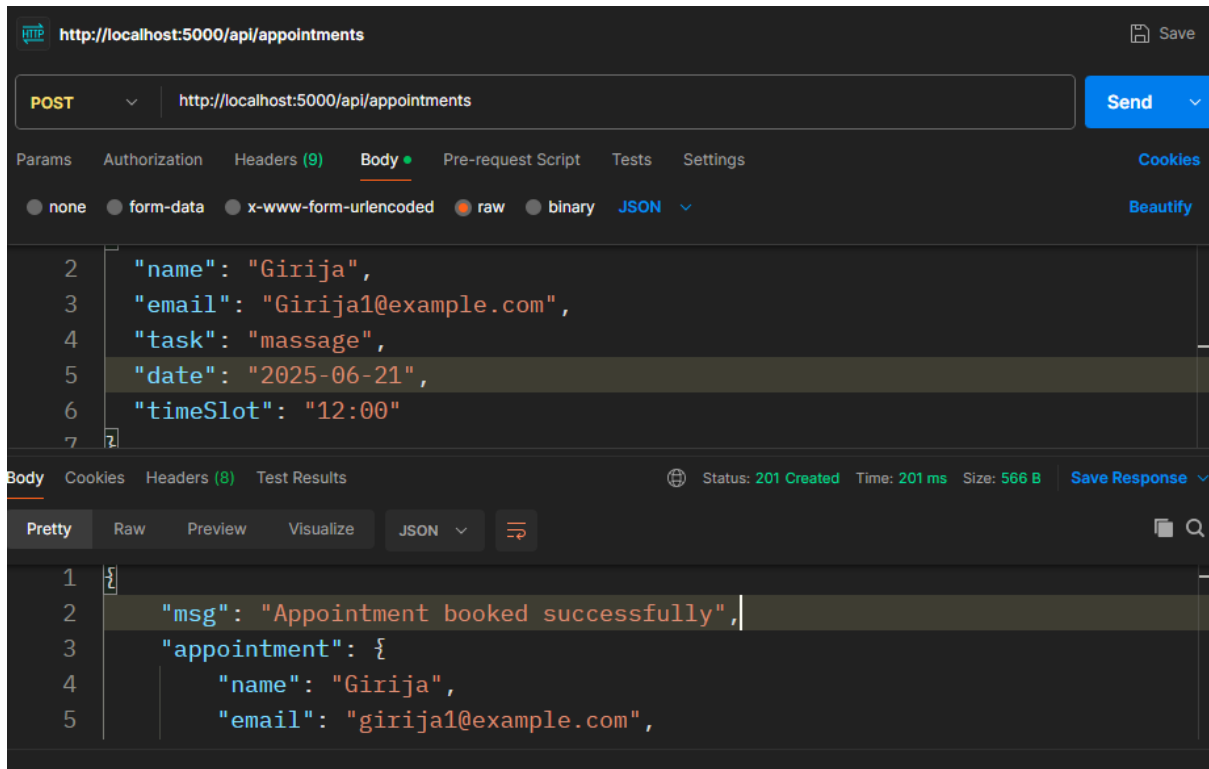
2. User Login (Get Token)

- **Type:** POST
- **URL:** `http://localhost:5000/api/auth/login`

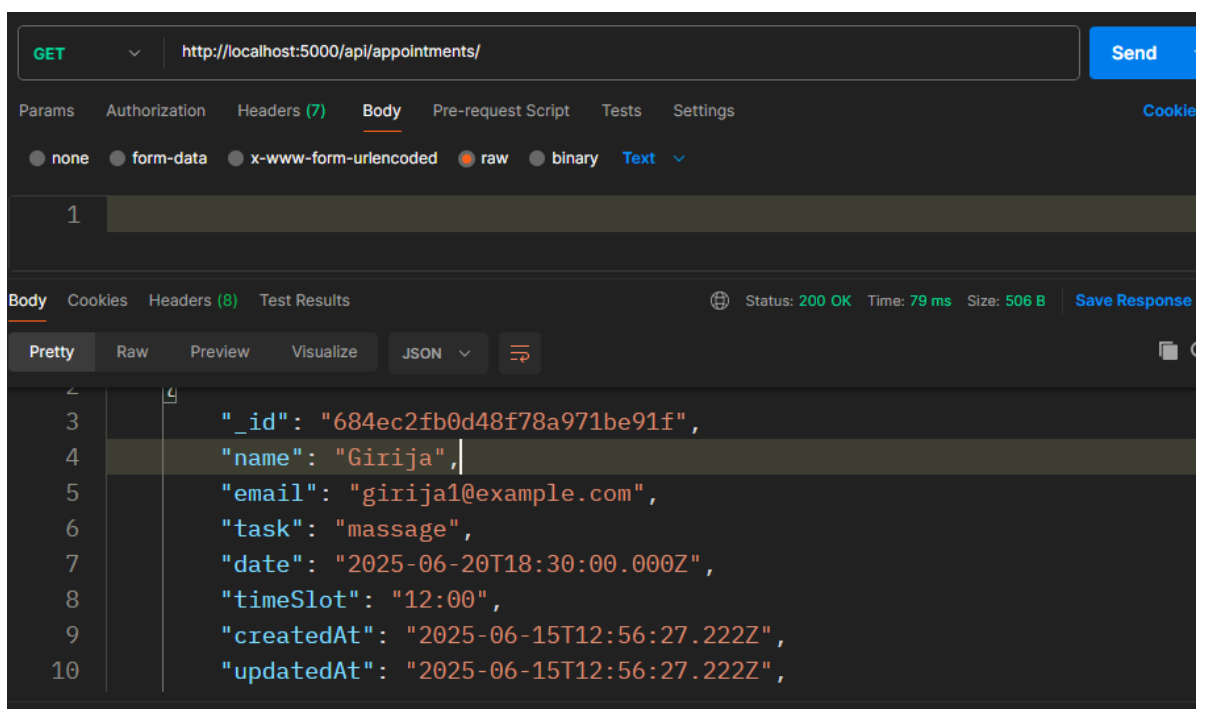


3. Book an Appointment

- **Type:** POST
- **URL:** http://localhost:5000/api/appointments



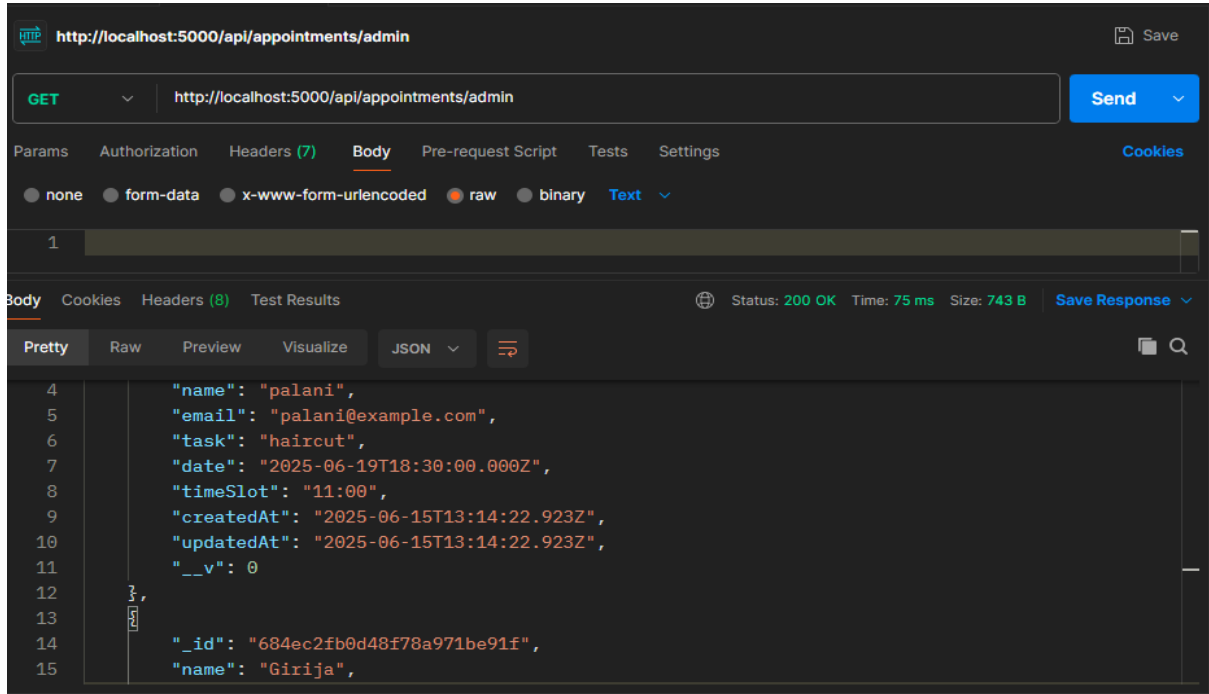
4. Get Appointment



5. Get All Appointments (Admin only)

□ **Type:** GET

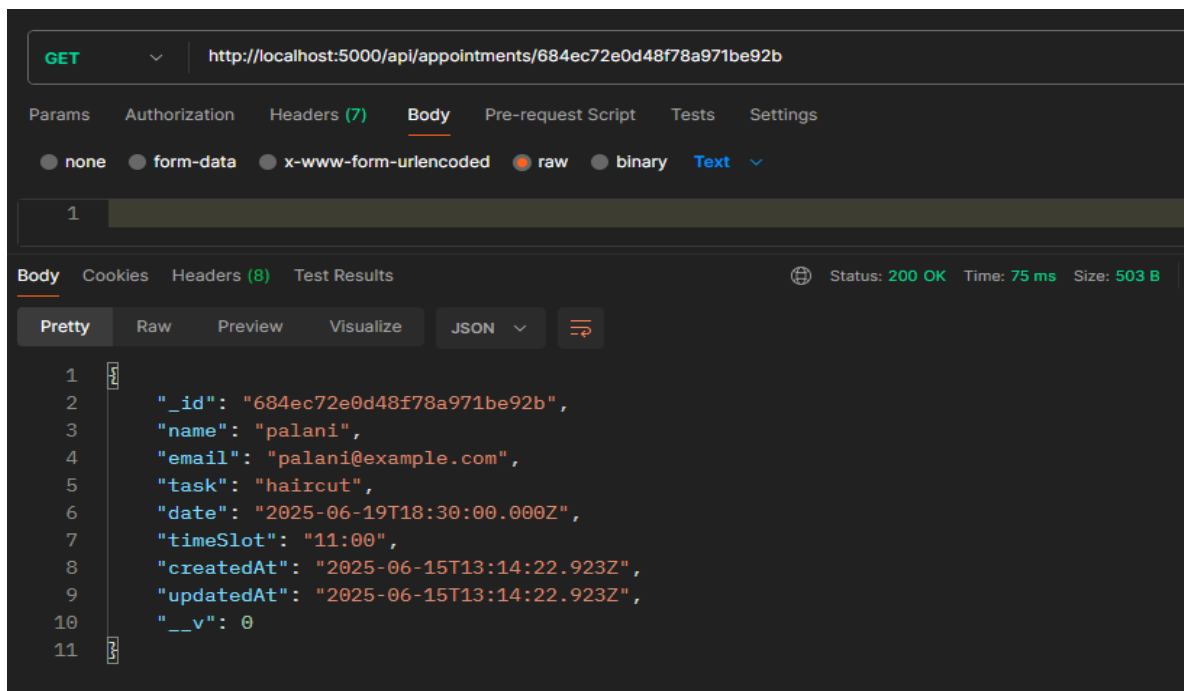
□ **URL:** http://localhost:5000/api/appointments/admin



6. Get Appointment by ID

Method: GET

URL: http://localhost:5000/appointments/<APPOINTMENT_ID>



7. Update Appointment

Method: PUT

URL: http://localhost:5000/appointments/<APPOINTMENT_ID>

The screenshot shows a REST client interface with a PUT request to `http://localhost:5000/api/appointments/684ec72e0d48f78a971be92b`. The request body is a JSON object with the following fields:

```
{  "name": "Palani",  "email": "palani@example.com",  "task": "pedicure",  "date": "2025-06-21",  "timeSlot": "09:45"}
```

The response is a JSON object with a success message and the updated appointment details:

```
{  "msg": "Appointment updated",  "updated": {    "_id": "684ec72e0d48f78a971be92b",    "name": "Palani",    "email": "palani@example.com",    "task": "pedicure",    "date": "2025-06-21T00:00:00.000Z",    "timeSlot": "09:45",  }}
```

Status: 200 OK, Time: 86 ms, Size: 545 B.

8. Delete Appointment

The screenshot shows a REST client interface with a DELETE request to `http://localhost:5000/api/appointments/684ec72e0d48f78a971be92b`. The request headers are set to `Content-Type: application/json`. The response is a JSON object with a success message:

```
{  "msg": "Appointment deleted"}
```

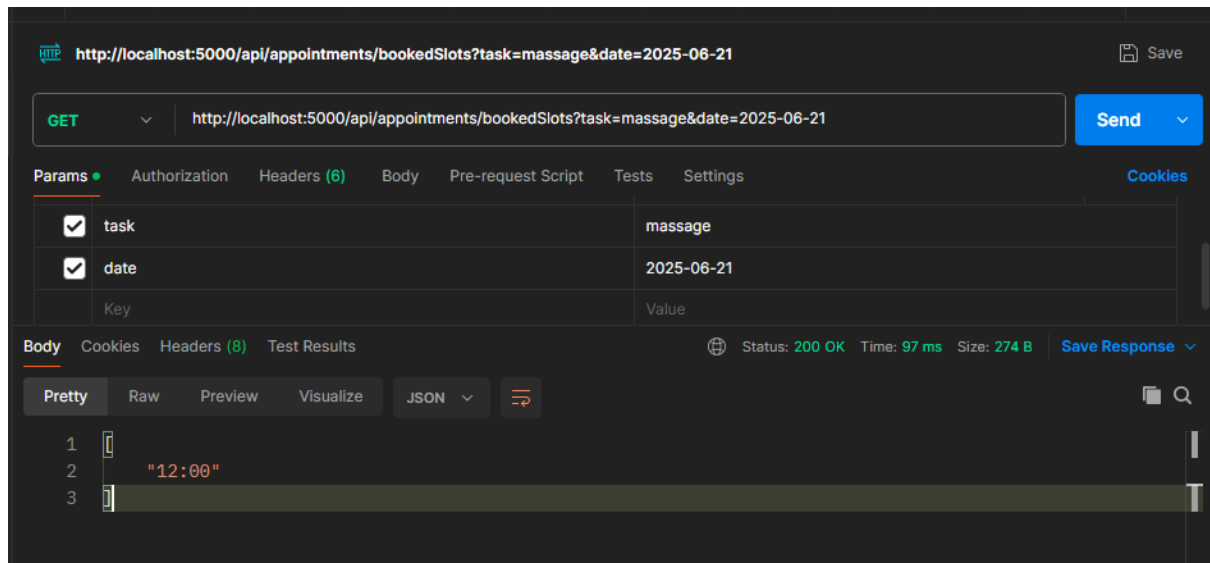
Status: 200 OK, Time: 106 ms, Size: 296 B.

9. Get Booked Time Slots for a Task and Date

Method: GET

Endpoint: <http://localhost:5000/api/appointments/bookedSlots?task=message&date=2025-06-21>

(Because the stored dates are in UTC, a booking made on 2025-06-20 UTC might correspond to 2025-06-21 in IST. Therefore, the query should use the user's local date to retrieve accurate booked slots.)



10. Postman Request for Creating Feedback & Get All Feedbacks

- **Method:** POST & GET
- **URL:** <http://localhost:5000/api/feedback>

POST `http://localhost:5000/api/feedback`

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary **JSON**

```
1 {
2   "name": "Girija",
3   "email": "girija1@example.com",
4   "message": "Great service!",
5   "rating": 5
6 }
```

Body Cookies Headers (8) Test Results Status: **201 Created** Time: **114 ms** Size: **313 B**

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "msg": "Feedback submitted successfully"
3 }
```

http://localhost:5000/api/feedback

GET `http://localhost:5000/api/feedback`

Params Authorization Headers (7) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary **JSON**

```
1
```

Body Cookies Headers (8) Test Results Status: **200 OK** Time:

Pretty Raw Preview Visualize **JSON**

```
1 {
2   {
3     "_id": "684ed4373f6eeb3ff7b18fed",
4     "name": "Girija",
5     "email": "girija1@example.com",
6     "message": "Great service!",
7     "rating": 5,
8     "createdAt": "2025-06-15T14:09:59.916Z",
9     "updatedAt": "2025-06-15T14:09:59.916Z",
10    "__v": 0
11  }
12 }
```

MONGODB ATLAS AS A DATABASE:

APPOINTMENTS FROM POSTMAN:

The screenshot shows the MongoDB Atlas interface for a project named 'Project 0'. The left sidebar contains navigation options: Overview, DATABASE, Clusters, SERVICES, Atlas Search, Stream Processing, Triggers, Migration, Data Federation, SECURITY, Quickstart, Backup, Database Access, Network Access, and Advanced. The 'Data Services' tab is active, showing a search bar for namespaces and a list of collections under the 'salon-app' namespace: 'appointments', 'feedbacks', and 'users'. The 'appointments' collection is selected, displaying its metadata: STORAGE SIZE: 36KB, LOGICAL DATA SIZE: 122KB, TOTAL DOCUMENTS: 7, and INDEXES TOTAL SIZE: 36KB. The 'Find' tab is active, showing a query filter bar with the text 'Type a query: { field: 'value' }'. Below the filter, the 'QUERY RESULTS: 1-2 OF 2' are displayed, showing a single document for 'Girija' with a task of 'massage' and a time slot of '12:00'.

```
{
  "_id": ObjectId("684ec2fb0d48f78a971be91f"),
  "name": "Girija",
  "email": "girijal@example.com",
  "task": "massage",
  "date": "2025-06-20T18:30:00.000+00:00",
  "timeSlot": "12:00",
  "createdAt": "2025-06-15T12:56:27.222+00:00",
  "updatedAt": "2025-06-15T12:56:27.222+00:00",
  "__v": 0
}
```

FEEDBACKS FROM POSTMAN:

The screenshot shows the MongoDB Atlas interface for the 'salon-app' namespace, with the 'feedbacks' collection selected. The metadata shows STORAGE SIZE: 36KB, LOGICAL DATA SIZE: 790B, TOTAL DOCUMENTS: 6, and INDEXES TOTAL SIZE: 36KB. The 'Find' tab is active, showing a query filter bar with the text 'Type a query: { field: 'value' }'. Below the filter, the 'QUERY RESULTS: 1-1 OF 1' are displayed, showing a single document for 'Girija' with a message of 'Great service!' and a rating of 5.

```
{
  "_id": ObjectId("684ed4373f6eeb3ff7b18fed"),
  "name": "Girija",
  "email": "girijal@example.com",
  "message": "Great service!",
  "rating": 5,
  "createdAt": "2025-06-15T14:09:59.916+00:00",
  "updatedAt": "2025-06-15T14:09:59.916+00:00",
  "__v": 0
}
```

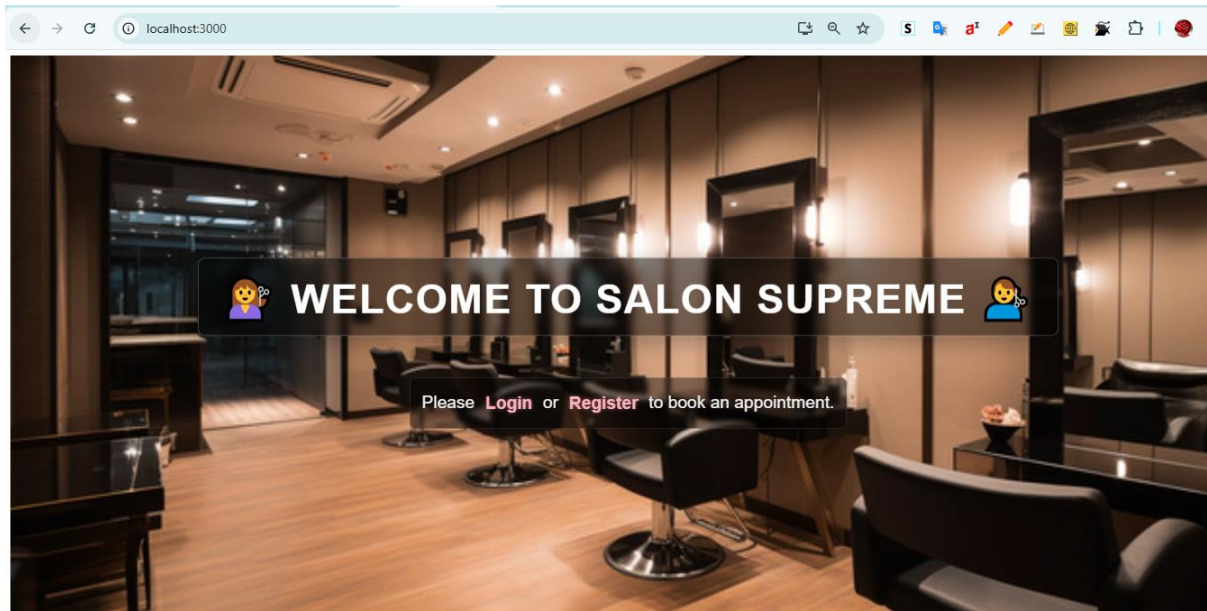
USERS FROM POSTMAN:

The screenshot shows the MongoDB Atlas interface for the 'salon-app' namespace, with the 'users' collection selected. The metadata shows STORAGE SIZE: 36KB, LOGICAL DATA SIZE: 1.6KB, TOTAL DOCUMENTS: 9, and INDEXES TOTAL SIZE: 72KB. The 'Find' tab is active, showing a query filter bar with the text 'Type a query: { field: 'value' }'. Below the filter, the 'QUERY RESULTS: 1-4 OF 4' are displayed, showing a single document for 'Admin' with a password and isAdmin set to true.

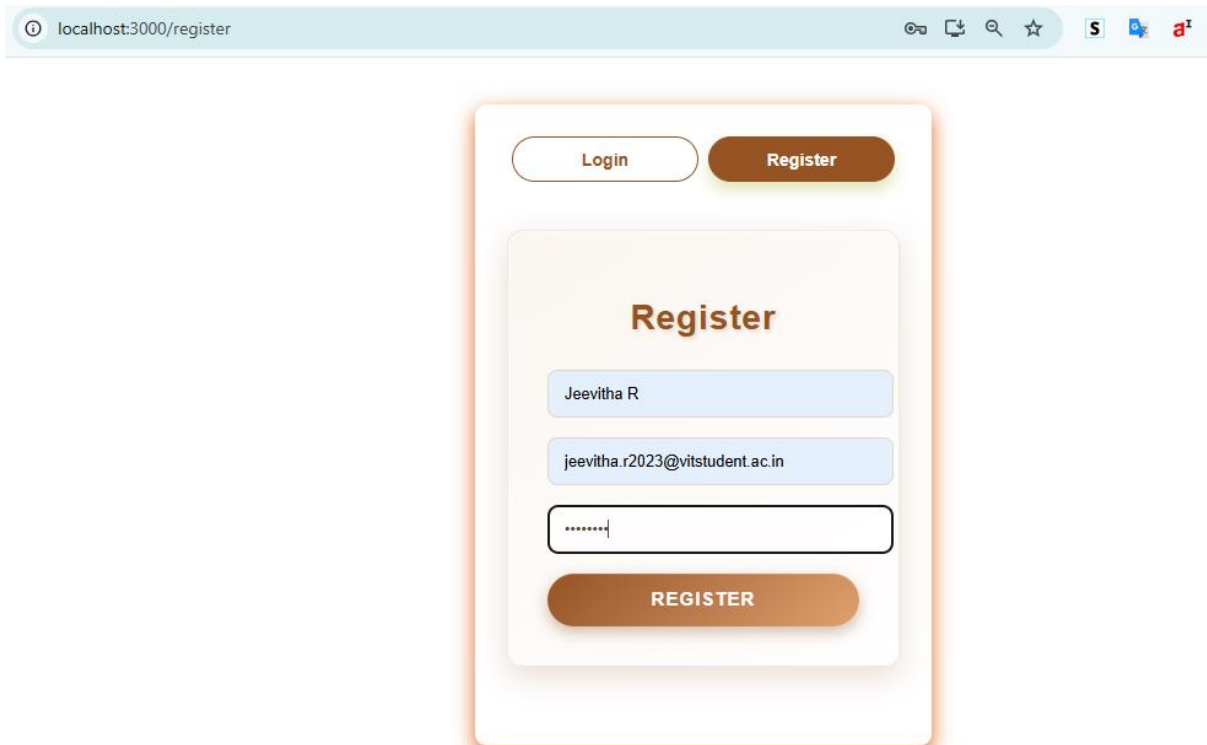
```
{
  "_id": ObjectId("684ebc2947cca3bc08715d5d"),
  "name": "Admin",
  "email": "jeevitha.r2023@vitstudent.ac.in",
  "password": "$2b$10$jn0oTakFPFuFUFrnvLucwu2RW6fZFj1awwz13H.mBo79.B04xge3a",
  "isAdmin": true,
  "createdAt": "2025-06-15T12:27:21.505+00:00",
  "updatedAt": "2025-06-15T12:27:21.505+00:00",
  "__v": 0
}
```


FRONTEND:

HOME PAGE:



REGISTER PAGE:

A screenshot of a web browser displaying the register page. The browser's address bar shows 'localhost:3000/register'. The page features a white background with a central white card that has a subtle orange glow. At the top of the card, there are two buttons: 'Login' (white with an orange border) and 'Register' (solid orange). Below these buttons, the word 'Register' is displayed in a bold, orange font. Underneath, there are three input fields: the first contains the text 'Jeevitha R', the second contains 'jeevitha.r2023@vitstudent.ac.in', and the third is a password field with masked characters '*****'. At the bottom of the card, there is a large, solid orange button with the text 'REGISTER' in white.

LOGIN PAGE:

localhost:3000/register

Login Register

Login

jeevitha.r2023@vitstudent.ac.in

LOGIN

APPOINTMENT BOOKING PAGE:

localhost:3000/book

Book an Appointment

Name

JEEVITHAR

Email

jeevitha.r2023@vitstudent.ac.in

Task

Facial

Date

28-06-2025

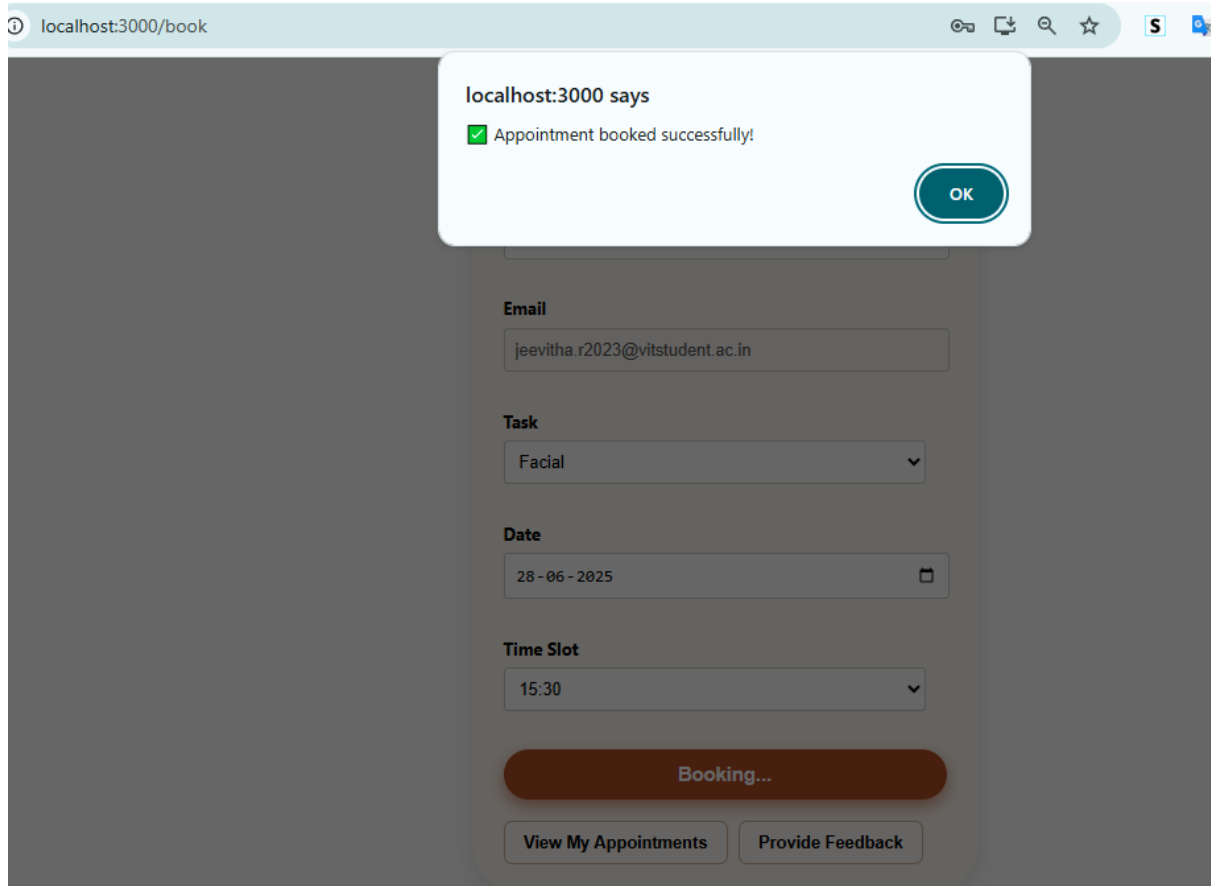
Time Slot

15:30

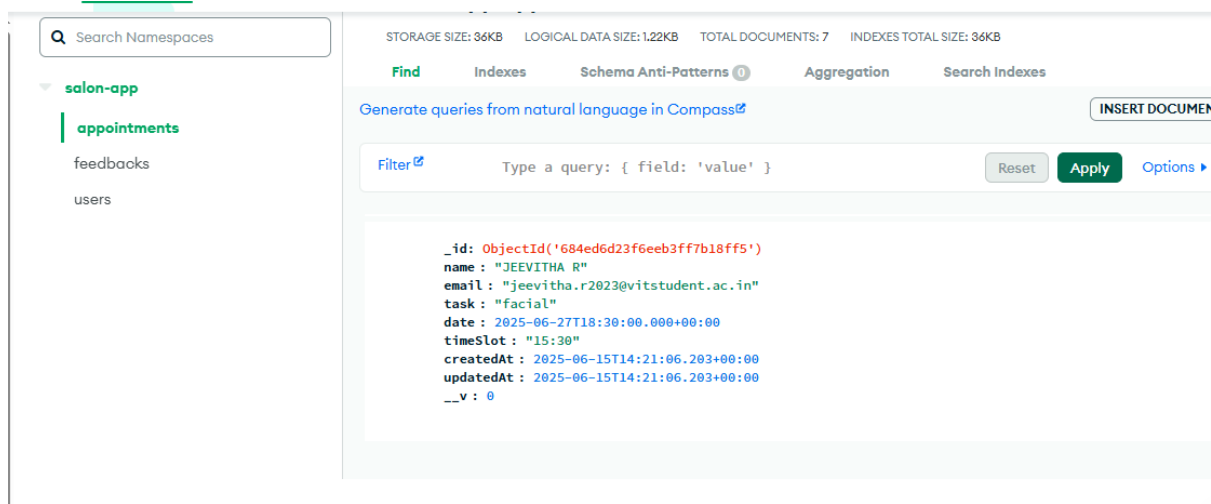
Book Appointment

View My Appointments Provide Feedback

BOOKING CONFIRMED!



STORED IN BACKEND (MONGODB ATLAS)



VIEW MY APPOINTMENTS:

❖ OPTIONS FOR USER :

- TO EDIT
- TO DELETE

Book Appointment

Hide My Appointments

Provide Feedback

Your Appointments

Task	Date	Time Slot	Actions
Facial	2025-06-27T18:30:00.000Z	15:30	<div>Edit</div> <div>Delete</div>

TO UPDATE USER APPOINTMENT (Edit) :

Hide My Appointments

Provide Feedback

Your Appointments

Task	Date	Time Slot	Actions
<input type="text" value="HairColoring"/>	<input type="text" value="dd-mm-yyyy"/>	<input type="text" value="15:30"/>	<div>Save</div> <div>Cancel</div>

localhost:3000 says

✔ Appointment updated!

OK

Hide My Appointments

Provide Feedback

Your Appointments

Task	Date	Time Slot	Actions
<input type="text" value="HairColoring"/>	<input type="text" value="dd-mm-yyyy"/>	<input type="text" value="15:30"/>	<div>Save</div> <div>Cancel</div>

UPDATED IN BACKEND:

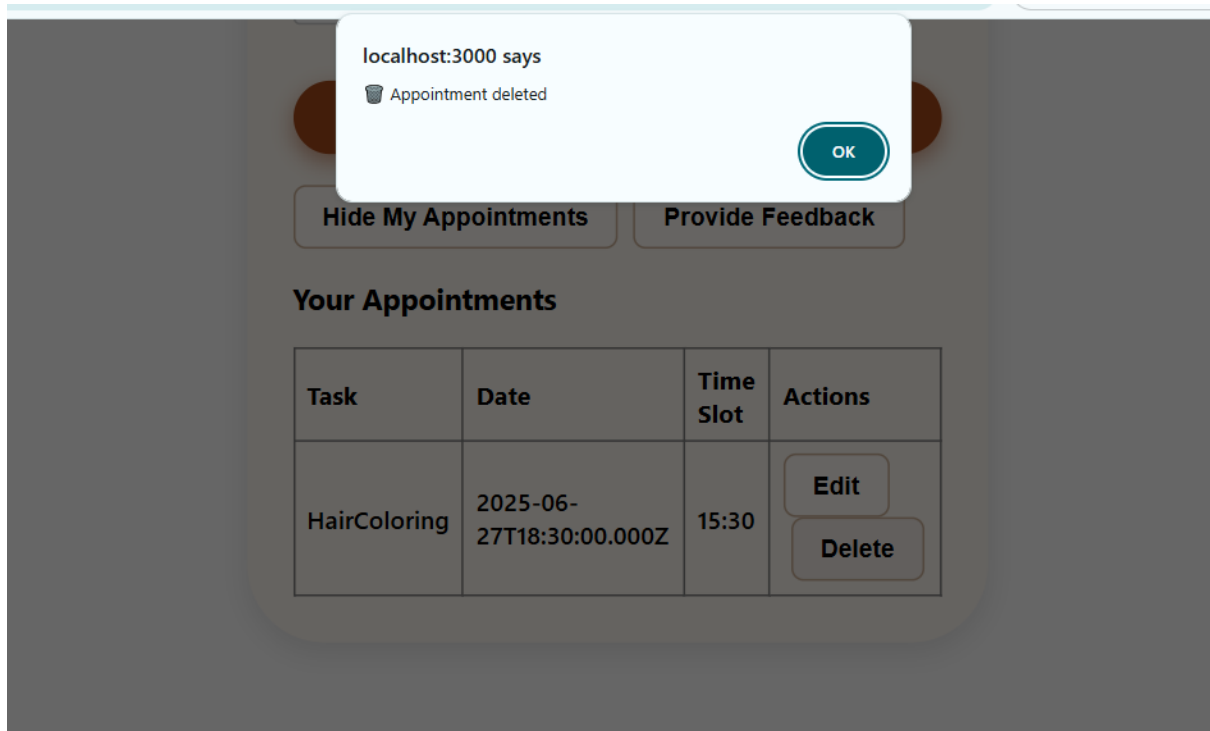
The screenshot shows the MongoDB Compass interface. On the left sidebar, the 'salon-app' database is selected, and the 'appointments' collection is highlighted. The main panel displays a document with the following fields:

```
_id: ObjectId('684ed6d23f6eeb3ff7b18ff5')
name: "JEEVITHA R"
email: "jeevitha.r2023@vitstudent.ac.in"
task: "hairColoring"
date: 2025-06-27T18:30:00.000+00:00
timeSlot: "15:30"
createdAt: 2025-06-15T14:21:06.203+00:00
updatedAt: 2025-06-15T14:27:50.097+00:00
__v: 0
```

DELETE THE APPOINTMENT (delete):

The screenshot shows a web application running on localhost:3000/book. A confirmation dialog is displayed over the page, asking: "Are you sure you want to delete this appointment?". The dialog has "OK" and "Cancel" buttons. Below the dialog, the "Your Appointments" section is visible, containing a table with one appointment:

Task	Date	Time Slot	Actions
HairColoring	2025-06-27T18:30:00.000Z	15:30	<div>Edit</div> <div>Delete</div>



DELETED IN BACKEND:

Data Services

Charts

salon-app

appointments

feedbacks

users

Find

Indexes

Schema Anti-Patterns

Aggregation

Search Indexes

Generate queries from natural language in Compass

INSERT DOCUMENT

FilterType a query: { field: 'value' }ResetApplyOptions

QUERY RESULTS: 1-2 OF 2

```
{
  "_id": ObjectId('684ec2fb0d48f78a971be91f'),
  "name": "Girija",
  "email": "girijal@example.com",
  "task": "massage",
  "date": "2025-06-20T18:30:00.000+00:00",
  "timeSlot": "12:00",
  "createdAt": "2025-06-15T12:56:27.222+00:00",
  "updatedAt": "2025-06-15T12:56:27.222+00:00",
  "__v": 0
}
```

System Status: All Good

FEEDBACK PAGE:

Select a task

Date

dd - mm - yyyy

Time Slot

Select Time Slot

Book Appointment

View My Appointments

Provide Feedback

localhost:3000/feedback

Submit Feedback

JEEVITHA RAVISHANKAR

jeevitha.r2023@vitstudent.ac.in

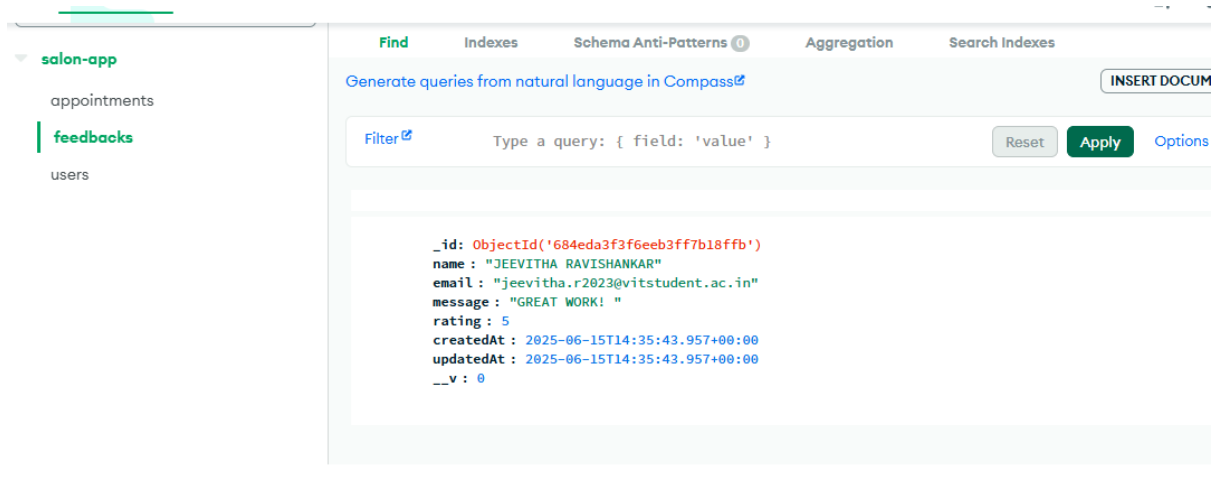
GREAT WORK!

★★★★★

Submit

THANKS FOR YOUR VALUABLE FEEDBACK AND
RATING, JEEVITHA RAVISHANKAR!

FEEDBACK STORED IN BACKED:



The screenshot shows the MongoDB Compass interface. On the left, the 'salon-app' collection is expanded, showing 'appointments', 'feedbacks', and 'users'. The 'feedbacks' collection is selected. The main area displays a document with the following fields:

```
{
  "_id": ObjectId('684eda3f3f6eeb3ff7b18ffb'),
  "name": "JEEVITHA RAVISHANKAR",
  "email": "jeevitha.r2023@vitstudent.ac.in",
  "message": "GREAT WORK! ",
  "rating": 5,
  "createdAt": 2025-06-15T14:35:43.957+00:00,
  "updatedAt": 2025-06-15T14:35:43.957+00:00,
  "__v": 0
}
```

ADMIN PAGE:

TO VIEW ALL APPOINTEMENTS AND FEEDBACK OF THE CUSTOMER:

Admin Login

Email:

jeevitha.r2023@vitstudent.ac.in

Password:

Login

Admin Dashboard

All Appointments

Customer	Email	Task	Date	Time Slot	Created At
palani	palani@example.com	haircut	6/20/2025	11:00	6/15/2025, 6:57:05 PM
Girija	girija1@example.com	massage	6/21/2025	12:00	6/15/2025, 6:26:27 PM

Customer Feedback

Email: palani@example.com

Name: Palani

Message: "Booking my appointment was so easy and quick! The time slots were clear, and I loved how I could see real-time availability. I'm super impressed with how smooth the process was. Can't wait for my haircut next week!"

Rating: ★★★★★

Submitted on 6/15/2025, 8:15:12 PM

Email: jeevitha.r2023@vitstudent.ac.in

Name: JEEVITHA RAVISHANKAR

Message: GREAT WORK!

Rating: ★★★★★

Submitted on 6/15/2025, 8:05:43 PM

Admin Dashboard

All Appointments

Customer	Email	Task	Date	Time Slot	Created At
palani	palani@example.com	haircut	6/20/2025	11:00	6/15/2025, 6:57:05 PM
Girija	girija1@example.com	massage	6/21/2025	12:00	6/15/2025, 6:26:27 PM

Customer Feedback

Email: jeevitha.r2023@vitstudent.ac.in

Name: JEEVITHA RAVISHANKAR

Message: GREAT WORK!

Rating: ★★★★★

Submitted on 6/15/2025, 8:05:43 PM

Email: girija1@example.com

Name: Girija

Message: Great service!

Rating: ★★★★★

Submitted on 6/15/2025, 7:39:59 PM