

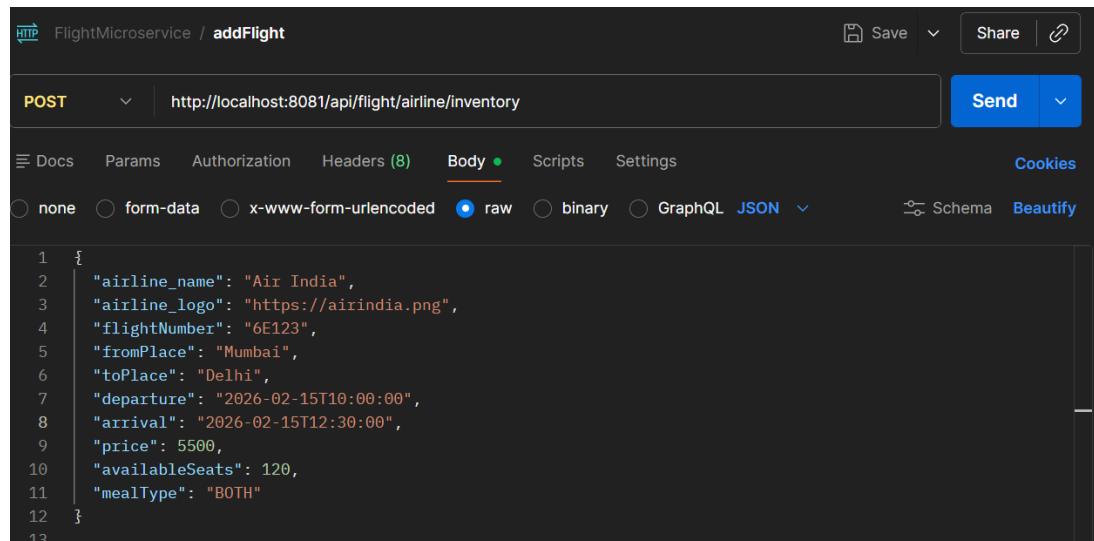
MICROSERVICES ASSIGNMENT

NEHA SHARMA

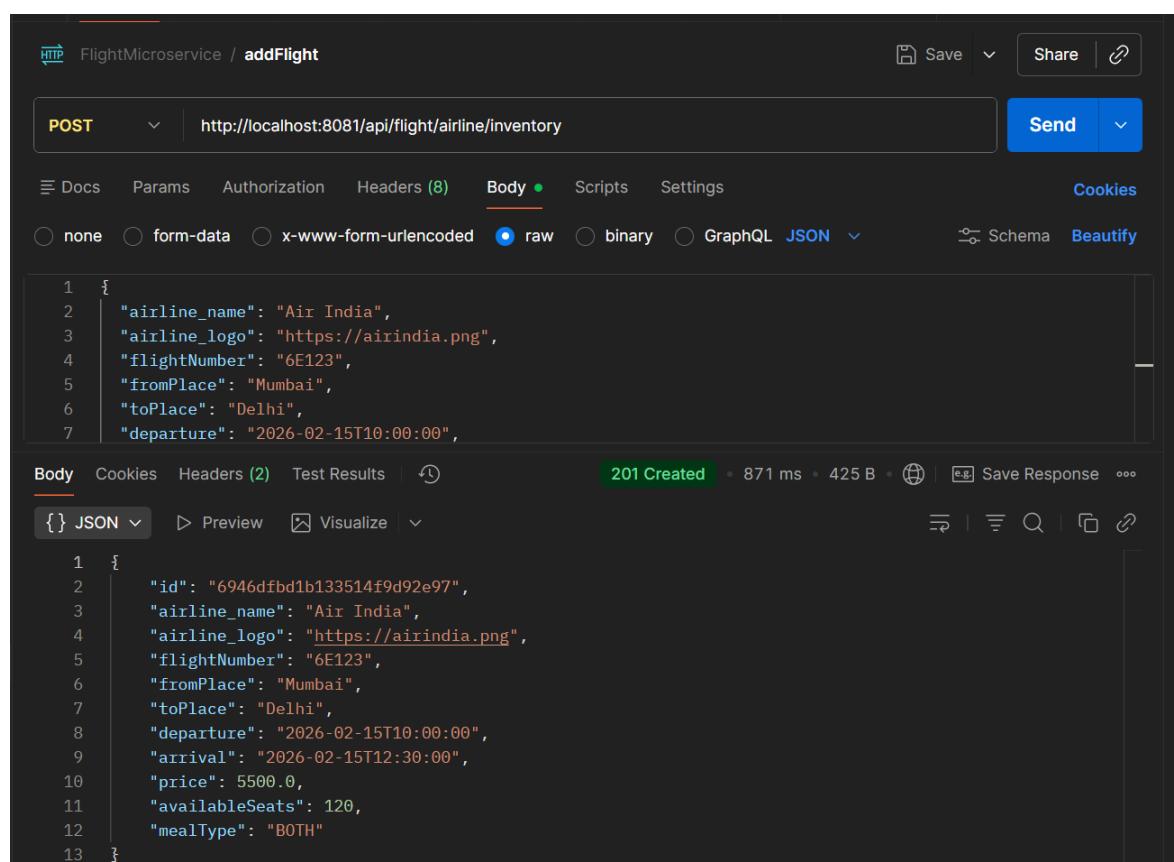
POSTMAN

1) FLIGHT_MICROSERVICE

<http://localhost:8081/api/flight/airline/inventory>



```
HTTP FlightMicroservice / addFlight
POST http://localhost:8081/api/flight/airline/inventory
Body (raw)
1 {
2     "airline_name": "Air India",
3     "airline_logo": "https://airindia.png",
4     "flightNumber": "6E123",
5     "fromPlace": "Mumbai",
6     "toPlace": "Delhi",
7     "departure": "2026-02-15T10:00:00",
8     "arrival": "2026-02-15T12:30:00",
9     "price": 5500,
10    "availableSeats": 120,
11    "mealType": "BOTH"
12 }
13 
```



```
HTTP FlightMicroservice / addFlight
POST http://localhost:8081/api/flight/airline/inventory
Body (raw)
1 {
2     "airline_name": "Air India",
3     "airline_logo": "https://airindia.png",
4     "flightNumber": "6E123",
5     "fromPlace": "Mumbai",
6     "toPlace": "Delhi",
7     "departure": "2026-02-15T10:00:00",
8
9
10
11
12
13 }
```

201 Created • 871 ms • 425 B • Save Response

{ } JSON ▾ Preview Visualize

```
1 {
2     "id": "6946dfbd1b133514f9d92e97",
3     "airline_name": "Air India",
4     "airline_logo": "https://airindia.png",
5     "flightNumber": "6E123",
6     "fromPlace": "Mumbai",
7     "toPlace": "Delhi",
8     "departure": "2026-02-15T10:00:00",
9     "arrival": "2026-02-15T12:30:00",
10    "price": 5500.0,
11    "availableSeats": 120,
12    "mealType": "BOTH"
13 }
```

HTTP FlightMicroservice / **addFlight**

POST http://localhost:8081/api/flight/airline/inventory

Body **raw**

```
1 {
2     "airline_name": "",
3     "airline_logo": "https://indigo.png",
4     "flightNumber": "6E123",
5     "fromPlace": "Mumbai",
6     "toPlace": "Delhi",
7     "departure": "2026-02-20T10:00:00",
8     "arrival": "2026-02-20T12:30:00",
9     "price": 5500,
10    "availableSeats": 120,
11    "mealType": "BOTH"
12 }
```

400 Bad Request 989 ms 301 B Save Response

{ } JSON Preview Debug with AI

```
1 {
2     "validationErrors": {
3         "airline_name": "must not be blank"
4     },
5     "error": "Validation Failed",
6     "message": "Input validation failed",
7     "timestamp": "2025-12-20T13:01:37.0879039",
8     "status": 400
9 }
```

<http://localhost:8081/api/flight/search>

HTTP FlightMicroservice / **searchFlight**

POST http://localhost:8081/api/flight/search

Body **raw**

```
1 {
2     "fromPlace": "Delhi",
3     "toPlace": "Mumbai",
4     "travelDate": "2026-02-15",
5     "returnDate": "2026-02-20",
6     "tripType": "ROUND_TRIP"
7 }
```

Body Cookies Headers (2) Test Results ⏱

200 OK • 377 ms • 1.42 KB • 🌐 | 📁 Save Response

{ } JSON ▾ ▶ Preview 🖥 Visualize ▾

```
1  {
2   "id": "694317de950e240f7686bc79",
3   "airline_name": "AIRASIA",
4   "airline_logo": "https://indigo.png",
5   "flightNumber": "6E123",
6   "fromPlace": "Mumbai",
7   "toPlace": "Delhi",
8   "departure": "2026-02-20T10:00:00",
9   "arrival": "2026-02-20T12:30:00",
10  "price": 5500.0,
11  "availableSeats": 104,
12  "mealType": "BOTH"
13 },
14 }
```

Body Cookies Headers (2) Test Results ⏱

200 OK • 377 ms • 1.42 KB • 🌐 | 📁 Save Response ⚙

{ } JSON ▾ ▶ Preview 🖥 Visualize ▾

```
41 {
42   "id": "6945af0a0d7f02024a7d82c5",
43   "airline_name": "INDIGO",
44   "airline_logo": "https://indigo.png",
45   "flightNumber": "6E123",
46   "fromPlace": "Mumbai",
47   "toPlace": "Delhi",
48   "departure": "2026-02-20T10:00:00",
49   "arrival": "2026-02-20T12:30:00",
50   "price": 5500.0,
51   "availableSeats": 120,
52   "mealType": "BOTH"
53 }
54 ]
```

HTTP FlightMicroservice / searchFlight

Save Share ⚙

POST ▾ | http://localhost:8081/api/flight/search

Send ⚙

Docs Params Authorization Headers (8) Body ● Scripts Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON ⚙ Schema Beautify

```
1 {
2   "fromPlace": "Delhi",
3   "toPlace": "Mumbai",
4   "travelDate": "2025-02-15",
5   "returnDate": "2025-02-16",
6   "tripType": "ROUND_TRIP"
7 }
```

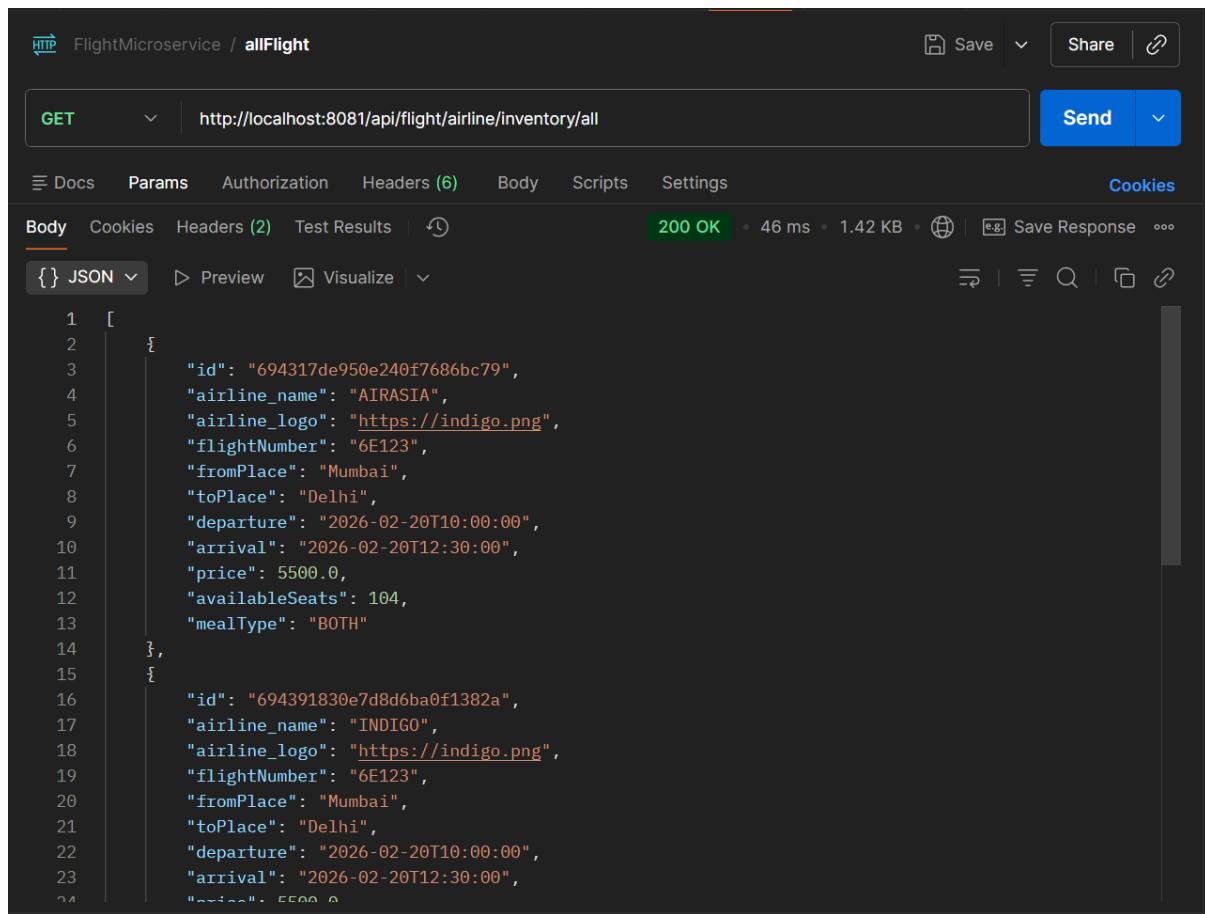
Body Cookies Headers (2) Test Results ⏱

400 Bad Request • 17 ms • 315 B • 🌐 | 📁 Save Response ⚙

{ } JSON ▾ ▶ Preview 🗃 Debug with AI ▾

```
1 {
2   "validationErrors": {
3     "travelDate": "Travel date must be in the future"
4   },
5   "error": "Validation Failed",
6   "message": "Input validation failed",
7   "timestamp": "2025-12-20T13:03:25.2404484",
8   "status": 400
9 }
```

<http://localhost:8081/api/flight/airline/inventory/all>

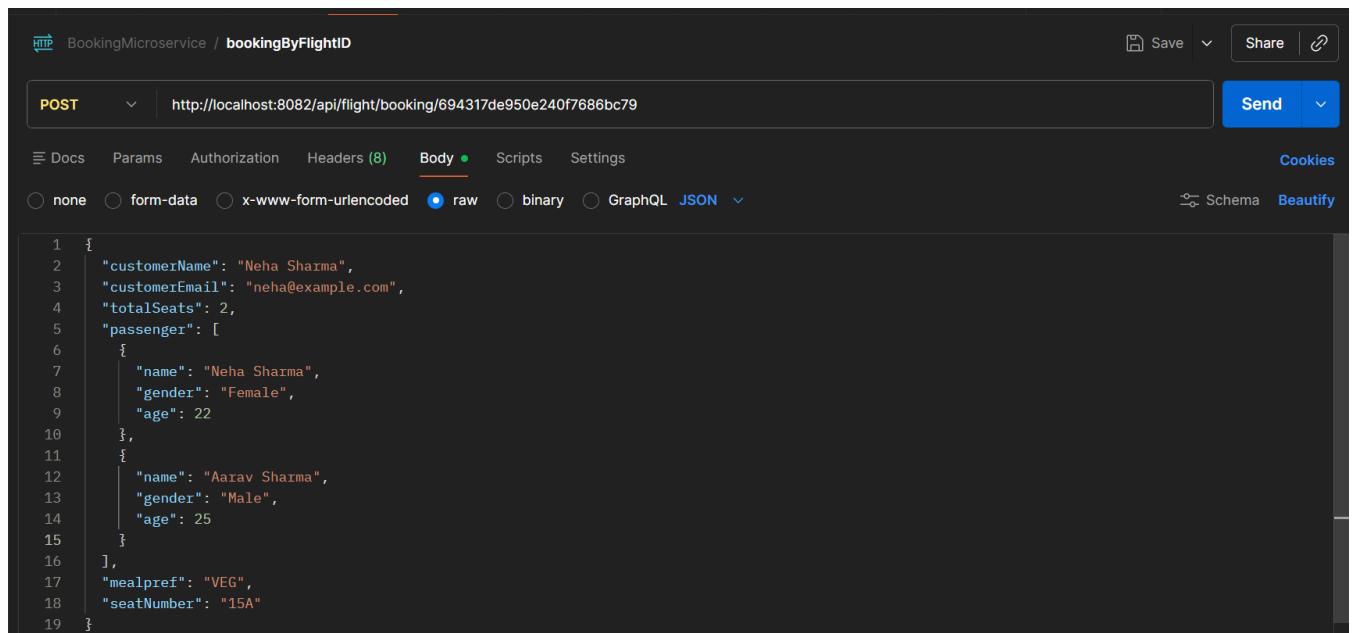


The screenshot shows the Postman application interface. At the top, it says "FlightMicroservice / allFlight". Below that is a search bar with "GET" and the URL "http://localhost:8081/api/flight/airline/inventory/all". To the right are "Save", "Share", and a copy icon. Underneath the search bar are tabs for "Docs", "Params", "Authorization", "Headers (6)", "Body", "Scripts", and "Settings". The "Body" tab is selected, showing "200 OK" status, "46 ms" response time, "1.42 KB" size, and a "Save Response" button. Below the status are buttons for "Preview" and "Visualize". The main area displays the JSON response:

```
1 [  
2   {  
3     "id": "694317de950e240f7686bc79",  
4     "airline_name": "AIRASIA",  
5     "airline_logo": "https://indigo.png",  
6     "flightNumber": "6E123",  
7     "fromPlace": "Mumbai",  
8     "toPlace": "Delhi",  
9     "departure": "2026-02-20T10:00:00",  
10    "arrival": "2026-02-20T12:30:00",  
11    "price": 5500.0,  
12    "availableSeats": 104,  
13    "mealType": "BOTH"  
14  },  
15  {  
16    "id": "694391830e7d8d6ba0f1382a",  
17    "airline_name": "INDIGO",  
18    "airline_logo": "https://indigo.png",  
19    "flightNumber": "6E123",  
20    "fromPlace": "Mumbai",  
21    "toPlace": "Delhi",  
22    "departure": "2026-02-20T10:00:00",  
23    "arrival": "2026-02-20T12:30:00",  
24    "price": 5500.0  
]
```

2) BOOKING MICROSERVICE

<http://localhost:8082/api/flight/booking/694317de950e240f7686bc79>



The screenshot shows the Postman application interface. At the top, it says "BookingMicroservice / bookingByFlightID". Below that is a search bar with "POST" and the URL "http://localhost:8082/api/flight/booking/694317de950e240f7686bc79". To the right are "Save", "Share", and a copy icon. Underneath the search bar are tabs for "Docs", "Params", "Authorization", "Headers (8)", "Body", "Scripts", and "Settings". The "Body" tab is selected, showing "200 OK" status, "46 ms" response time, "1.42 KB" size, and a "Save Response" button. Below the status are buttons for "Schema" and "Beautify". The main area displays the JSON response:

```
1 {  
2   "customerName": "Neha Sharma",  
3   "customerEmail": "neha@example.com",  
4   "totalSeats": 2,  
5   "passenger": [  
6     {  
7       "name": "Neha Sharma",  
8       "gender": "Female",  
9       "age": 22  
10      },  
11      {  
12        "name": "Aarav Sharma",  
13        "gender": "Male",  
14        "age": 25  
15      }  
16    ],  
17    "mealpref": "VEG",  
18    "seatNumber": "15A"  
19 }
```

OUTPUT:

Body Cookies Headers (5) Test Results | ⏱

201 Created • 1.50 s • 665 B • ⓘ Save Response ⚙️

{ } JSON ▾ ▶ Preview ⌂ Visualize ▾

```
1 {  
2   "pnr": "42272408",  
3   "customerName": "Neha Sharma",  
4   "customerEmail": "neha@example.com",  
5   "flightId": "694317de950e240f7686bc79",  
6   "numSeats": 2,  
7   "passenger": [  
8     {  
9       "id": null,  
10      "name": "Neha Sharma",  
11      "gender": "Female",  
12      "age": 22  
13    },  
14    {  
15      "id": null,  
16      "name": "Aarav Sharma",  
17      "gender": "Male",  
18      "age": 25  
19    }  
20  ]  
1.
```

Body Cookies Headers (5) Test Results | ⏱

201 Created • 1.50 s • 665 B • ⓘ Save Response ⚙️

{ } JSON ▾ ▶ Preview ⌂ Visualize ▾

```
8   "passenger": [  
9     {  
10       "id": null,  
11       "name": "Neha Sharma",  
12       "gender": "Female",  
13       "age": 22  
14     },  
15     {  
16       "id": null,  
17       "name": "Aarav Sharma",  
18       "gender": "Male",  
19       "age": 25  
20     }  
21   ],  
22   "mealpref": "VEG",  
23   "seatNumber": "15A",  
24   "bookingDate": "2025-12-20T12:54:27",  
25   "status": "CONFIRMED"  
26 }
```

<http://localhost:8082/api/flight/ticket/B3ECE00C>

HTTP BookingMicroservice / ticketByPNR

GET http://localhost:8082/api/flight/ticket/B3ECE00C Send

Docs Params Authorization Headers (6) Body Scripts Settings Cookies

Body Cookies Headers (5) Test Results ⏪ 200 OK · 656 ms · 660 B · 🌐 e.g. Save Response ⋮

{ } JSON ▾ ▷ Preview ⚡ Visualize ▾

```
1 {
2     "pnr": "B3ECE00C",
3     "customerName": "Neha Sharma",
4     "customerEmail": "neha@example.com",
5     "flightId": "694317de950e240f7686bc79",
6     "numSeats": 2,
7     "passenger": [
8         {
9             "id": null,
10            "name": "Neha Sharma",
11            "gender": "Female",
12            "age": 22
13        },
14        {
15            "id": null,
16            "name": "Aarav Sharma",
17            "gender": "Male",
18            "age": 25
19        }
20    ],
21    "mealpref": "VEG",
22    "seatNumber": "15A",
23    "bookingDate": "2025-12-19T05:42:16",
24    "status": "CONFIRMED"
25 }
```

HTTP BookingMicroservice / ticketByPNR

GET http://localhost:8082/api/flight/ticket/B3ECE00B Send

Docs Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description	...	

Body Cookies Headers (5) Test Results ⏪ 404 Not Found · 481 ms · 206 B · 🌐 e.g. Save Response ⋮

Raw ▾ ▷ Preview ⚡ Debug with AI ▾

```
1 Ticket not found with PNR: B3ECE00B
```

<http://localhost:8082/api/flight/booking/history/neha@gmail.com>

The screenshot shows a Postman request for `http://localhost:8082/api/flight/booking/history/neha@gmail.com`. The response is a 200 OK status with a JSON payload containing booking details for two passengers.

```
[{"id": "CED9C1F4", "customerName": "Neha Sharma", "customerEmail": "neha@gmail.com", "flightId": "694317de950e240f7686bc79", "numSeats": 2, "passenger": [{"id": null, "name": "Neha Sharma", "gender": "Female", "age": 22}, {"id": null, "name": "Aazav Sharma", "gender": "Male", "age": 25}], "mealpref": "VEG", "seatNumber": "15A", "bookingDate": "2025-12-20T14:37:21", "status": "CONFIRMED"}]
```

The screenshot shows a Postman request for `http://localhost:8082/api/flight/booking/history/arnav@gmail.com`. The response is a 404 Not Found status with a message indicating no booking history was found for the specified email.

This request does not have a body

```
1 No booking history found for email: arnav@gmail.com
```

<http://localhost:8082/api/flight/booking/cancel/7987240E>

The screenshot shows a Postman request for the endpoint `http://localhost:8082/api/flight/booking/cancel/7987240E`. The status bar indicates a **200 OK** response with a duration of 845 ms and a body size of 205 B. The response body contains the message: **1 Ticket cancelled and removed successfully**.

The screenshot shows a Postman request for the endpoint `http://localhost:8082/api/flight/booking/cancel/7987240E`. The status bar indicates a **404 Not Found** response with a duration of 15 ms and a body size of 206 B. The response body contains the message: **1 Ticket not found with PNR: 7987240E**.

3) API_GATEWAY

<http://localhost:8091/microservice-flightservice/api/flight/airline/inventory>

The screenshot shows the Postman interface with a POST request to <http://localhost:8091/microservice-flightservice/api/flight/airline/inventory>. The request body is a JSON object representing a flight:

```
1 {
2     "airline_name": "INDIGO",
3     "airline_logo": "https://indigo.png",
4     "flightNumber": "6E123",
5     "fromPlace": "Chennai",
6     "toPlace": "Delhi",
7     "departure": "2026-02-20T10:00:00",
8     "arrival": "2026-02-20T12:30:00",
9     "price": 5500,
10    "availableSeats": 120,
11    "mealType": "BOTH"
12 }
```

The screenshot shows the Postman interface after sending the POST request. The response status is 200 OK, with a response time of 31 ms and a response size of 416 B. The response body is identical to the request body, indicating a successful creation of the flight record:

```
1 {
2     "id": "69468dd155f11835dc0a3195",
3     "airline_name": "INDIGO",
4     "airline_logo": "https://indigo.png",
5     "flightNumber": "6E123",
6     "fromPlace": "Chennai",
7     "toPlace": "Delhi",
8     "departure": "2026-02-20T10:00:00",
9     "arrival": "2026-02-20T12:30:00",
10    "price": 5500.0,
11    "availableSeats": 120,
12    "mealType": "BOTH"
13 }
```

<http://localhost:8091/microservice-flightservice/api/flight/search>

The screenshot shows the API Gateway interface with the following details:

- HTTP API:** APIGateway / searchFlight
- Method:** POST
- URL:** http://localhost:8091/microservice-flightservice/api/flight/search
- Body:** Raw JSON input (selected):

```
1 {  
2   "fromPlace": "Delhi",  
3   "toPlace": "Chennai",  
4   "travelDate": "2026-02-15",  
5   "returnDate": "2026-02-20",  
6   "tripType": "ROUND_TRIP"  
7 }
```

- Headers:** (8)
- Body:** (selected) raw JSON
- Test Results:** 200 OK, 50 ms, 770 B
- Body Response:** JSON (selected) showing two flight records:

```
1 [  
2   {  
3     "id": "69468e0f55f11835dc0a3196",  
4     "airline_name": "INDIGO",  
5     "airline_logo": "https://indigo.png",  
6     "flightNumber": "6E123",  
7     "fromPlace": "Delhi",  
8     "toPlace": "Chennai",  
9     "departure": "2026-02-15T10:00:00",  
10    "arrival": "2026-02-15T12:30:00",  
11    "price": 5500.0,  
12    "availableSeats": 120,  
13    "mealType": "BOTH"  
14  },  
15  {  
16    "id": "69468dd155f11835dc0a3195",  
17    "airline_name": "INDIGO",  
18    "airline_logo": "https://indigo.png",  
19    "flightNumber": "6E123",  
20    "fromPlace": "Chennai",  
21    "toPlace": "Delhi",  
22    "departure": "2026-02-20T10:00:00",  
23    "arrival": "2026-02-20T12:30:00",  
24    "price": 5500.0,  
25    "availableSeats": 120,  
26    "mealType": "BOTH"  
27  }]  
28 ]
```

HTTP APIGateway / **searchFlight2**

Save Share ⌂

POST http://localhost:8091/microservice-flightservice/api/flight/search Send

Docs Params Authorization Headers (8) **Body** • Scripts Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Schema Beautify

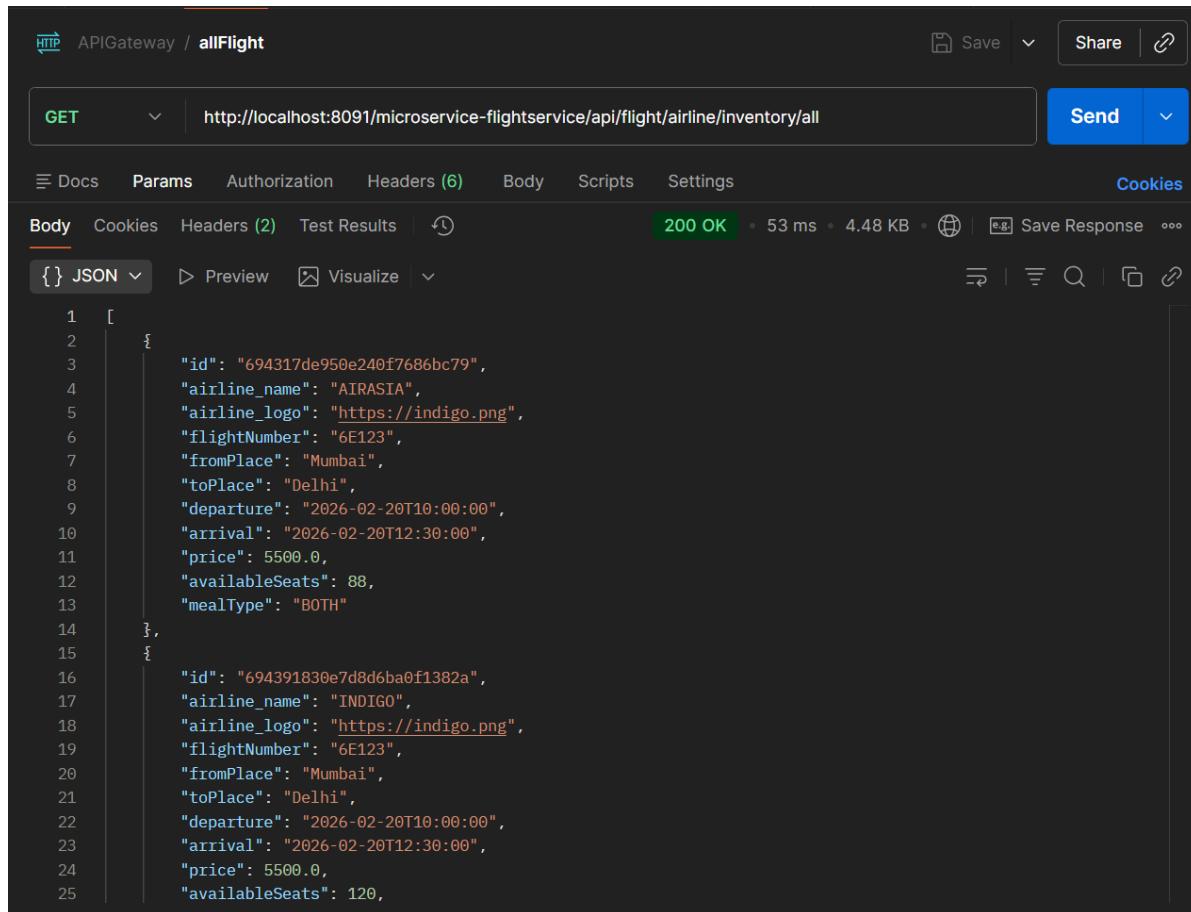
```
1 {  
2   "fromPlace": "Chennai",  
3   "toPlace": "Delhi",  
4   "travelDate": "2026-02-20",  
5   "tripType": "ONE WAY"  
6 }
```

Body Cookies Headers (2) Test Results ⌂ 200 OK • 36 ms • 425 B • Save Response ⌂

{ } JSON ▾ ▶ Preview ⌂ Visualize ▾

```
1 [  
2   {  
3     "id": "69468dd155f11835dc0a3195",  
4     "airline_name": "INDIGO",  
5     "airline_logo": "https://indigo.png",  
6     "flightNumber": "6E123",  
7     "fromPlace": "Chennai",  
8     "toPlace": "Delhi",  
9     "departure": "2026-02-20T10:00:00",  
10    "arrival": "2026-02-20T12:30:00",  
11    "price": 5500.0,  
12    "availableSeats": 120,  
13    "mealType": "BOTH"  
14  }  
15 ]
```

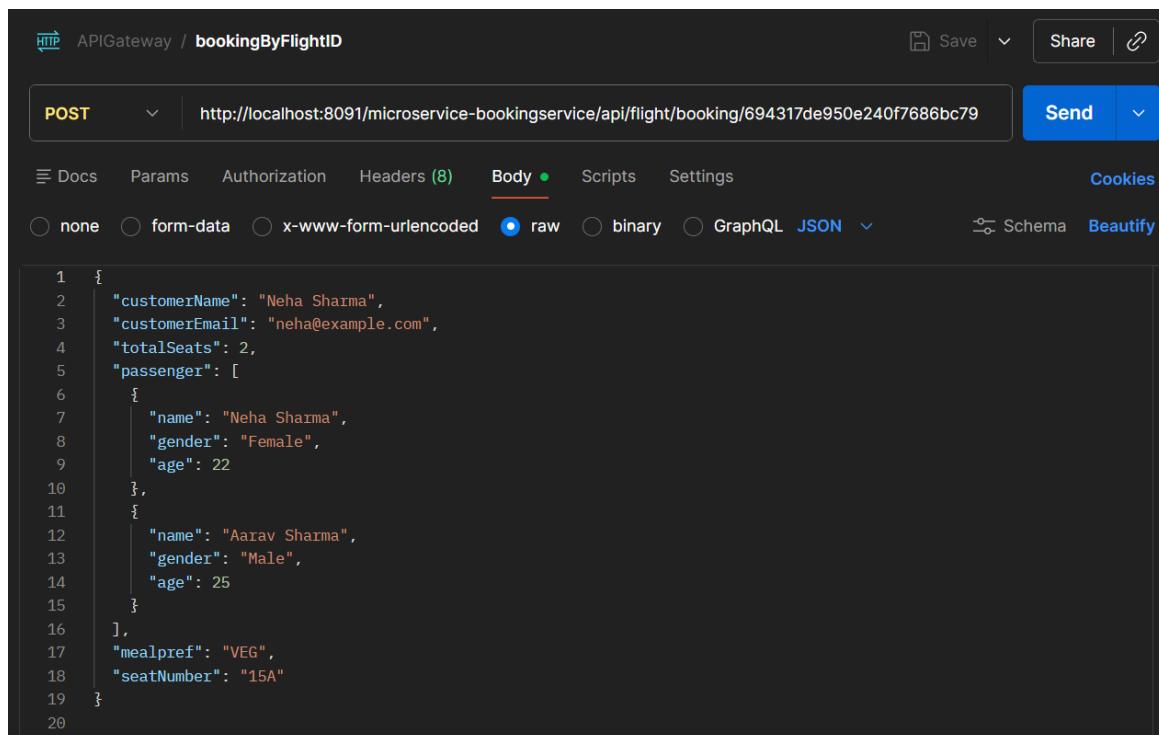
<http://localhost:8091/microservice-flightservice/api/flight/airline/inventory/all>



The screenshot shows the API Gateway interface with a successful GET request to `http://localhost:8091/microservice-flightservice/api/flight/airline/inventory/all`. The response status is 200 OK, with a response time of 53 ms and a size of 4.48 KB. The response body is a JSON array containing two flight records:

```
1 [  
2 {  
3     "id": "694317de950e240f7686bc79",  
4     "airline_name": "AIRASIA",  
5     "airline_logo": "https://indigo.png",  
6     "flightNumber": "6E123",  
7     "fromPlace": "Mumbai",  
8     "toPlace": "Delhi",  
9     "departure": "2026-02-20T10:00:00",  
10    "arrival": "2026-02-20T12:30:00",  
11    "price": 5500.0,  
12    "availableSeats": 88,  
13    "mealType": "BOTH"  
14 },  
15 {  
16     "id": "694391830e7d8d6ba0f1382a",  
17     "airline_name": "INDIGO",  
18     "airline_logo": "https://indigo.png",  
19     "flightNumber": "6E123",  
20     "fromPlace": "Mumbai",  
21     "toPlace": "Delhi",  
22     "departure": "2026-02-20T10:00:00",  
23     "arrival": "2026-02-20T12:30:00",  
24     "price": 5500.0,  
25     "availableSeats": 120,  
26 }
```

<http://localhost:8091/microservice-bookingservice/api/flight/booking/694317de950e240f7686bc79>



The screenshot shows the API Gateway interface with a POST request to `http://localhost:8091/microservice-bookingservice/api/flight/booking/694317de950e240f7686bc79`. The request body is a JSON object representing a booking for flight ID `694317de950e240f7686bc79`:

```
1 {  
2     "customerName": "Neha Sharma",  
3     "customerEmail": "neha@example.com",  
4     "totalSeats": 2,  
5     "passenger": [  
6         {  
7             "name": "Neha Sharma",  
8             "gender": "Female",  
9             "age": 22  
10        },  
11        {  
12            "name": "Aarav Sharma",  
13            "gender": "Male",  
14            "age": 25  
15        }  
16    ],  
17    "mealpref": "VEG",  
18    "seatNumber": "15A"  
19 }
```

Body 201 Created • 136 ms • 617 B • e.g.

{ } JSON Preview Visualize

```
1  {
2      "pnr": "3FCEBD12",
3      "customerName": "Neha Sharma",
4      "customerEmail": "neha@example.com",
5      "flightId": "694317de950e240f7686bc79",
6      "numSeats": 2,
7      "passenger": [
8          {
9              "id": null,
10             "name": "Neha Sharma",
11             "gender": "Female",
12             "age": 22
13         },
14         {
15             "id": null,
16             "name": "Aarav Sharma",
17             "gender": "Male",
18             "age": 25
19         }
20     ],
21     "mealpref": "VEG",
22     "seatNumber": "15A",
23     "bookingDate": "2025-12-20T17:28:36",
24     "status": "CONFIRMED"
25 }
```

<http://localhost:8091/microservice-bookingservice/api/flight/ticket/3FCEBD12>

The screenshot shows the API Gateway interface with the following details:

- Method:** GET
- URL:** http://localhost:8091/microservice-bookingservice/api/flight/ticket/3FCEBD12
- Headers:** (8)
- Body:** (JSON) - Displays the response body as JSON. The JSON object contains information about a flight ticket, including passenger details, meal preference, seat number, booking date, and status.
- Response Headers:** 200 OK, 86 ms, 612 B, Save Response

```
1  {
2   "pnr": "3FCEBD12",
3   "customerName": "Neha Sharma",
4   "customerEmail": "neha@example.com",
5   "flightId": "694317de950e240f7686bc79",
6   "numSeats": 2,
7   "passenger": [
8     {
9       "id": null,
10      "name": "Neha Sharma",
11      "gender": "Female",
12      "age": 22
13    },
14    {
15      "id": null,
16      "name": "Aarav Sharma",
17      "gender": "Male",
18      "age": 25
19    }
20  ],
21  "mealpref": "VEG",
22  "seatNumber": "15A",
23  "bookingDate": "2025-12-20T17:28:36",
24  "status": "CONFIRMED"
25 }
```

<http://localhost:8091/microservice-booking-service/api/flight/booking/history/rmneha@gmail.com>

The screenshot shows the API Gateway interface with a successful GET request. The URL is <http://localhost:8091/microservice-booking-service/api/flight/booking/history/rmneha@gmail.com>. The response status is 200 OK, with a response time of 21 ms and a size of 660 B. The response body is a JSON object representing a flight booking history entry:

```
1 [  
2 {  
3   "pnr": "11AC5FEB",  
4   "customerName": "Neha Sharma",  
5   "customerEmail": "rmneha@gmail.com",  
6   "flightId": "694317de950e240f7686bc79",  
7   "numSeats": 2,  
8   "passenger": [  
9     {  
10       "id": null,  
11       "name": "Neha Sharma",  
12       "gender": "Female",  
13       "age": 22  
14     },  
15     {  
16       "id": null,  
17       "name": "Aarav Sharma",  
18       "gender": "Male",  
19       "age": 25  
20     }  
21   ],  
22   "mealpref": "VEG",  
23   "seatNumber": "15A",  
24   "bookingDate": "2025-12-20T17:33:26",  
25   "status": "CONFIRMED"  
26 }]  
27 ]
```

<http://localhost:8091/microservice-booking-service/api/flight/booking/cancel/11AC5FEB>

The screenshot shows the API Gateway interface with a successful DELETE request. The URL is <http://localhost:8091/microservice-booking-service/api/flight/booking/cancel/11AC5FEB>. The response status is 200 OK, with a response time of 116 ms and a size of 15. The response body is a simple message indicating the ticket was cancelled successfully:

```
1 Ticket cancelled and removed successfully
```

4) NewMan (POSTMAN using CLI)

```
C:\Users\Neha Sharma>newman run "C:\Users\Neha Sharma\Desktop\FlightMicroservice.postman_collection.json"
newman

FlightMicroservice

→ addFlight
  POST http://localhost:8081/api/flight/airline/inventory [200 OK, 415B, 40ms]

→ searchFlight
  POST http://localhost:8081/api/flight/search [200 OK, 3.19kB, 13ms]

→ allFlight
  GET http://localhost:8081/api/flight/airline/inventory/all [200 OK, 5.27kB, 16ms]



|                        | executed                                | failed |
|------------------------|-----------------------------------------|--------|
| iterations             | 1                                       | 0      |
| requests               | 3                                       | 0      |
| test-scripts           | 0                                       | 0      |
| prerequest-scripts     | 0                                       | 0      |
| assertions             | 0                                       | 0      |
| total run duration:    | 325ms                                   |        |
| total data received:   | 8.64kB (approx)                         |        |
| average response time: | 23ms [min: 13ms, max: 40ms, s.d.: 12ms] |        |


```

```
C:\Users\Neha Sharma>
```

```
C:\Users\Neha Sharma>newman run "C:\Users\Neha Sharma\Desktop\BookingMicroservice.postman_collection.json"
newman

BookingMicroservice

→ bookingByFlightID
  POST http://localhost:8082/api/flight/booking/694317de950e240f7686bc79 [201 Created, 665B, 59ms]

→ ticketByPNR
  GET http://localhost:8082/api/flight/ticket/DC6EA884 [200 OK, 658B, 20ms]

→ ticketByEmail
  GET http://localhost:8082/api/flight/booking/history/arnav@example.com [404 Not Found, 268B, 19ms]

→ cancelTicket
  DELETE http://localhost:8082/api/flight/booking/cancel/F174B495 [404 Not Found, 206B, 16ms]



|                        | executed                                | failed |
|------------------------|-----------------------------------------|--------|
| iterations             | 1                                       | 0      |
| requests               | 4                                       | 0      |
| test-scripts           | 0                                       | 0      |
| prerequest-scripts     | 0                                       | 0      |
| assertions             | 0                                       | 0      |
| total run duration:    | 407ms                                   |        |
| total data received:   | 1.08kB (approx)                         |        |
| average response time: | 28ms [min: 16ms, max: 59ms, s.d.: 17ms] |        |


```

```
C:\Users\Neha Sharma>newman run "C:\Users\Neha Sharma\Desktop\APIGateway.postman_collection.json"
newman

APIGateway

→ addFlight
  POST http://localhost:8091/microservice-flightservice/api/flight/airline/inventory [200 OK, 415B, 64ms]

→ searchFlight
  POST http://localhost:8091/microservice-flightservice/api/flight/search [200 OK, 3.53kB, 31ms]

→ searchFlight2
  POST http://localhost:8091/microservice-flightservice/api/flight/search [200 OK, 3.53kB, 26ms]

→ allFlight
  GET http://localhost:8091/microservice-flightservice/api/flight/airline/inventory/all [200 OK, 5.62kB, 23ms]

→ bookingByFlightID
  POST http://localhost:8091/microservice-bookingservice/api/flight/booking/694317de950e240f7686bc79 [201 Created, 617B, 78ms]

→ ticketbyPNR
  GET http://localhost:8091/microservice-bookingservice/api/flight/ticket/B3ECE00C [404 Not Found, 158B, 27ms]

→ ticketByEmail
  GET http://localhost:8091/microservice-bookingservice/api/flight/booking/history/rmnneha@gmail.com [404 Not Found, 219B, 27ms]

→ cancelTicket
  DELETE http://localhost:8091/microservice-bookingservice/api/flight/booking/cancel/11AC5FEB [404 Not Found, 158B, 24ms]



|            | executed | failed |
|------------|----------|--------|
| iterations | 1        | 0      |
| requests   | 8        | 0      |



→ ticketByEmail
  GET http://localhost:8091/microservice-bookingservice/api/flight/booking/history/rmnneha@gmail.com [404 Not Found, 219B, 27ms]

→ cancelTicket
  DELETE http://localhost:8091/microservice-bookingservice/api/flight/booking/cancel/11AC5FEB [404 Not Found, 158B, 24ms]



|                        | executed                                | failed |
|------------------------|-----------------------------------------|--------|
| iterations             | 1                                       | 0      |
| requests               | 8                                       | 0      |
| test-scripts           | 0                                       | 0      |
| prerequest-scripts     | 0                                       | 0      |
| assertions             | 0                                       | 0      |
| total run duration:    | 910ms                                   |        |
| total data received:   | 13.4kB (approx)                         |        |
| average response time: | 37ms [min: 23ms, max: 78ms, s.d.: 19ms] |        |



C:\Users\Neha Sharma>
```

5) Eureka Integration

The screenshot shows the Spring Eureka dashboard. At the top, it displays "HOME LAST 1000 SINCE STARTUP". Below this, the "System Status" section shows environment details like "Environment: test", "Data center: default", and various uptime metrics. A red warning message at the bottom states: "THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.". The "DS Replicas" section lists a single replica for "localhost". The "Instances currently registered with Eureka" table lists three services: MICROSERVICE-FLIGHTSERVICE, MICROSERVICES_APIGATEWAY, and MICROSERVICE_BOOKINGSERVICE, each with one instance and status UP.

6) Circuit Breaker

The screenshot shows a POST request to `http://localhost:8082/api/flight/booking/694317de950e240f7686bc79`. The request body is a JSON object representing a booking for two passengers. The response status is 503 Service Unavailable, indicating a temporary failure. The response body contains the message "Flight server down".

```
1 {
2   "customerName": "Neha Sharma",
3   "customerEmail": "neha@gmail.com",
4   "totalSeats": 2,
5   "passenger": [
6     {
7       "name": "Neha Sharma",
8       "gender": "Female",
9       "age": 22
10    },
11    {
12      "name": "Aarav Sharma",
13      "gender": "Male",
14      "age": 25
15    }
16  ],
17  "mealpref": "VEG",
```

7) Message Broker (KAFKA+ MailHog)

The screenshot shows the MailHog inbox interface. On the left, there's a sidebar with a green power icon labeled 'Connected', a message count 'Inbox (10)', and a 'Delete all messages' button. Below that is a section for 'Jim' with a bio: 'Jim is a chaos monkey.' and a 'Find out more at GitHub.' link, followed by an 'Enable Jim' button. The main area displays 10 messages in a table format:

From	Subject	Time Ago	Size
{{ getSender(message) }} neha@example.com	Flight Booking Confirmed	4 hours ago	295 B
{{ getSender(message) }} neha@example.com	Flight Booking Confirmed	4 hours ago	295 B
{{ getSender(message) }} neha@example.com	Flight Booking Confirmed	4 hours ago	295 B
{{ getSender(message) }} rmneha@gmail.com	Flight Booking Cancelled	4 hours ago	302 B
{{ getSender(message) }} rmneha@gmail.com	Flight Booking Confirmed	4 hours ago	296 B
{{ getSender(message) }} neha@gmail.com	Flight Booking Confirmed	4 hours ago	296 B
{{ getSender(message) }} neha@gmail.com	Flight Booking Confirmed	4 hours ago	291 B
{{ getSender(message) }} neha@gmail.com	Flight Booking Confirmed	4 hours ago	293 B
{{ getSender(message) }} neha@gmail.com	Flight Booking Confirmed	4 hours ago	294 B
{{ getSender(message) }} arnav@example.com	Flight Booking Cancelled	6 hours ago	303 B

This screenshot shows the MailHog inbox with a single message selected. The message details are visible in the header: From "Neha Sharma" @DESKTOP-QU5R752, Subject: Flight Booking Confirmed, To: neha@example.com. The message content is displayed below in plain text:

From "Neha Sharma" @DESKTOP-QU5R752
Subject: Flight Booking Confirmed
To: neha@example.com

Plain text Source

Your booking is confirmed.
PNR: 8D06BECF

This screenshot shows the MailHog inbox with another message selected. The message details are: From "Neha Sharma" @DESKTOP-QU5R752, Subject: Flight Booking Cancelled, To: rmneha@gmail.com. The message content is:

From "Neha Sharma" @DESKTOP-QU5R752
Subject: Flight Booking Cancelled
To: rmneha@gmail.com

Plain text Source

Your booking has been cancelled.
PNR: 11AC5FEB

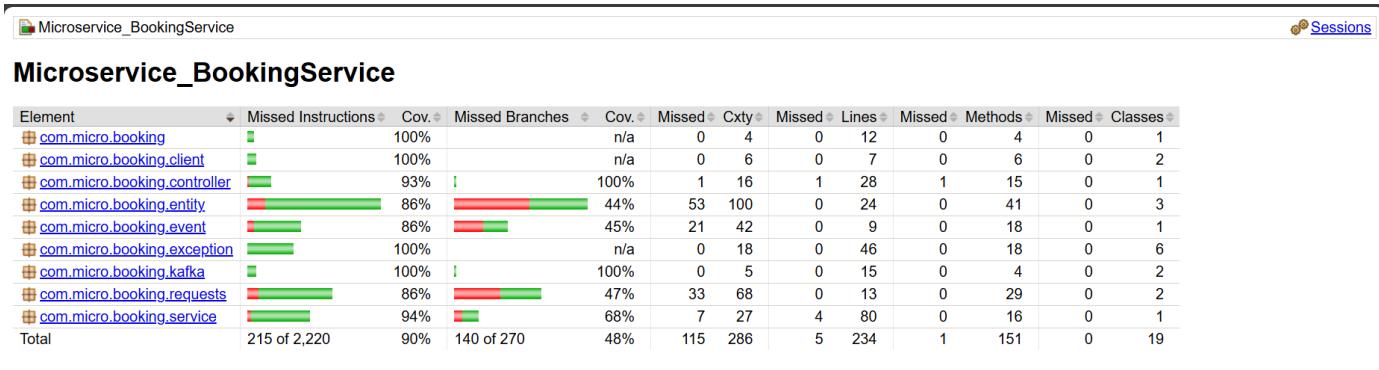
```

2025-12-20T21:25:23.022+05:30  INFO 40048 --- [MICROSERVICE-BOOKINGSERVICE] [main] o.a.k.clients.consumer.ConsumerConfig
: ConsumerConfig values:
  allow.auto.create.topics = true
  auto.commit.interval.ms = 5000
  auto.include.jmx.reporter = true
  auto.offset.reset = earliest
  bootstrap.servers = [localhost:9092]
  check.crcs = true
  client.dns.lookup = use_all_dns_ips
  client.id = consumer-booking-email-group-1
  client.rack =
  connections.max.idle.ms = 540000
  default.api.timeout.ms = 60000
  enable.auto.commit = false
  exclude.internal.topics = true
  fetch.max.bytes = 52428800
  fetch.max.wait.ms = 500
  fetch.min.bytes = 1
  group.id = booking-email-group
  group.instance.id = null
  heartbeat.interval.ms = 3000
  interceptor.classes = []
  internal.leave.group.on.close = true
  internal.throw.on.fetch.stable.offset.unsupported = false
  isolation.level = read_uncommitted
  key.deserializer = class org.springframework.kafka.support.serializer.ErrorHandlingDeserializer
  max.partition.fetch.bytes = 1048576
  max.poll.interval.ms = 300000
  max.poll.records = 500
  metadata.max.age.ms = 300000
  metric.reporters = []
  metrics.num.samples = 2
  metrics.recording.level = INFO
  metrics.sample.window.ms = 30000
  partition.assignment.strategy = [class org.apache.kafka.clients.consumer.RangeAssignor, class org.apache.kafka.clients.consumer.CooperativeStickyAssignor]

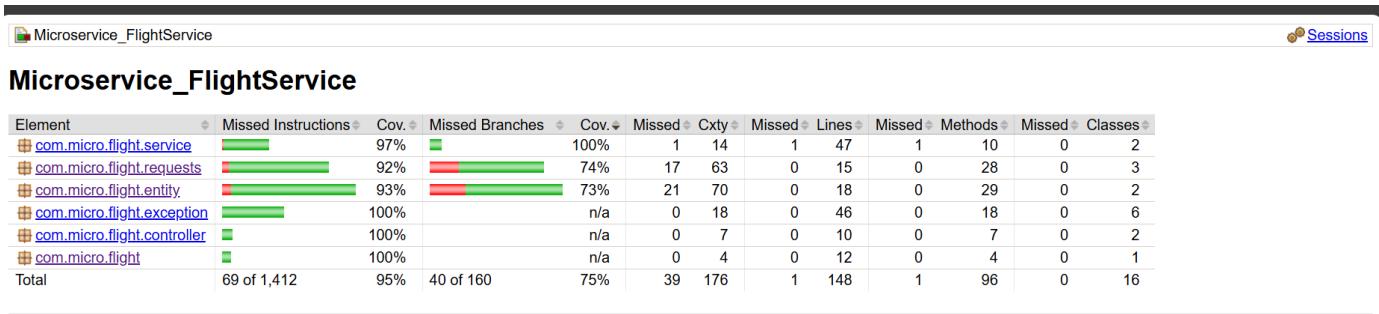
```

8) JACOCO REPORT

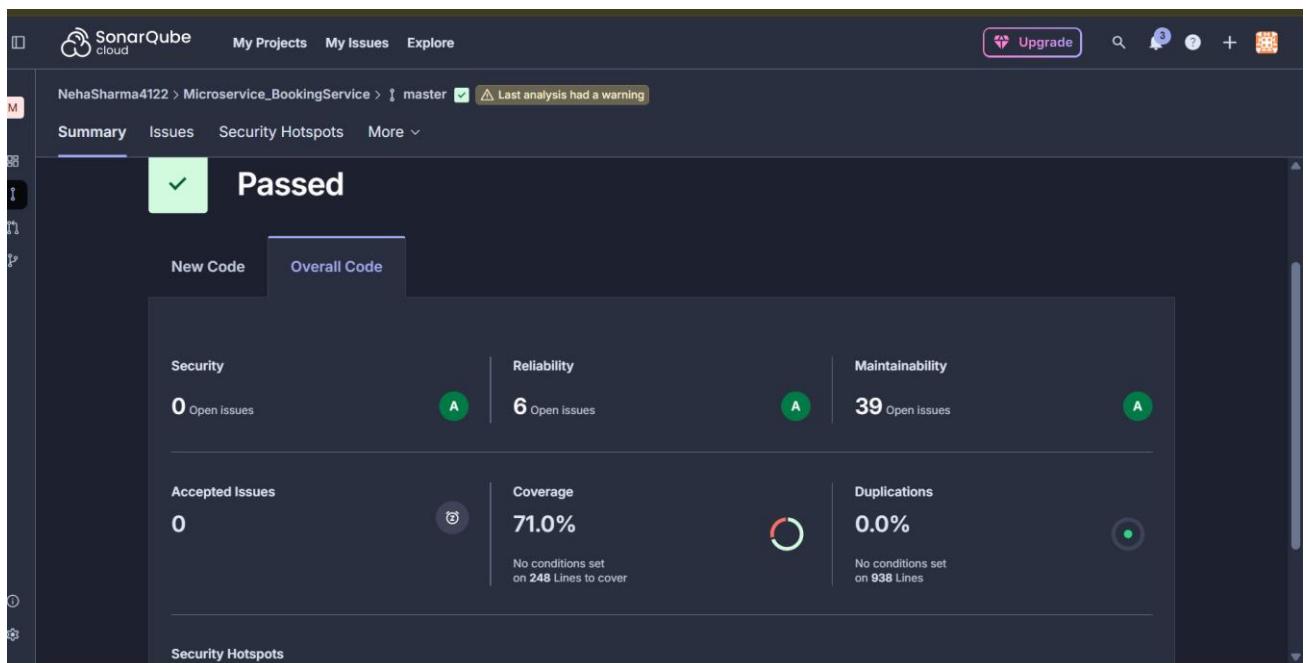
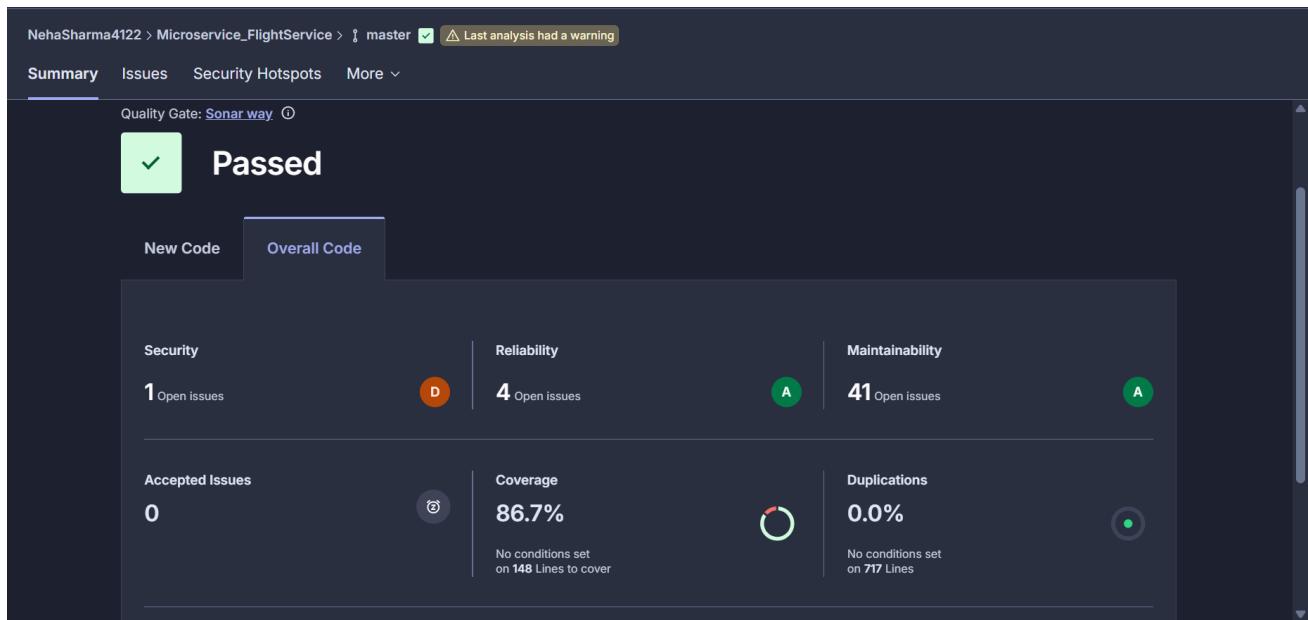
a) Booking Service



b) Flight Service



9) SONARCUBE



10) JMETER TESTING

```
C:\Users\Neha Sharma>C:\Users\Neha Sharma\Downloads\apache-jmeter-5.6.3\apache-jmeter-5.6.3\bin\jmeter.bat" -n -t "C:\Users\Neha Sharma\Desktop\GET.jmx" -l results.jtl -e -o report
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
Creating summariser <summary>
Created the tree successfully using C:\Users\Neha Sharma\Desktop\GET.jmx
Starting standalone test @ 2025 Dec 20 22:07:23 IST (1766248643549)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +    71 in 00:00:07 =   10.1/s Avg:   10 Min:     5 Max:   58 Err:      0 (0.00%) Active: 1 Started: 8 Finished: 7
summary +    29 in 00:00:02 =  13.5/s Avg:   10 Min:     6 Max:   61 Err:      0 (0.00%) Active: 0 Started: 10 Finished: 10
summary =   100 in 00:00:09 =   10.9/s Avg:   10 Min:     5 Max:   61 Err:      0 (0.00%)
Tidying up ... @ 2025 Dec 20 22:07:32 IST (1766248652931)
... end of run
```

11) CONFIG SERVER

https://github.com/NehaSharma4122/Microservices_configServer