

ASSIGNMENT – 8

Report by Bhavana Sree Naidu Yeluri

Contents

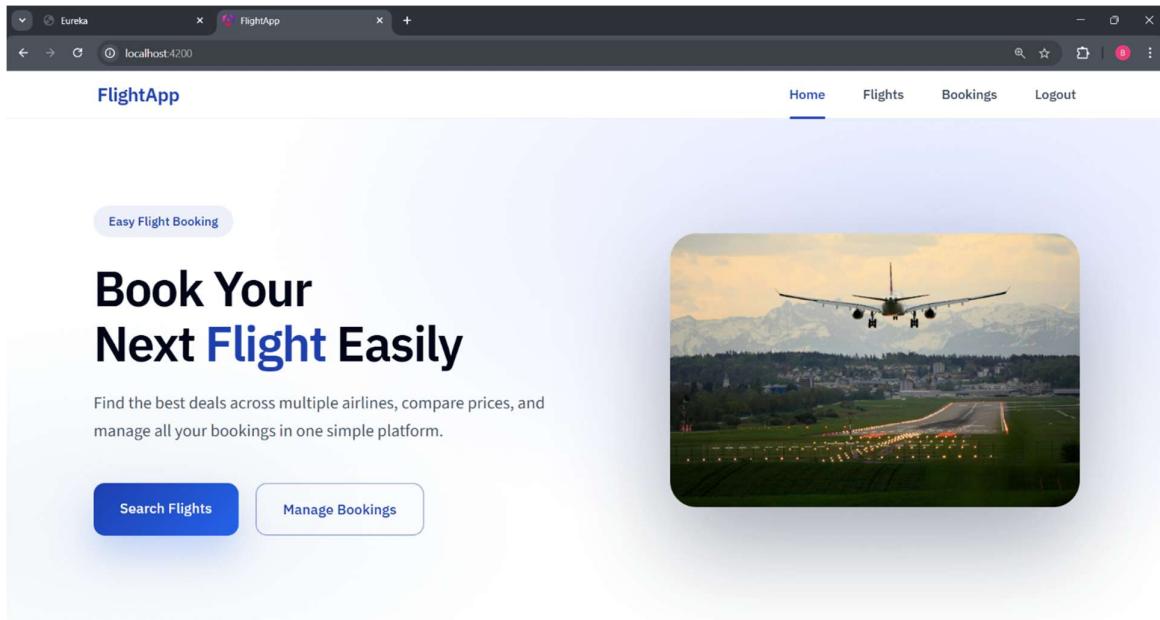
1. Project Overview
2. Frontend Screenshots
3. System Architecture
4. Eureka Dashboard
5. Docker
6. MongoDB Screenshots
7. Postman Screenshots
8. Email Notification
9. Message Broker – RabbitMQ
10. SonarQube
11. Jacoco
12. Jmeter
13. Newman
14. Swagger
15. Config Server

Project Overview

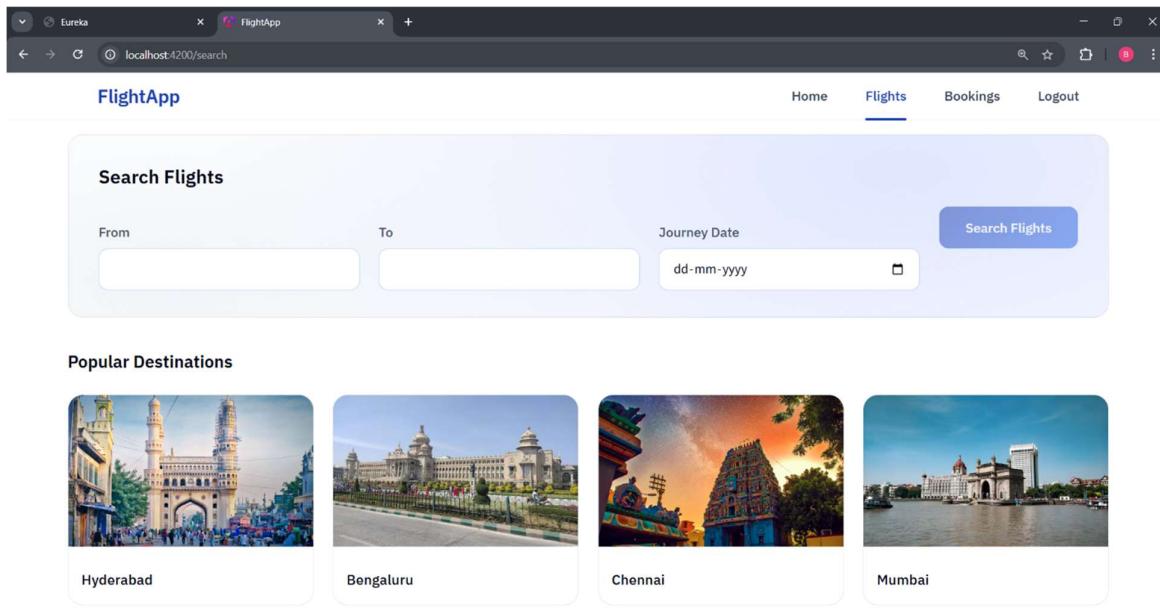
- Developed a responsive **Angular-based frontend** for a flight booking platform.
- Implemented **user authentication flows** including login and registration with **JWT**-based session handling.
- Built **flight search, booking history, and admin inventory**.
- Integrated **role-based navigation** to redirect users and admins to appropriate pages.
- Used **HTTP interceptors** for automatic Bearer token attachment to secured API requests.

Frontend Screenshots

- Landing page



- Search Flights



- Search Flights Results

The screenshot shows the FlightApp interface with a search query for a flight from Hyderabad to Bengaluru on 20-12-2025. The results show one available flight from Indigo.

Flight Details	Date	Price
Indigo Hyderabad → Bengaluru 12/20/25, 10:30AM - 12/20/25, 12:00PM Seats: 56	12/20/25, 10:30AM - 12/20/25, 12:00PM	₹3500

The screenshot shows the FlightApp interface with a search query for a flight from Hyderabad to Bengaluru on 20-12-2025. The results show two available flights: one from Indigo and one from Air India.

Flight Details	Date	Price
Indigo Hyderabad → Bengaluru 12/20/25, 10:30AM - 12/20/25, 12:00PM Seats: 56	12/20/25, 10:30AM - 12/20/25, 12:00PM	₹3500
Air India Hyderabad → Bengaluru 12/20/25, 1:15PM - 12/20/25, 2:35PM Seats: 80	12/20/25, 1:15PM - 12/20/25, 2:35PM	₹4500

- Admin Dashboard

The screenshot shows a web browser window titled "FlightApp" at the URL "localhost:4200/admin". The page has a header with "FlightApp" and navigation links for "Home", "Flights", "Bookings", and "Logout". Below the header is a section titled "Flight Inventory" with the sub-instruction "Manage airline inventory and seat availability". A blue button labeled "Add Inventory" is visible. The main content is a table listing flight details:

Flight No	Airline	From	To	Date	Departure	Arrival	Seats	Price
6E-131	Indigo	Hyderabad	Bengaluru	20-12-2025	10:30 AM	12:00 PM	60	₹3500
AI-258	Air India	Hyderabad	Bengaluru	20-12-2025	13:15 AM	14:45 PM	90	₹4500
AI-203	Air India	Bengaluru	Hyderabad	20-12-2025	10:30 AM	11:45 AM	120	₹4599
6E-512	IndiGo	Chennai	Delhi	20-12-2025	02:15 PM	05:10 PM	90	₹5299
SG-889	SpiceJet	Mumbai	Goa	20-12-2025	08:00 AM	09:10 AM	60	₹3499

- User Dashboard

The screenshot shows a web browser window titled "FlightApp" at the URL "localhost:4200/booking". The page has a header with "FlightApp" and navigation links for "Home", "Flights", "Bookings" (which is underlined, indicating it's the active tab), and "Logout". Below the header is a greeting "Hello, bhavana" and the sub-instruction "Your booking history and travel details".

Two flight bookings are listed side-by-side:

Flight ID: 693e75274c9c6f997d8c5980

ACTIVE

Seats	Meal	Selected Seats
2	non-veg	B1, B2

Booked on Dec 14, 2025, 1:19:27PM

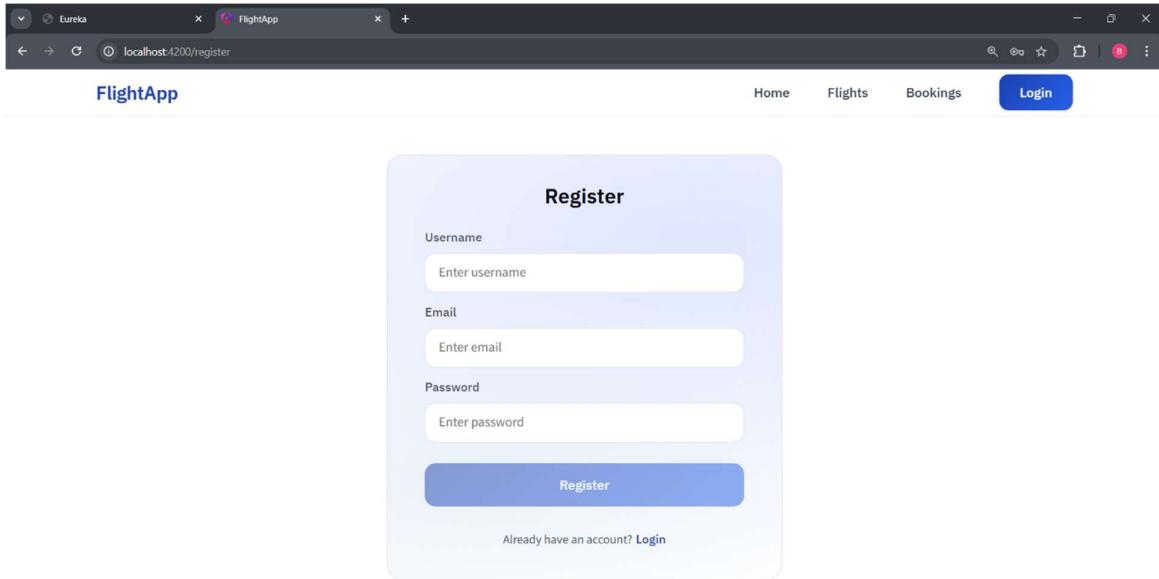
Flight ID: 693e75274c9c6f997d8c5980

ACTIVE

Seats	Meal	Selected Seats
2	veg	B3, B4

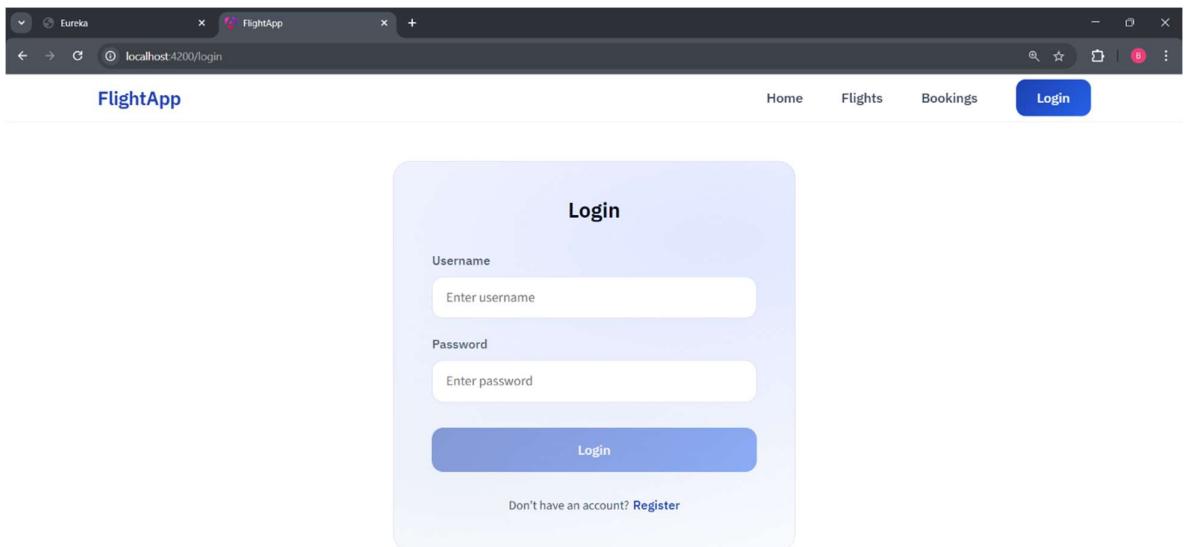
Booked on Dec 14, 2025, 2:03:29PM

- Register



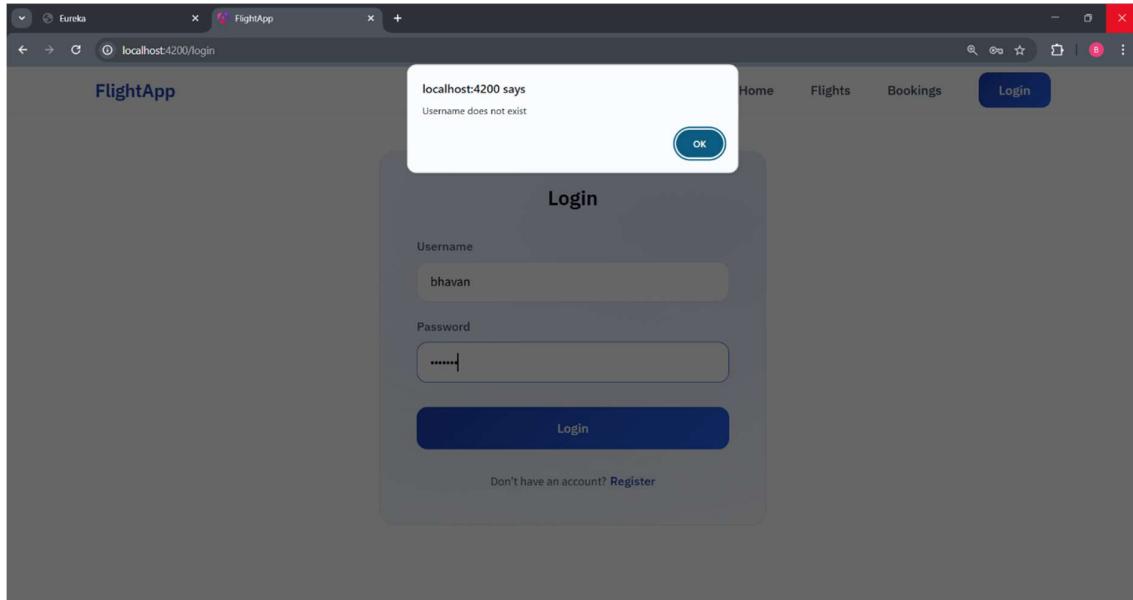
The screenshot shows a web browser window for the FlightApp. The address bar displays "localhost:4200/register". The main content area has a light blue background and features a "Register" form. The form includes three input fields: "Username" (placeholder: "Enter username"), "Email" (placeholder: "Enter email"), and "Password" (placeholder: "Enter password"). Below the inputs is a blue "Register" button. At the bottom of the form, there is a link "Already have an account? [Login](#)". The browser's navigation bar includes tabs for "Eureka" and "FlightApp", and buttons for back, forward, search, and refresh.

- Login

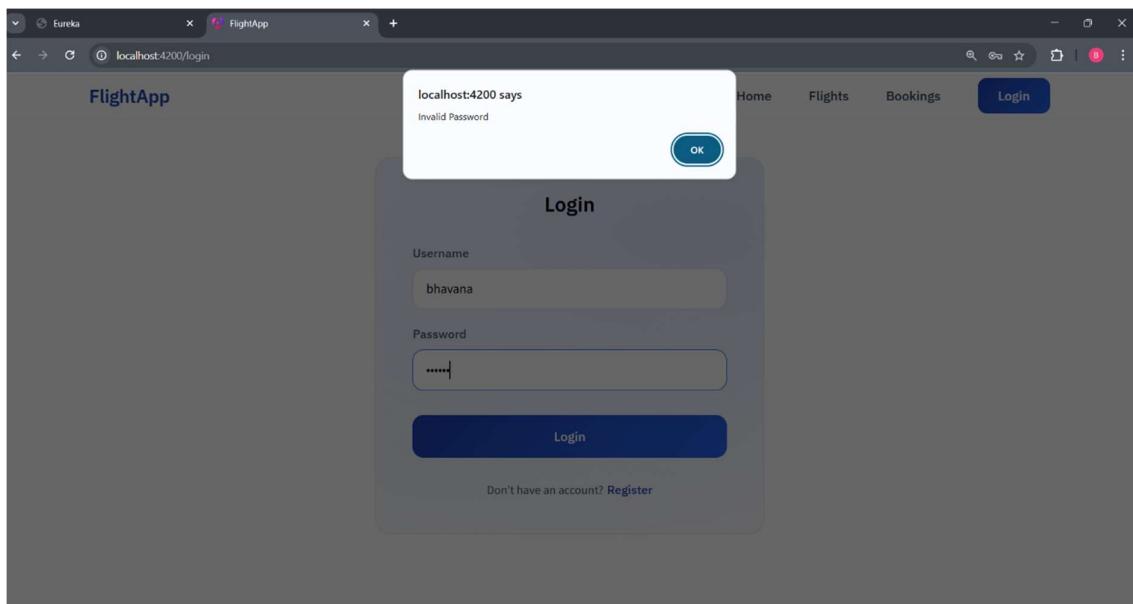


The screenshot shows a web browser window for the FlightApp. The address bar displays "localhost:4200/login". The main content area has a light blue background and features a "Login" form. The form includes two input fields: "Username" (placeholder: "Enter username") and "Password" (placeholder: "Enter password"). Below the inputs is a blue "Login" button. At the bottom of the form, there is a link "Don't have an account? [Register](#)". The browser's navigation bar includes tabs for "Eureka" and "FlightApp", and buttons for back, forward, search, and refresh.

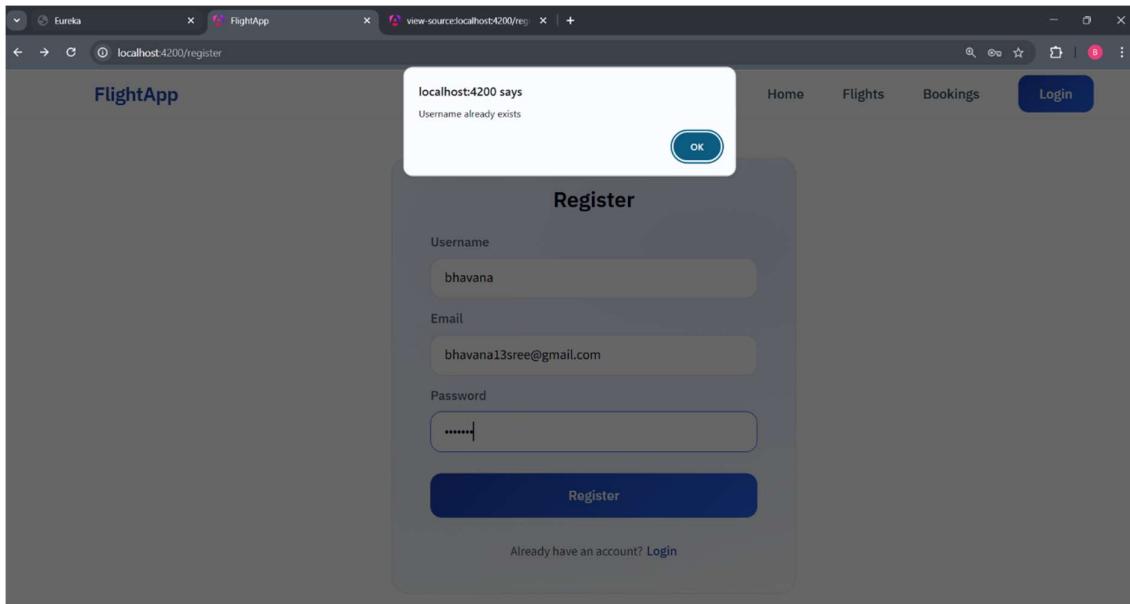
- Validations
- Invalid username



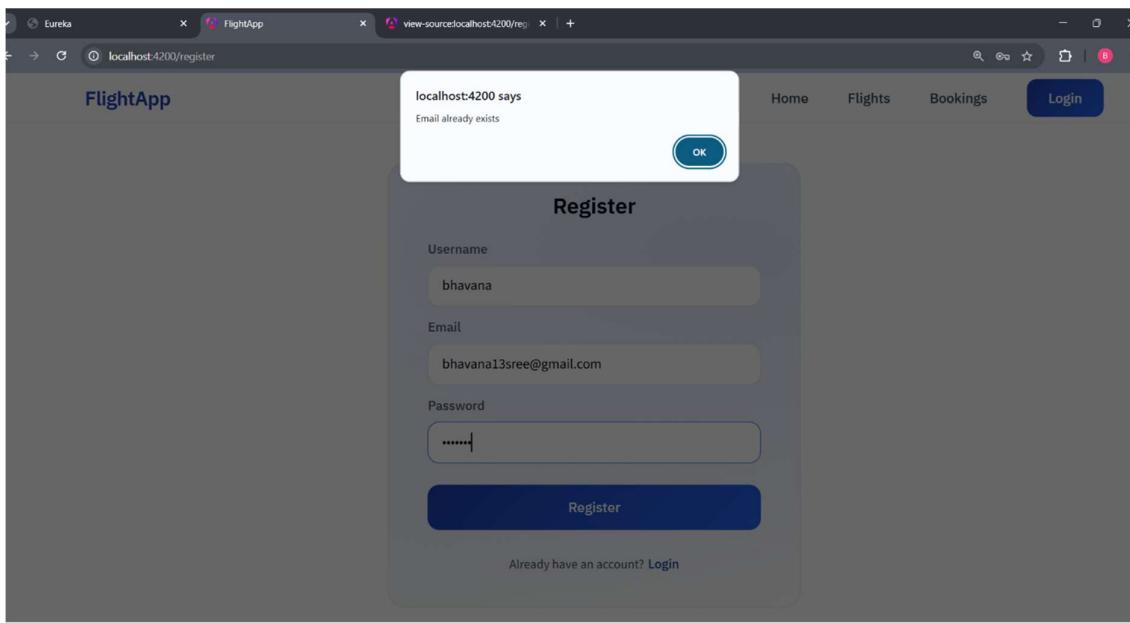
- Invalid password



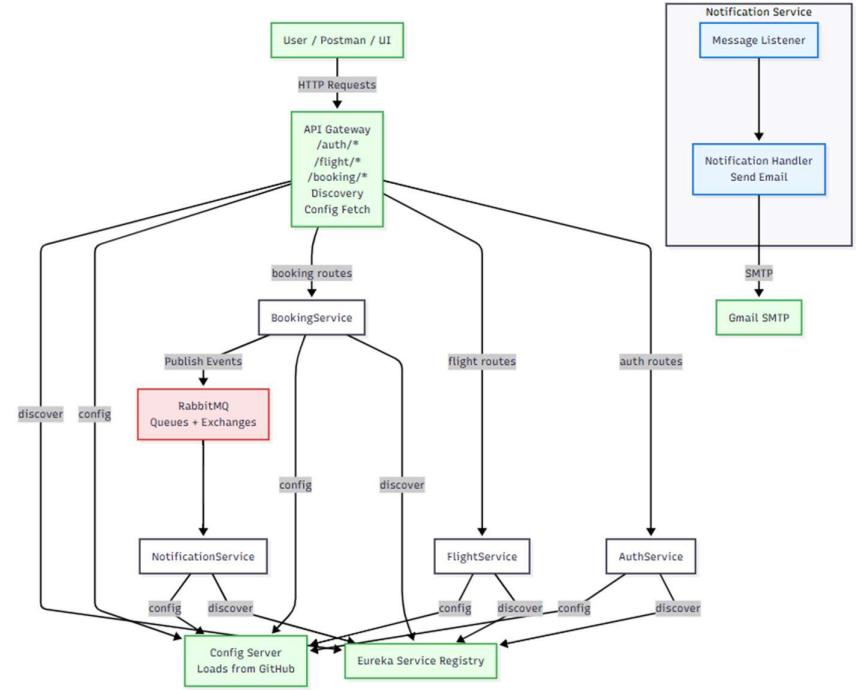
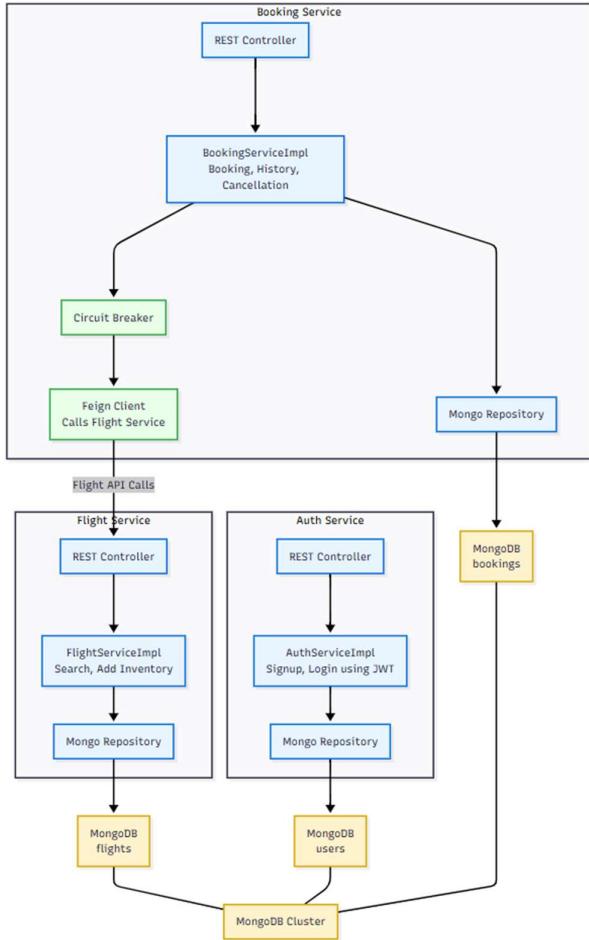
- Username already taken



- Email already in use



System Architecture



Eureka Dashboard

The screenshot shows the Eureka Dashboard interface. At the top, there's a header with tabs for 'DS Replicas' and 'localhost'. Below the header, a section titled 'Instances currently registered with Eureka' displays a table of registered services. The table has columns for Application, AMIs, Availability Zones, and Status. Each row contains a link to the service's configuration page.

Application	AMIs	Availability Zones	Status
API-GATEWAY-MICROSERVICES	n/a (1)	(1)	UP (1) - 7ecfbf7cd6be:api-gateway-microservices:9090
AUTH-SERVICE	n/a (1)	(1)	UP (1) - 87a008492562:auth-service:8090
BOOKING-MICROSERVICE	n/a (1)	(1)	UP (1) - b888516127e3:booking-microservice:8082
CONFIG-SERVER-MICROSERVICE	n/a (1)	(1)	UP (1) - 540f6eb2a79e:config-server-microservice:8888
FLIGHT-MICROSERVICE	n/a (1)	(1)	UP (1) - f880a1acf0ad:flight-microservice:8081
NOTIFICATION-MICROSERVICE	n/a (1)	(1)	UP (1) - c8b8df22b1a7:notification-microservice:8083

Docker

The screenshot shows the Docker dashboard. At the top, there are sections for 'Containers' and 'Give feedback'. Below this, there are performance metrics for CPU usage (52.77% / 1200%) and memory usage (2.51GB / 7.42GB). A 'Show charts' button is also present.

The main area displays a table of running containers. Each container entry includes a checkbox, the container name, container ID, image, port(s), CPU (%), memory usage, memory (%), disk read/write, and actions (stop, start, logs).

	Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/w	Actions
<input type="checkbox"/>	flight-app-micro	-	-	-	52.77%	2.51GB / 68.35Gi	32.98%	111.17MB /	Stop Start Logs
<input type="checkbox"/>	mongo	9c39db8ffd82	mongo:6	27017:27017	45.32%	173.4MB / 7.59Gi	2.23%	26.4MB / 2	Stop Start Logs
<input type="checkbox"/>	rabbitmq	606633e66506	rabbitmq:3	15672:15672	1.62%	144.8MB / 7.59Gi	1.86%	16.2MB / 7	Stop Start Logs
<input type="checkbox"/>	eureka-server	0dbe71cf2838	flight-app-r	8761:8761	2.79%	336.9MB / 7.59Gi	4.33%	11MB / 2.1	Stop Start Logs
<input type="checkbox"/>	config-server	540f6eb2a79e	flight-app-r	8888:8888	0.16%	331.6MB / 7.59Gi	4.26%	13.6MB / 2	Stop Start Logs
<input type="checkbox"/>	notification-s	c8b8df22b1a7	flight-app-r	8083:8083	0.19%	309.5MB / 7.59Gi	3.98%	2.06MB / 1	Stop Start Logs
<input type="checkbox"/>	auth-service	87a008492562	flight-app-r	8090:8090	0.16%	309.7MB / 7.59Gi	3.98%	11.1MB / 1	Stop Start Logs
<input type="checkbox"/>	flight-service	f880a1acf0ad	flight-app-r	8081:8081	1.6%	326.2MB / 7.59Gi	4.19%	18.2MB / 1	Stop Start Logs
<input type="checkbox"/>	api-gateway	7ecfbf7cd6be	flight-app-r	9090:9090	0.15%	326.2MB / 7.59Gi	4.19%	9.37MB / 1	Stop Start Logs
<input type="checkbox"/>	booking-serv	b888516127e3	flight-app-r	8082:8082	0.78%	307.9MB / 7.59Gi	3.96%	3.24MB / 1	Stop Start Logs

Below the table, there's a detailed view of a specific container named 'flight-app-microservices-mongodb'. It shows a tree view of the container's structure, including sub-containers like 'mongo:6' and 'rabbitmq'. On the right, there's a large JSON document showing the application configuration for this container, including details like runtime, environment variables, and connection details.

MongoDB Screenshots

- Roles in authdb

This screenshot shows the MongoDB Compass interface for the 'roles' collection in the 'authdb' database. The interface includes a search bar, filter options, and buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. Two documents are listed:

```
_id: ObjectId('6935bef3e9117f62656adce3')
name : "ROLE_ADMIN"
__class : "com.flighthapp.authservice.model.Role"

_id: ObjectId('6935bef3e9117f62656adce4')
name : "ROLE_USER"
__class : "com.flighthapp.authservice.model.Role"
```

- Users in authdb

This screenshot shows the MongoDB Compass interface for the 'users' collection in the 'authdb' database. The interface includes a search bar, filter options, and buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. Two documents are listed:

```
_id: ObjectId('6936bbdae4ec3c6220a14503')
username : "admin1"
email : "admin@example.com"
password : "$2a$10$/NuscM1ZP36j0TxWIt/yG0Zj8Z6zbzdIK84Izxp/T08GPXerqe2"
roles : Array (1)
__class : "com.flighthapp.authservice.model.User"

_id: ObjectId('6936bbe0e4ec3c6220a14504')
username : "user1"
email : "user1@example.com"
password : "$2a$10$WV846U/gT4/f7wzG5Nipc.FnfcbAJn.DlRlyQVtDOC/jrC/TQEK"
roles : Array (1)
__class : "com.flighthapp.authservice.model.User"
```

- Bookings in bookingdb

This screenshot shows the MongoDB Compass interface for the 'bookings' collection in the 'bookingdb' database. The interface includes a search bar, filter options, and buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. Two documents are listed:

```
_id: "696547F6"
flightId : "6936aaa615965f40316a9487"
user Name : "Bhavana"
user Email : "bhavana13sree@gmail.com"
number Of Seats : 2
passengers : Array (2)
selected Seats : Array (2)
meal Preference : "non-veg"
booking Time : 2025-12-08T10:38:40.248+00:00
journey Date : 2025-12-22T10:38:00.000+00:00
booking Status : "ACTIVE"
__class : "com.flighthapp.bookingsservice.model.Booking"

_id: "8889D573"
flightId : "6936aaa615965f40316a9487"
user Name : "Bhavana"
user Email : "bhavana13sree@gmail.com"
number Of Seats : 2
passengers : Array (2)
selected Seats : Array (2)
meal Preference : "non-veg"
booking Time : 2025-12-08T10:59:27.762+00:00
journey Date : 2025-12-22T10:38:00.000+00:00
booking Status : "CANCELLED"
__class : "com.flighthapp.bookingsservice.model.Booking"
```

- Flights in flightdb

The screenshot shows the MongoDB Compass interface with the following details:

- URL: `localhost:27017 > flightdb > flights`
- Collection: `flights`
- Document ID: `_id: ObjectId('6936aaa615965f40316a9487')`
- Fields:
 - `airlineName : "Indigo"`
 - `airlineLogoUrl : "https://example.com/logo.png"`
 - `fromPlace : "Bengaluru"`
 - `toPlace : "Hyderabad"`
 - `departureDateTime : 2025-12-22T10:30:00.000+00:00`
 - `arrivalDateTime : 2025-12-22T12:00:00.000+00:00`
 - `seats : Array (60)`
 - `_class : "com.flightapp.flightservice.model.Flight"`

Postman Screenshots

- Admin signup

The screenshot shows a POST request to `http://localhost:9090/auth/signup` with the following body:

```

1 {
2   "username": "admin1",
3   "email": "admin@example.com",
4   "password": "Admin@123",
5   "role": ["admin"]
6 }

```

The response status is `200 OK` with a message: `1 User registered successfully!`

- Admin sign in

The screenshot shows a POST request to `http://localhost:9090/auth/signin` with the following body:

```

1 {
2   "username": "admin1",
3   "password": "Admin@123"
4 }

```

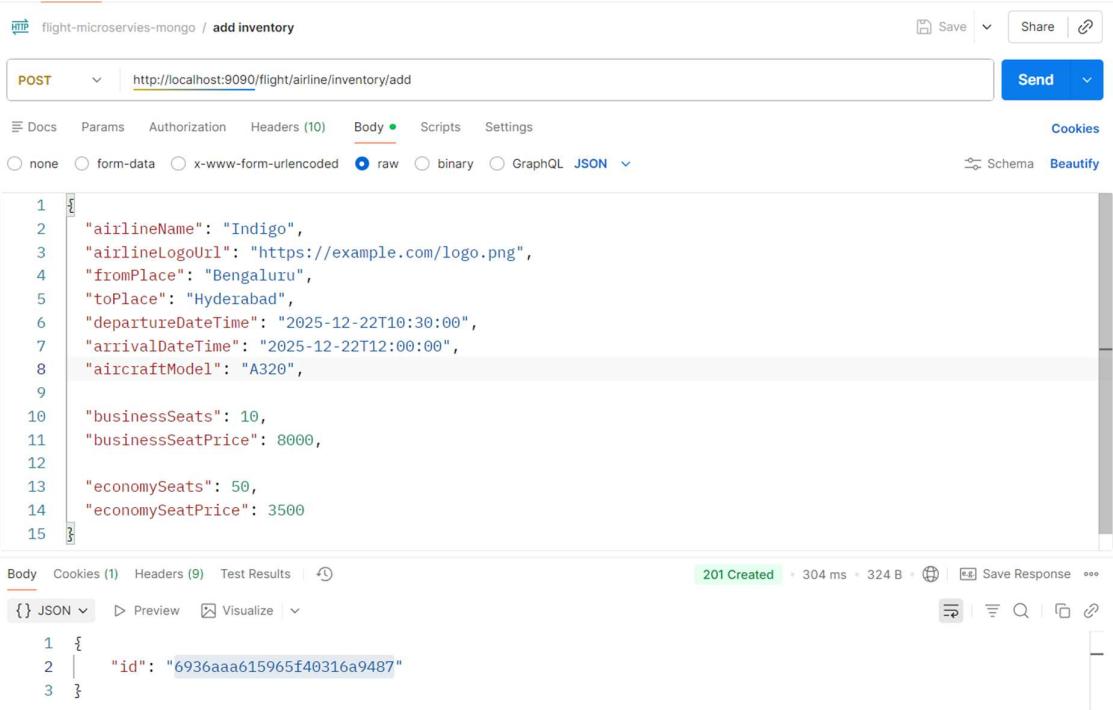
The response status is `200 OK` and contains a JSON token response:

```

1 {
2   "token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJhZGipbEiLCyb2xlcI6WyJST0xFETUlOIl0sImhdCI6MTc2NTE4DczMCwiZXhwIjoxNzY1Mjc1MTMwfQ.o-iFa-Yuy5FEEdRFyEa5kIjNjYIC-9dW-kNYFTinxhS0",
3   "type": "Bearer",
4   "id": "6936a46a07d55a3faae97760",
5   "username": "admin1",
6   "email": "admin@example.com",
7   "roles": [
8     "ROLE_ADMIN"
9   ]
10 }

```

- Add Inventory – admin route



The screenshot shows a Postman interface with the following details:

- Request URL:** `http://localhost:9090/flight/airline/inventory/add`
- Method:** POST
- Body:** Raw JSON (selected)
- JSON Content:**

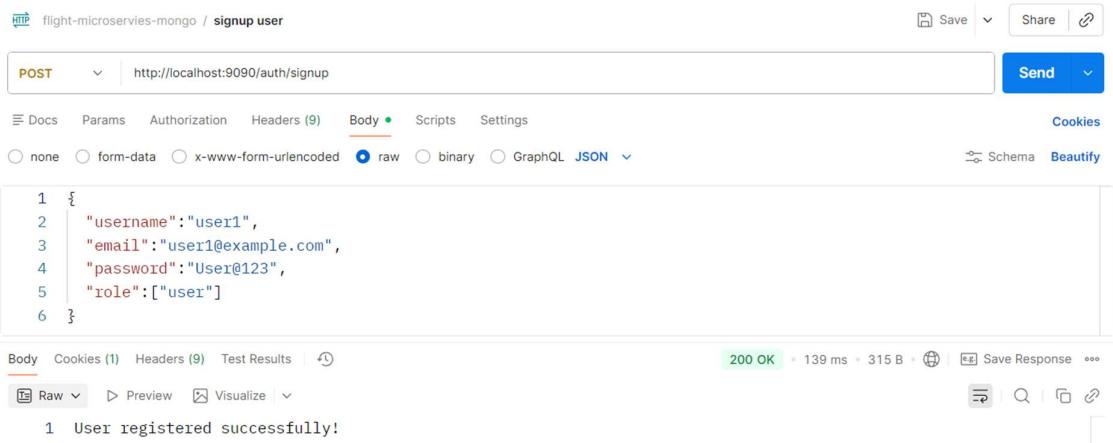
```

1  {
2    "airlineName": "Indigo",
3    "airlineLogoUrl": "https://example.com/logo.png",
4    "fromPlace": "Bengaluru",
5    "toPlace": "Hyderabad",
6    "departureDateTime": "2025-12-22T10:30:00",
7    "arrivalDateTime": "2025-12-22T12:00:00",
8    "aircraftModel": "A320",
9
10   "businessSeats": 10,
11   "businessSeatPrice": 8000,
12
13   "economySeats": 50,
14   "economySeatPrice": 3500
15 }
```
- Response Status:** 201 Created
- Response Body:**

```

1 {
2   "id": "6936aaa615965f40316a9487"
3 }
```

- User signup



The screenshot shows a Postman interface with the following details:

- Request URL:** `http://localhost:9090/auth/signup`
- Method:** POST
- Body:** Raw JSON (selected)
- JSON Content:**

```

1 {
2   "username": "user1",
3   "email": "user1@example.com",
4   "password": "User@123",
5   "role": ["user"]
6 }
```
- Response Status:** 200 OK
- Response Body:**

```
1 User registered successfully!
```

- User sign in

flight-microservices-mongo / **signin user**

POST http://localhost:9090/auth/signin

Body raw

```

1 {
2   "username": "user1",
3   "password": "User@123"
4 }

```

200 OK 139 ms 575 B

```

1 {
2   "token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJcVMSisInJvbGVzIjpblJPTEvfVVNFUiJdLCJpYXQiOjE3NjUxODY4NDgsImV4cCI6MTc2NTI3MzI0OH0.c8QwnVEAA1GD615EZ7Lcfbc48IZRCix-gSA_DggjDT0",
3   "type": "Beater",
4   "id": "69369d1cad3fd27667205c5",
5   "username": "user1",
6   "email": "user1@example.com",
7   "roles": [
8     "ROLE_USER"
9   ]
10 }

```

- Cancel ticket – user route

flight-microservices-mongo / **cancel ticket**

DELETE http://localhost:9090/booking/cancel/88B9D573

Body raw

```

1
2   "message": "Ticket cancelled successfully"
3

```

200 OK 112 ms 418 B

- Book flight – user route

flight-microservices-mongo / **book ticket**

POST http://localhost:9090/booking/6936aaa615965f40316a9487

Body raw

```

1 {
2   "userEmail": "bhavana13sree@gmail.com",
3   "userFirstName": "Bhavana",
4   "userLastName": "Siddharth",
5   "numberOfSeats": 2,
6   "passengers": [
7     {
8       "name": "Bhavana", "gender": "Female", "age": 22
9     },
10    {
11      "name": "Siddharth", "gender": "Male", "age": 23
12    }
13  ],
14  "selectedSeats": ["B1", "B2"],
15  "mealPreference": "non-veg"
16 }

```

200 OK 585 ms 393 B

```

1
2   "pnr": "88B9D573"
3

```

- Get history using email – user route

```

200 OK
88 ms
1.18 KB
200 OK
88 ms
1.18 KB
{
  "id": "88B9D573",
  "flightId": "6936aaa615965f40316a9487",
  "userName": "Bhavana",
  "userEmail": "bhavana13sree@gmail.com",
  "numberOfSeats": 2,
  "passengers": [
    {
      "name": "Bhavana",
      "gender": "Female",
      "age": 22
    },
    {
      "name": "Siddharth",
      "gender": "Male",
      "age": 23
    }
  ],
  "selectedSeats": [
    "B1",
    "B2"
  ]
}

```

- Get ticket by PNR – user route

```

200 OK
111 ms
770 B
200 OK
111 ms
770 B
{
  "gender": "Female",
  "age": 22
},
{
  "name": "Siddharth",
  "gender": "Male",
  "age": 23
},
"selectedSeats": [
  "B1",
  "B2"
],
"mealPreference": "non-veg",
"bookingTime": "2025-12-08T10:59:27.762",
"journeyDate": "2025-12-22T10:30:00",
"bookingStatus": "ACTIVE"
}

```

- Search Flight – user route

```

POST http://localhost:9090/flight/search
{
  "fromPlace": "Bengaluru",
  "toPlace": "Hyderabad",
  "journeyDate": "2025-12-22"
}
[{"flightId": "6936aaa615965f40316a9487", "airlineName": "Indigo", "airlineLogoUrl": "https://example.com/logo.png", "departureDateTime": "2025-12-22T10:30:00", "arrivalDateTime": "2025-12-22T12:00:00", "fromPlace": "Bengaluru", "toPlace": "Hyderabad", "availableSeats": 58, "lowestPrice": 3500.0}]
  
```

Email Notification

Flight Ticket Booking Confirmation - PNR 88B9D573

22071a66d9@vnrvjet.in to me 4:29PM (1 hour ago)

Hello Bhavana,

Your flight ticket has been booked successfully.
PNR Number: 88B9D573
Status: BOOKED
Seats: [B1, B2]

Thank you for choosing FlightApp

22071a66d9@vnrvjet.in to me 4:32PM (1 hour ago)

Hello Bhavana,

Your flight ticket has been booked successfully.
PNR Number: 88B9D573
Status: CANCELLED

Seats: [B1, B2]

Thank you for choosing FlightApp

Message broker – RabbitMQ

Booking Notification Received:

User: Bhavana

Email: bhavana@gmail.com

PNR: 0EE5A502

Flight: 6933126926e290a752340073

Status: BOOKED

Seats: [B7, B8]

Cancellation Notification Received:

PNR: 0EE5A502

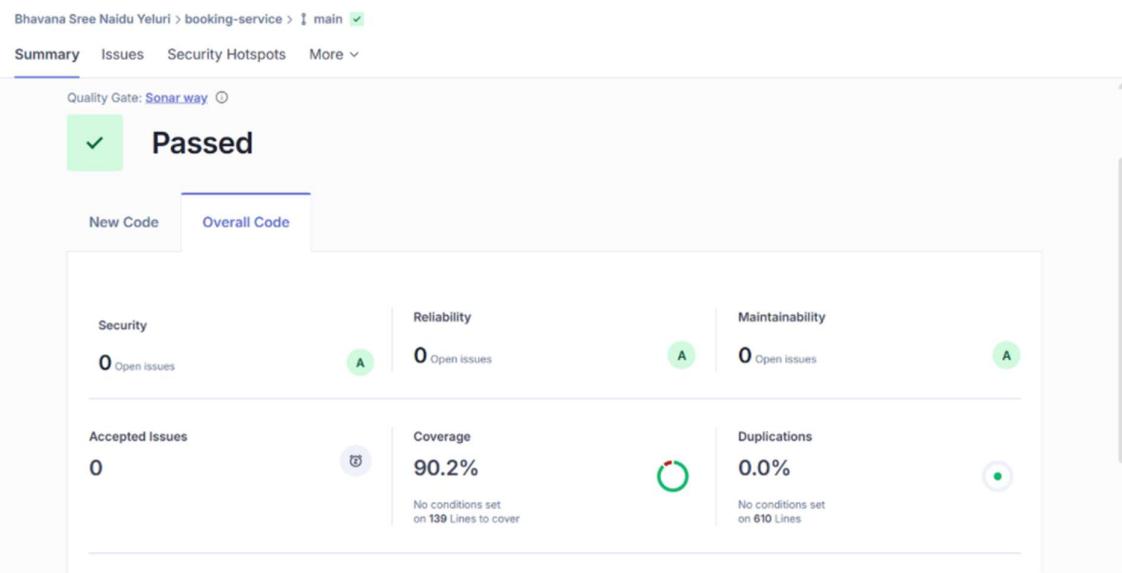
Email: bhavana@gmail.com

Flight: 6933126926e290a752340073

Status: CANCELLED

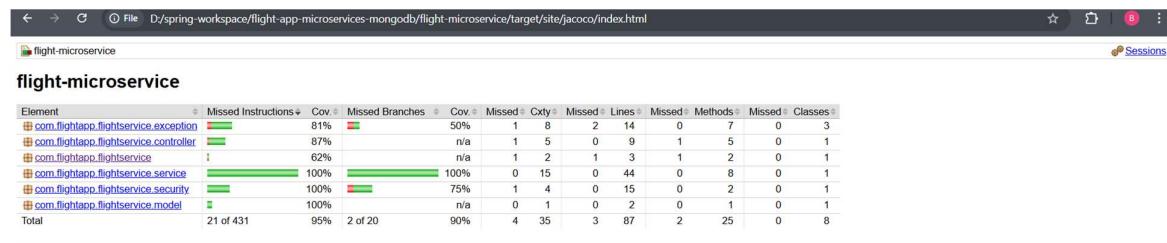
Seats: [B7, B8]

Sonarqube



Jacoco

- 95% coverage



Jmeter

Summary Report											
Name:	Summary Report										
Comments:											
Write results to file / Read from file											
Filename							Browse...	Log/Display Only:	<input type="checkbox"/> Errors	<input type="checkbox"/> Successes	<input type="checkbox"/> Configure
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes	
HTTP Request	20	6	4	8	1.08	0.00%	20.8/sec	9.46	2.92	466.0	
TOTAL	20	6	4	8	1.08	0.00%	20.8/sec	9.46	2.92	466.0	

Summary Report

Name:

Comments:

Write results to file / Read from file

Filename <input type="text"/>	<input type="button" value="Browse..."/>	<input type="checkbox"/> Log/Display Only:	<input type="checkbox"/> Errors	<input type="checkbox"/> Successes	<input type="button" value="Configure"/>					
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
HTTP Request	50	6	5	9	0.90	0.00%	50.4/sec	22.94	7.09	466.0
TOTAL	50	6	5	9	0.90	0.00%	50.4/sec	22.94	7.09	466.0

Summary Report										
Name:	Summary Report									
Comments:										
Write results to file / Read from file										
Filename	<input type="text"/>				<input type="button" value="Browse..."/>	<input type="checkbox"/> Log/Display Only:	<input type="checkbox"/> Errors	<input type="checkbox"/> Successes	<input type="button" value="Configure"/>	
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	100	15	4	106	21.33	0.00%	98.8/sec	44.97	13.90	466.0
TOTAL	100	15	4	106	21.33	0.00%	98.8/sec	44.97	13.90	466.0

The screenshot shows the JMeter interface with the 'View Results Tree' listener selected in the Test Plan tree. The main panel displays a tree view of 'Sampler result' entries, each representing an 'HTTP Request'. There are 15 such entries listed. At the bottom of the tree view, there are two tabs: 'Raw' and 'Parsed', with 'Raw' currently selected. The top right corner of the window shows the status '00:00:01' and metrics '0 0/100'.

Newman

```
newman
Booking-service(micro)

→ book ticket
  POST http://localhost:8082/api/flight/booking/2 [201 Created, 478B, 138ms]

→ search by pnr
  GET http://localhost:8082/api/flight/ticket/106A56B0 [200 OK, 496B, 112ms]

→ search by email
  GET http://localhost:8082/api/flight/booking/history/alice@example.com [200 OK, 1.04kB, 14ms]

→ New Request
  DELETE http://localhost:8082/api/flight/booking/cancel/106A56B0 [200 OK, 246B, 9ms]

      iterations | executed | failed
      :-----: | :-----: | :-----:
      requests | 1 | 0
      test-scripts | 4 | 0
      prerequest-scripts | 0 | 0
      assertions | 0 | 0

      total run duration: 311ms
      total data received: 1.5kB (approx)
      average response time: 68ms [min: 9ms, max: 138ms, s.d.: 57ms]
```

```
newman
API-Gateway(Flight-micro)

→ search flights
  POST http://localhost:8081/FLIGHT-SERVICE/api/flights/search [201 Created, 338B, 402ms]

→ get by pnr
  GET http://localhost:8081/BOOKING-SERVICE/api/flight/ticket/106A56B0 [200 OK, 448B, 57ms]

→ Book ticket
  POST http://localhost:8081/BOOKING-SERVICE/api/flight/booking/2 [201 Created, 429B, 49ms]

      iterations | executed | failed
      :-----: | :-----: | :-----:
      requests | 1 | 0
      test-scripts | 3 | 0
      prerequest-scripts | 0 | 0
      assertions | 0 | 0

      total run duration: 538ms
      total data received: 795B (approx)
      average response time: 169ms [min: 49ms, max: 402ms, s.d.: 164ms]
```

Newman Report

Collection

flight-microservices

Time

Tue Dec 02 2025 01:08:55 GMT+0530 (India Standard Time)

Exported with

Newman v6.2.1

	Total	Failed
Iterations	1	0
Requests	6	0
Prerequest Scripts	0	0
Test Scripts	0	0
Assertions	0	0
Total run duration	659ms	
Total data received	953B (approx)	
Average response time	34ms	
Total Failures	0	

Swagger

The screenshot shows the Swagger UI interface for a RESTful API. It displays two main sections: **booking-controller** and **flight-controller**.

booking-controller (Top Section):

- POST /api/flight/booking/{flightId}** (Green button)
- GET /api/flight/booking/ticket/{pnr}** (Blue button)
- GET /api/flight/booking/history/{email}** (Blue button)
- GET /api/flight/booking/download/{pnr}** (Blue button)
- DELETE /api/flight/booking/cancel/{pnr}** (Red button)

Schemas (Sub-section):

- BookingRequest >
- Booking >
- TicketResponse >

flight-controller (Bottom Section):

- PUT /api/flight/update-seats/{flightId}/{count}** (Orange button)
- PUT /api/flight/rollback-seats/{flightId}/{count}** (Orange button)
- POST /api/flight/search** (Green button)
- POST /api/flight/airline/inventory/add** (Green button)
- GET /api/flight/get/{id}** (Blue button)

Schemas (Sub-section):

- FlightSearchRequest >
- Flight >
- FlightInventoryRequest >

Config Server

```
localhost:8888/notification-microservice/default
Pretty-print   
  
{  
    "name": "notification-microservice",  
    "profiles": [  
        "default"  
    ],  
    "label": null,  
    "version": "e482c88225bd928d6d2a073e82802fba56e186c1",  
    "state": "",  
    "propertySources": [  
        {  
            "name": "https://github.com/bhavana1312/config-server-microservices/notification-microservice.properties",  
            "source": {  
                "server.port": "8083",  
                "spring.rabbitmq.host": "localhost",  
                "spring.rabbitmq.port": "5672",  
                "spring.rabbitmq.username": "guest",  
                "spring.rabbitmq.password": "guest",  
                "spring.mail.host": "smtp.gmail.com",  
                "spring.mail.port": "587",  
                "spring.mail.username": "${MAIL_USERNAME}",  
                "spring.mail.password": "${MAIL_PASSWORD}",  
                "spring.mail.properties.mail.smtp.auth": "true",  
                "spring.mail.properties.mail.smtp.starttls.enable": "true",  
                "spring.mail.properties.mail.smtp.connectiontimeout": "5000",  
                "spring.mail.properties.mail.smtp.timeout": "5000",  
                "spring.mail.properties.mail.smtp.writetimeout": "5000"  
            }  
        }  
    ]  
}  
  
localhost:8888/booking-microservice/default
Pretty-print   
  
[  
    {"name": "booking-microservice",  
     "profiles": [  
         "default"  
     ],  
     "label": null,  
     "version": "e482c88225bd928d6d2a073e82802fba56e186c1",  
     "state": "",  
     "propertySources": [  
         {  
             "name": "https://github.com/bhavana1312/config-server-microservices/booking-microservice.properties",  
             "source": {  
                 "server.port": "8082",  
                 "spring.data.mongodb.uri": "mongodb://localhost:27017",  
                 "spring.data.mongodb.database": "bookingdb",  
                 "logging.level.com.flighthapp.bookingservice.feign": "DEBUG",  
                 "spring.rabbitmq.host": "localhost",  
                 "spring.rabbitmq.port": "5672",  
                 "spring.rabbitmq.username": "guest",  
                 "spring.rabbitmq.password": "guest",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.register-health-indicator": "true",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.sliding-window-size": "5",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.minimum-number-of-calls": "3",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.failure-rate-threshold": "50",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.wait-duration-in-open-state": "10s",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.permitted-number-of-calls-in-half-open-state": "2",  
                 "resilience4j.circuitbreaker.instances.flightServiceBreaker.automatic-transition-from-open-to-half-open-enabled": "true"  
             }  
         }  
     ]  
]
```

```
localhost:8888/flight-microservice/default
Pretty-print   
  
{  
    "name": "flight-microservice",  
    "profiles": [  
        "default"  
    ],  
    "label": null,  
    "version": "e482c88225bd928d6d2a073e82802fba56e186c1",  
    "state": "",  
    "propertySources": [  
        {  
            "name": "https://github.com/bhavana1312/config-server-microservices/flight-microservice.properties",  
            "source": {  
                "server.port": "8081",  
                "spring.mongodb.uri": "mongodb://localhost:27017/flightdb",  
                "logging.level.org.springframework": "INFO",  
                "spring.mvc.throw-exception-if-no-handler-found": "true",  
                "spring.web.resources.add-mappings": "false"  
            }  
        }  
    ]  
}
```

```
localhost:8888/api-gateway/default
Pretty-print   
  
{  
    "name": "api-gateway",  
    "profiles": [  
        "default"  
    ],  
    "label": null,  
    "version": "e482c88225bd928d6d2a073e82802fba56e186c1",  
    "state": "",  
    "propertySources": []  
}
```

config-server-microservices Public

[Pin](#) [Watch 0](#)

[main](#) [1 Branch](#) [0 Tags](#) [Go to file](#) [Add file](#) [Code](#)

 bhavana1312	Update notification-microservice.properties	d51dd33 · 4 hours ago	28 Commits
api-gateway-microservices.properties	Update api-gateway-microservices.properties	8 hours ago	
auth-service.properties	Update auth-service.properties	9 hours ago	
booking-microservice.properties	Update booking-microservice.properties	5 hours ago	
flight-microservice.properties	Update flight-microservice.properties	9 hours ago	
notification-microservice.properties	Update notification-microservice.properties	4 hours ago	