

Log files docker

```
tp://eureka-server:8761/eureka/apps/FLIGHTS-SERVICE]: Connect to http://eureka-server:8761 [eureka-server/172.19.0.4] failed: Connection refused
flights-service [2025-12-08T09:53:16.846Z] WARN 1 --- [flights-service] [foReplicator->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry registration failed Cannot execute request on any known server
flights-service
flights-service com.netflix.discovery.shared.transport.TransportException: Cannot execute request on any known server
flights-service at com.netflix.discovery.shared.transport.decorator.RetryableEurekaHttpClient.execute(RetryableEurekaHttpClient.java:172) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.EurekaHttpClientDecorator.execute(EurekaHttpClientDecorator.java:125) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.SessionedEurekaHttpClientDecorator$SessionedEurekaHttpClientDecorator.execute(SessionedEurekaHttpClientDecorator.java:59) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.SessionedEurekaHttpClient.execute(SessionedEurekaHttpClient.java:76) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.EurekaHttpclientDecorator.register(EurekaHttpclientDecorator.java:66) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.EurekaDiscoveryClient.register(DiscoveryClient.java:828) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.InstanceInfoReplicator.run(InstanceInfoReplicator.java:125) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.InstanceInfoReplicator$2.run(InstanceInfoReplicator.java:195) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:539) ~[na:na]
flights-service at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264) ~[na:na]
flights-service at java.base/java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.run(ScheduledThreadPoolExecutor.java:304) ~[na:na]
flights-service at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1134) ~[na:na]
flights-service at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635) ~[na:na]
flights-service at java.base/java.lang.Thread.run(Thread.java:833) ~[na:na]
2025-12-08T09:53:14.066Z WARN 1 --- [flights-service] [foReplicator->N] c.n.discovery.InstanceInfoReplicator : There was a problem with the instance info replicator
flights-service
flights-service com.netflix.discovery.shared.transport.TransportException: Cannot execute request on any known server
flights-service at com.netflix.discovery.shared.transport.decorator.RetryableEurekaHttpClient.execute(RetryableEurekaHttpClient.java:112) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.EurekaHttpClientDecorator.register(EurekaHttpClientDecorator.java:66) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.EurekaHttpclientDecorator$1.execute(EurekaHttpclientDecorator.java:59) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.shared.transport.decorator.SessionedEurekaHttpClientDecorator.execute(SessionedEurekaHttpClient.java:76) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.DiscoveryClient.register(DiscoveryClient.java:828) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.InstanceInfoReplicator.run(InstanceInfoReplicator.java:125) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at com.netflix.discovery.InstanceInfoReplicator$2.run(InstanceInfoReplicator.java:195) ~[eureka-client-2.0.3.jar!/:2.0.3]
flights-service at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:539) ~[na:na]
flights-service at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264) ~[na:na]
flights-service at java.base/java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.run(ScheduledThreadPoolExecutor.java:304) ~[na:na]
flights-service at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1134) ~[na:na]
flights-service at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635) ~[na:na]
flights-service at java.base/java.lang.Thread.run(Thread.java:833) ~[na:na]
2025-12-08T09:53:23.800Z INFO 1 --- [flights-service] [beatExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - Registering app/s/FLIGHTS-SERVICE
flights-service [2025-12-08T09:53:23.801Z] INFO 1 --- [flights-service] [beatExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registered
flights-service
flights-service [2025-12-08T09:53:23.802Z] INFO 1 --- [flights-service] [beatExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry status: 204
flights-service
flights-service [2025-12-08T09:53:43.548Z] INFO 1 --- [flights-service] [reshExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry refresh
flights-service [2025-12-08T09:53:43.549Z] INFO 1 --- [flights-service] [reshExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry refresh
flights-service [2025-12-08T09:53:43.549Z] INFO 1 --- [flights-service] [reshExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry refresh
flights-service [2025-12-08T09:53:43.549Z] INFO 1 --- [flights-service] [reshExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry refresh
flights-service [2025-12-08T09:53:43.549Z] INFO 1 --- [flights-service] [reshExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry refresh
flights-service [2025-12-08T09:53:43.549Z] INFO 1 --- [flights-service] [reshExecutor->N] com.netflix.discovery.DiscoveryClient : DiscoveryClient_FLIGHTS-SERVICE/9d5aea3ccb03:flights-service:8081 - registry refresh
flights-service [2025-12-08T09:53:43.621Z] INFO 1 --- [flights-service] [ctor-http-nio-3] c.f.flights.service.AirlineService : Creating airline and clearing cache
flights-service [2025-12-08T09:56:07.162Z] INFO 1 --- [flights-service] [ctor-http-nio-3] c.f.flights.service.AirlineService : Inserting document containing fields: {airlineName, airlineCode, contactEmail}
flights-service [2025-12-08T09:56:07.162Z] DEBUG 1 --- [flights-service] [ctor-http-nio-3] o.s.d.m.core.ReactiveMongoTemplate : Airline created: 493a0b78097b9c285f9afc
flights-service [2025-12-08T09:56:12.195Z] INFO 1 --- [flights-service] [ctor-http-nio-3] c.f.flights.service.AirlineService : CACHE MISS - Fetching all airlines from MongoDB
flights-service [2025-12-08T09:56:12.282Z] DEBUG 1 --- [flights-service] [ctor-http-nio-3] o.s.d.m.core.ReactiveMongoTemplate : find using query: {} fields: Document{} for class: class com.flightservice.model.Airline
flights-service [2025-12-08T09:56:12.258Z] INFO 1 --- [flights-service] [nitoLoopGroup-3-3] c.f.flights.service.AirlineService : All airlines fetched and cached
flights-service [2025-12-08T09:58:12.248Z] INFO 1 --- [flights-service] [rap-executor->N] c.n.d.s.r.aws.ConfigClusterResolver : Resolving eureka endpoints via configuration
syainumyggade@Sais-MacBook-Air-2: Flight-booking-microservices %
```

Flight-service

Bookings-service

Api-gateway

Eureka-server

Email sending using SMTP



nsvsujay@gmail.com

to me ▾

Dear Sujay NSVSU,

Your flight booking has been confirmed!

Booking Details:

=====

PNR: PNR09D91A93

Flight: SpiceJet-5

From: Mumbai

To: Goa

Departure: 2025-12-23T07:00

Number of Seats: 2

Seat Numbers: A1, A2

Total Price: ₹5000.00

Thank you for booking with us!

Best Regards,

Flight Booking Team



Postman Testing

The screenshot shows the Postman application interface. The left sidebar displays 'My Workspace' with various collections like 'airplane-microservices', 'Automation', and 'MongoSpring'. The main area shows a 'POST Request to add airlines' under the 'airplane-microservices' collection. The request details show a POST method to 'http://localhost:8080/airline'. The 'Headers' tab lists 8 items. The 'Body' tab is selected, showing a JSON response with one item: { "airlineId": "692db239a431c82390f3b637" }. The status bar at the bottom indicates '200 OK'.

View Window Help

Home Workspaces API Network

Search Postman

POST Post GET Get R POST Post POST Post POST Post GET Get R GET G

My Workspace

Collections Environments Flows History

POST airplane-microservices / Post Request to add airlines

POST http://localhost:8080/airline

Params Authorization Headers (8) Body Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (2) Test Results 200 OK 20 ms 111 B Save Response

{ } JSON Preview Visualize

```
1 {  
2   "airlineId": "692db239a431c82390f3b637"  
3 }
```

Online Find and replace Console

Runner Start Proxy Cookies Vault Trash

View Window Help

19°C Mon 1 Dec 11:53PM

Home Workspaces API Network Search Postman Invite Upgrade

My Workspace New Import POST Post GET Get R POST Post POST Post POST Post GET Get R GET G + No environment ↻

Collections Environments Flows History

airplane-microservices / Get Request to see all airlines

GET http://localhost:8080/airline

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

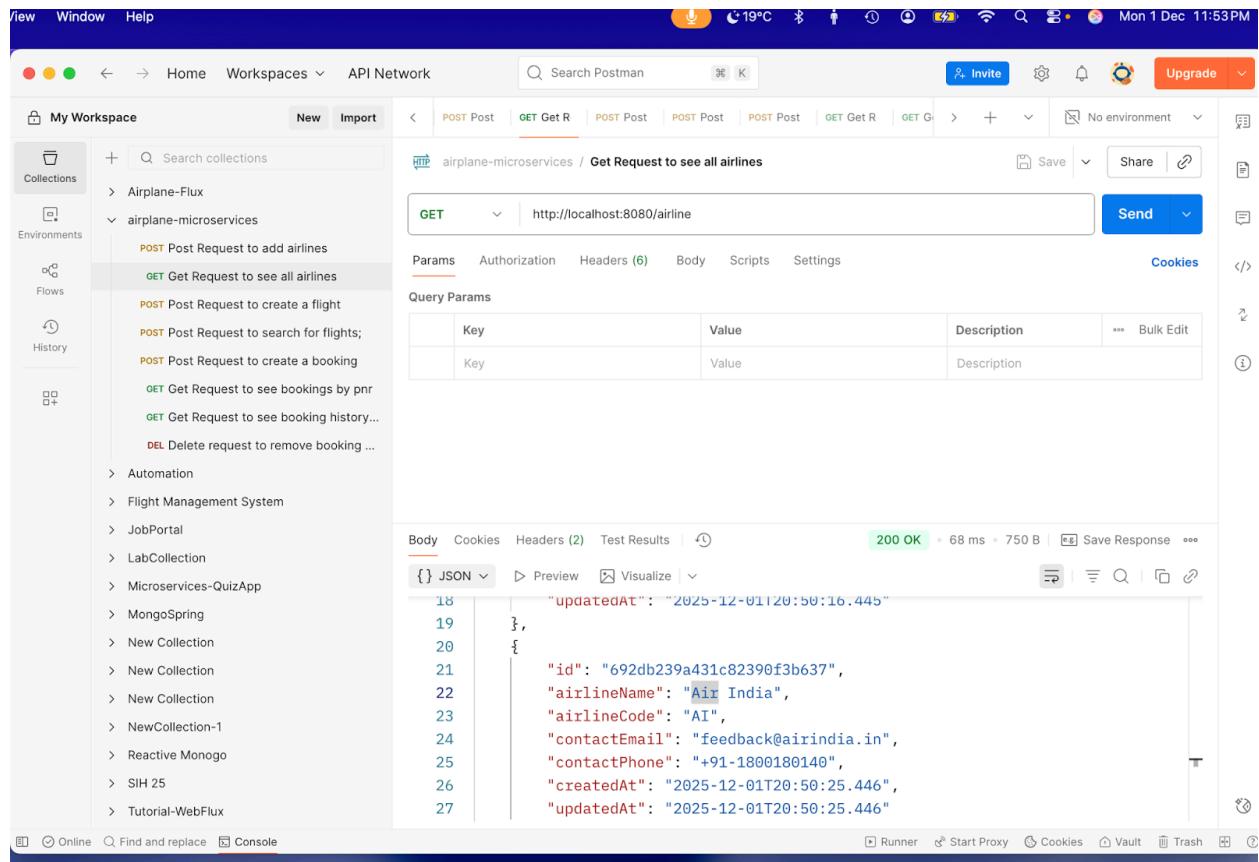
Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (2) Test Results 200 OK 68 ms 750 B Save Response

{ } JSON Preview Visualize

```
18 "updatedAt": "2025-12-01T20:50:16.445"
19 },
20 {
21   "id": "692db239a431c82390f3b637",
22   "airlineName": "Air India",
23   "airlineCode": "AI",
24   "contactEmail": "feedback@airindia.in",
25   "contactPhone": "+91-1800180140",
26   "createdAt": "2025-12-01T20:50:25.446",
27   "updatedAt": "2025-12-01T20:50:25.446"
```

Online Find and replace Console Runner Start Proxy Cookies Vault Trash



View Window Help

Home Workspaces API Network Search Postman 19°C 11:53PM Mon 1 Dec 2023

New Import POST Post GET Get R POST Post POST Post POST Post GET Get R GET G ... No environment Share Upgrade

My Workspace Collections Environments Flows History

airplane-microservices / Post Request to create a flight

POST http://localhost:8080/flight/airline/inventory

Params Authorization Headers (8) Body Scripts Settings Cookies

Body (raw JSON) Schema Beautify

POST Post Request to create a flight

POST Post Request to search for flights;

POST Post Request to create a booking

GET Get Request to see bookings by pnr

GET Get Request to see booking history...

DEL Delete request to remove booking ...

Automation

Flight Management System

JobPortal

LabCollection

Microservices-QuizApp

MongoSpring

New Collection

New Collection

New Collection

NewCollection-1

Reactive Monogo

SIH 25

Tutorial-WebFlux

Online Find and replace Console

Save Send

Body Cookies Headers (2) Test Results 200 OK 19 ms 110 B Save Response

{ } JSON Preview Visualize

```
1 {  
2   "airlineId": "692db239a431c82390f3b637",  
3   "airlineName": "SpiceJet",  
4   "flightNumber": "SG-789",  
5   "fromPlace": "Mumbai",  
6   "toPlace": "Goa",  
7   "departureDateTime": "2025-12-22T07:00:00",  
8   "arrivalDateTime": "2025-12-22T08:30:00",  
9   "price": 2500.00,  
10  "totalSeats": 150,  
11  "availableSeats": 150,  
12  "tripType": "ROUND_TRIP"  
13 }
```

```
1 {  
2   "flightId": "692db2c3a431c82390f3b63a"  
3 }
```

Runner Start Proxy Cookies Vault Trash

View Window Help

Home Workspaces API Network Search Postman 19°C 🔍 Mon 1 Dec 11:53PM

New Import POST Post GET Get R POST Post POST Post GET Get R GET G

My Workspace Collections Environments Flows History

airplane-microservices / Post Request to search for flights;

POST http://localhost:8080/flight/search

Params Authorization Headers (8) Body Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (2) Test Results 200 OK 38 ms 859 B Save Response

{ } JSON ▶ Preview Visualize

18	{
19	"id": "692db27ca431c82390f3b638",
20	"airlineId": "692db21ba431c82390f3b635",
21	"airlineName": "SpiceJet",
22	"flightNumber": "SG-123",
23	"fromPlace": "Bangalore",
24	"toPlace": "Hyderabad",

Online Find and replace Console Runner Start Proxy Cookies Vault Trash

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, Environments, Flows, and History. The main workspace shows a collection named 'airplane-microservices' with several requests listed under it. A specific POST request to 'http://localhost:8080/flight/search' is selected. The request details panel shows the method as POST, the URL, and various tabs like Params, Headers, Body, and Cookies. The Body tab is expanded, showing a JSON response with flight details. Below the request details, there are buttons for Runner, Start Proxy, Cookies, Vault, and Trash. The bottom navigation bar includes Online, Find and replace, and Console.

View Window Help

19°C 🔍 Mon 1 Dec 11:53PM

Home Workspaces API Network Search Postman ⌘ K

New Import < POST Post GET Get R POST Post POST Post POST Post GET Get R GET G > + ↻ No environment ↻ Share ↻

My Workspace Collections Environments Flows History

airplane-microservices / Post Request to create a booking

POST http://localhost:8080/flight/booking/692db27ca431c82390f3b638

Params Authorization Headers (8) Body Scripts Settings Cookies

Query Params

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

Body Cookies Headers (2) Test Results 200 OK 358 ms 92 B Save Response

{ } JSON Preview Visualize

```
1 {  
2 |   "pnr": "PNR41BCE254"  
3 }
```

Runner Start Proxy Cookies Vault Trash

Online Find and replace Console

View Window Help

Home Workspaces API Network

Search Postman

19°C 11:53PM

My Workspace

Collections Environments Flows History

airplane-microservices / Get Request to see bookings by pnr

GET http://localhost:8080/flight/ticket/PNR41BCE254

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (2) Test Results 200 OK 538 ms 525 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "id": "692db32d944a4a19cbad6583",
3   "pnr": "PNR41BCE254",
4   "flightId": "692db27ca431c82390f3b638",
5   "email": "sujaynsv@gmail.com",
```

Online Find and replace Console Runner Start Proxy Cookies Vault Trash

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, Environments, Flows, and History. The main workspace shows a collection named 'airplane-microservices' expanded, revealing several requests like 'Post Request to add airlines', 'Get Request to see all airlines', etc. A specific GET request to 'http://localhost:8080/flight/ticket/PNR41BCE254' is selected. The request details panel shows the method as 'GET', the URL, and various tabs for Params, Authorization, Headers, Body, Scripts, and Settings. Below this, the 'Body' tab is active, displaying a JSON response with fields: id, pnr, flightId, and email. The response body is shown as a JSON object with values: id: "692db32d944a4a19cbad6583", pnr: "PNR41BCE254", flightId: "692db27ca431c82390f3b638", and email: "sujaynsv@gmail.com". The status bar at the bottom indicates the response was successful (200 OK), took 538 ms, and was 525 bytes.

View Window Help

19°C Mon 1 Dec 11:53PM

Home Workspaces API Network Search Postman Invite Share Upgrade

My Workspace New Import < Post GET Get R POST Post POST Post GET Get R GET Get R > + No environment

Collections Environments Flows History

airplane-microservices / Get Request to see booking history by email

Save Share Send

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

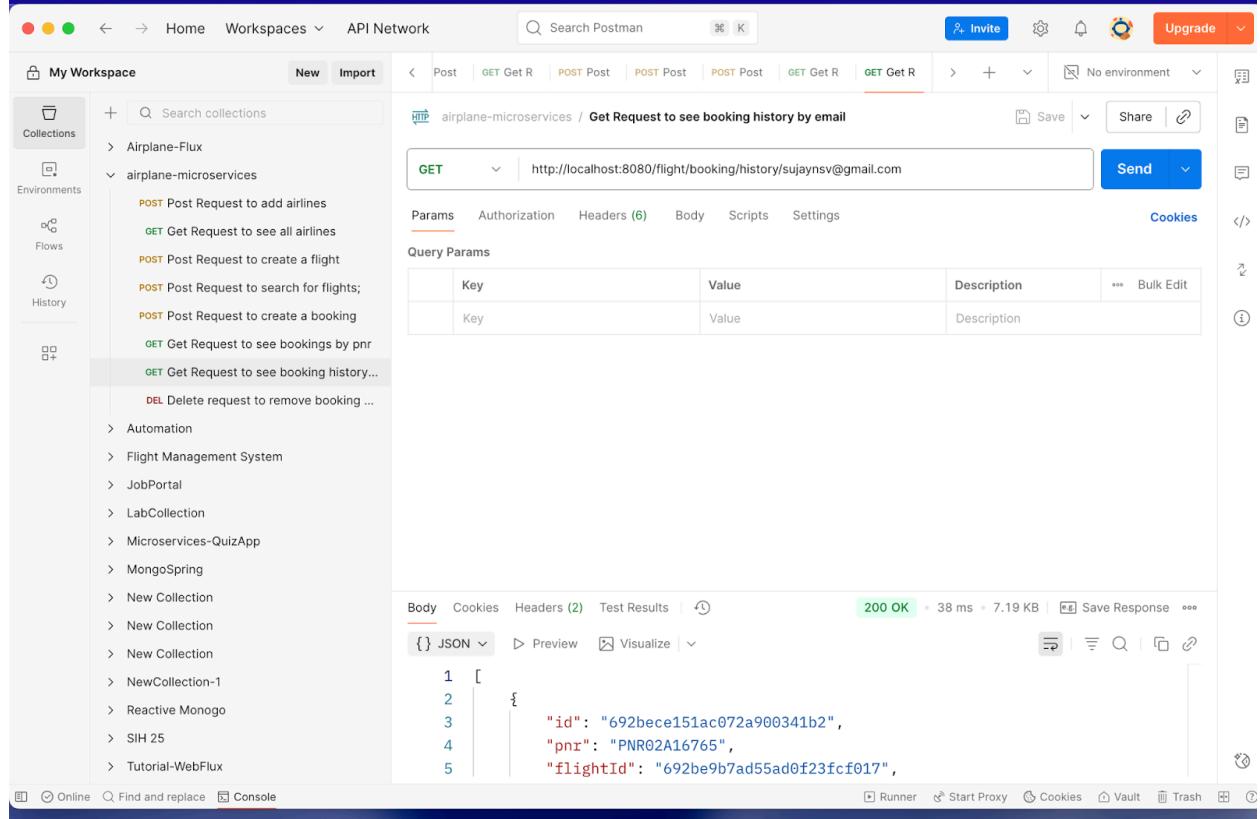
Key	Value	Description
Key	Value	Description

Body Cookies Headers (2) Test Results 200 OK 38 ms 7.19 KB Save Response

{ } JSON Preview Visualize

```
1 [  
2 {  
3   "id": "692bece151ac072a900341b2",  
4   "pnr": "PNR02A16765",  
5   "flightId": "692be9b7ad55ad0f23fcf017",
```

Online Find and replace Console Runner Start Proxy Cookies Vault Trash



View Window Help

Home Workspaces API Network Search Postman Save Share [Invite](#) [Upgrade](#)

My Workspace New Import < Set R POST Post POST Post GET Get R GET Get R DEL Delete > + No environment [Save](#) [Share](#) [Edit](#)

Collections Environments Flows History

+ Q. Search collections

> Airplane-Flux

> airplane-microservices

- [POST Post Request to add airlines](#)
- [GET Get Request to see all airlines](#)
- [POST Post Request to create a flight](#)
- [POST Post Request to search for flights;](#)
- [POST Post Request to create a booking](#)
- [GET Get Request to see bookings by pnr](#)
- [GET Get Request to see booking history...](#)
- [DEL Delete request to remove booking ...](#)

> Automation

> Flight Management System

> JobPortal

> LabCollection

> Microservices-QuizApp

> MongoSpring

> New Collection

> New Collection

> New Collection

> NewCollection-1

> Reactive Monogo

> SIH 25

> Tutorial-WebFlux

HTTP airplane-microservices / Delete request to remove booking of flight

DELETE http://localhost:8080/flight/booking/cancel/PNR41BCE254

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (1) Test Results 200 OK 32 ms 38 B Save Response

Raw Preview Visualize 1

Online Find and replace Console Runner Start Proxy Cookies Vault Trash

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, Environments, Flows, and History. Below that is a tree view of API collections: Airplane-Flux, airplane-microservices, Automation, and others. The airplane-microservices collection is expanded, showing various requests like POST for adding airlines, GET for seeing all airlines, and a DELETE for removing a booking. The main workspace shows a DELETE request for a flight booking with the URL http://localhost:8080/flight/booking/cancel/PNR41BCE254. The response pane shows a single digit '1' under the 'Raw' tab, indicating a successful 200 OK response. The status bar at the bottom shows '32 ms' and '38 B'.

Jacoco Report

RabbitMQ: Queue %2Fbooking.email.queue

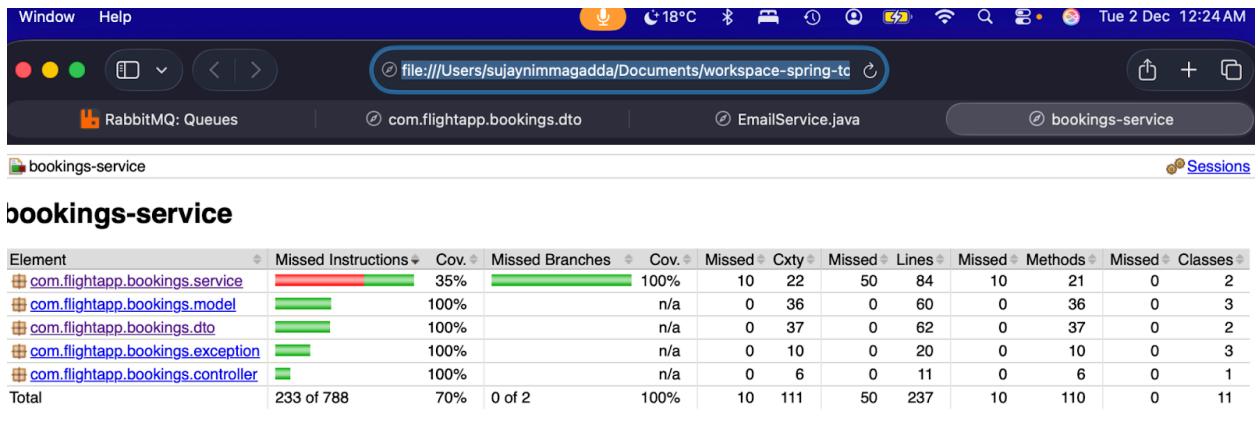
flights-service

Sessions

flights-service

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cqty	Missed	Lines	Missed	Methods	Missed	Classes
com.flighthapp.flights.model	███████	87%	n/a	6	44	8	71	6	44	0	2	
com.flighthapp.flights.dto	██████	77%	n/a	2	9	3	14	2	9	0	1	
com.flighthapp.flights.service	██████████	100%	██████████	100%	0	18	0	55	0	17	0	2
com.flighthapp.flights.controller	██████	100%	n/a	0	16	0	30	0	16	0	2	
com.flighthapp.flights.exception	██████	100%	n/a	0	5	0	11	0	5	0	3	
Total	27 of 565	95%	0 of 2	100%	8	92	11	181	8	91	0	10

Created with JaCoCo 0.8.11.202310140853



RabbitMq

The screenshot shows the RabbitMQ Management Interface running in a web browser on a Mac OS X desktop. The title bar indicates the window is titled "RabbitMQ: Queue %2F/booking.email.queue" and the address bar shows "localhost". The interface is dark-themed.

Overview: Shows a grid of queued messages over the last minute. The legend indicates:

- Ready: 23 (Yellow)
- Unacked: 0 (Blue)
- Total: 23 (Red)

Message rates: Shows message rates over the last minute. The legend indicates:

- Deliver (manual ack): 0.00/s (Yellow)
- Deliver (auto ack): 0.00/s (Blue)
- Consumer ack: 0.00/s (Red)
- Redelivered: 0.00/s (Green)
- Get (manual ack): 0.00/s (Purple)
- Get (auto ack): 0.00/s (Grey)
- Get (empty): 0.00/s (Black)

Details: Displays configuration details for the queue:

Features	arguments: x-queue-type: classic durable: true queue storage version: 2	State	idle
Policy		Consumers	0
Operator policy		Consumer capacity	0%
Effective policy definition			

The screenshot shows a Mac OS X desktop with a browser window open to the RabbitMQ: Queues interface at `localhost`. The browser's title bar includes the URL `RabbitMQ: Queues`, the page title `RabbitMQ: Queues`, and the user `flights-service`. The top right of the window shows the date and time `Mon 1 Dec 11:54PM` and system status like `19°C` and battery level. The main content area displays the RabbitMQ interface with tabs for Overview, Connections, Channels, Exchanges, and Queues and Streams (which is selected). Under Admin, it shows **All queues (1)**. A table provides details for the single queue:

Overview		Messages				Message rates				
Virtual host	Name	Type	Features	State	Ready	Unacked	Total	Incoming	deliver / get	ack
/	booking.email.queue	classic	D Args	running	23	0	23		0.00/s	0.00/s

Below the table, there's a link to **Add a new queue**. At the bottom of the interface, there are links for **HTTP API**, **Documentation**, **Tutorials**, **New releases**, **Commercial edition**, **Commercial support**, **Discussions**, **Discord**, **Plugins**, and a GitHub icon.

Window Help

localhost

RabbitMQ: Queue %2Fbooking.email.queue com.flightapp.bookings.dto

RabbitMQ™ RabbitMQ 4.2.1 Erlang 28.2

Refreshed 2025-12-02 00:25:36 Refresh every 5 seconds

Virtual host All Cluster rabbit@Sais-MacBook-Air-2 User guest Log out

Overview Connections Channels Exchanges Queues and Streams

Admin

Warning: getting messages from a queue is a destructive action. ?

Ack Mode: Nack message requeue true ?

Encoding: Auto string / base64 ?

Messages: 1

Get Message(s)

Message 1

The server reported 22 messages remaining.

Exchange	(AMQP default)
Routing Key	booking.email.queue
Redelivered	•
Properties	priority: 0 delivery_mode: 2 headers: __TypeId__: com.flightapp.bookings.dto.EmailNotification content_encoding: UTF-8 content_type: application/json
Payload	427 bytes Encoding: string <pre>{"to": "sujaynsv@gmail.com", "subject": "Flight Booking Confirmation - PNR: PNR1E35A21C", "body": "Dear Test 2,\n\nYour flight booking</pre>

▶ Move messages

▶ Delete

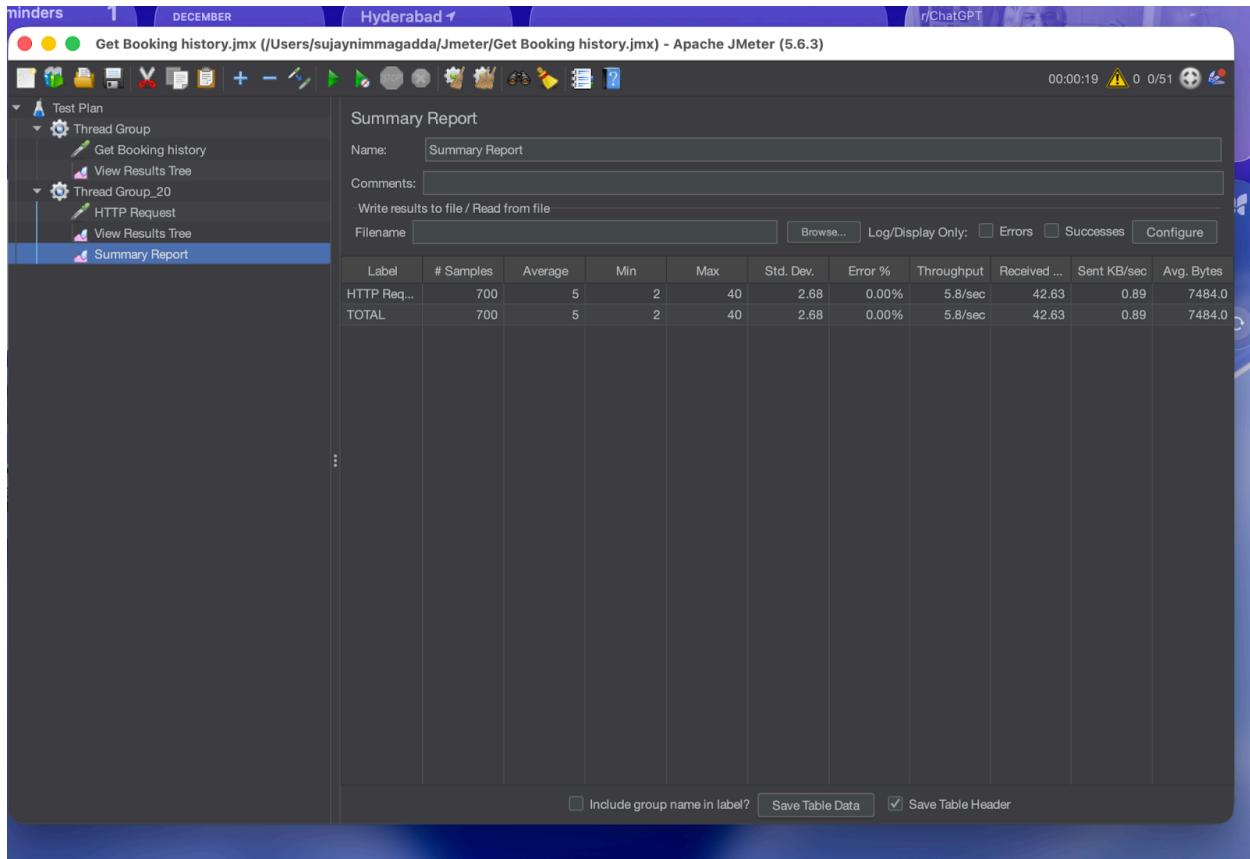
Jmeter

The screenshot shows the Apache JMeter interface version 5.6.3. The title bar indicates the file is 'Get Booking history.jmx' located at '/Users/sujaynimmagadda/Jmeter/Get Booking history.jmx'. The left sidebar displays the 'Test Plan' structure, which includes a 'Thread Group' containing 'Get Booking history' and 'View Results Tree', and a 'Thread Group_20' containing 'HTTP Request' and 'View Results Tree'. The 'Summary Report' item is selected. The main panel shows the 'Summary Report' configuration with 'Name: Summary Report' and 'Comments: Write results to file / Read from file'. A table below provides performance metrics:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received ...	Sent KB/sec	Avg. Bytes
HTTP Req...	200	5	2	40	3.38	0.00%	10.5/sec	76.67	1.61	7484.0
TOTAL	200	5	2	40	3.38	0.00%	10.5/sec	76.67	1.61	7484.0

At the bottom of the report panel, there are three checkboxes: 'Include group name in label?' (unchecked), 'Save Table Data' (unchecked), and 'Save Table Header' (checked).

20 requests



50 requests

SonarQube

The screenshot shows the SonarQube interface for a project named "flight-booking-microservices" at branch "main". The analysis was completed 37 minutes ago. The Quality Gate status is "Passed".

Summary Metrics:

- New Code:** Passed (Green)
- Overall Code:** Passed (Green)

Quality Gate Status: Passed (Green checkmark icon)

Security: 11 Open issues (D)

Reliability: 0 Open issues (A)

Maintainability: 5 Open issues (A)

Accepted Issues: 0

Coverage: A few extra steps are needed for SonarQube Cloud to analyze your code coverage. [Set up coverage analysis](#)

Duplications: 10.6% (Orange circle icon)

Security Hotspots: 0

