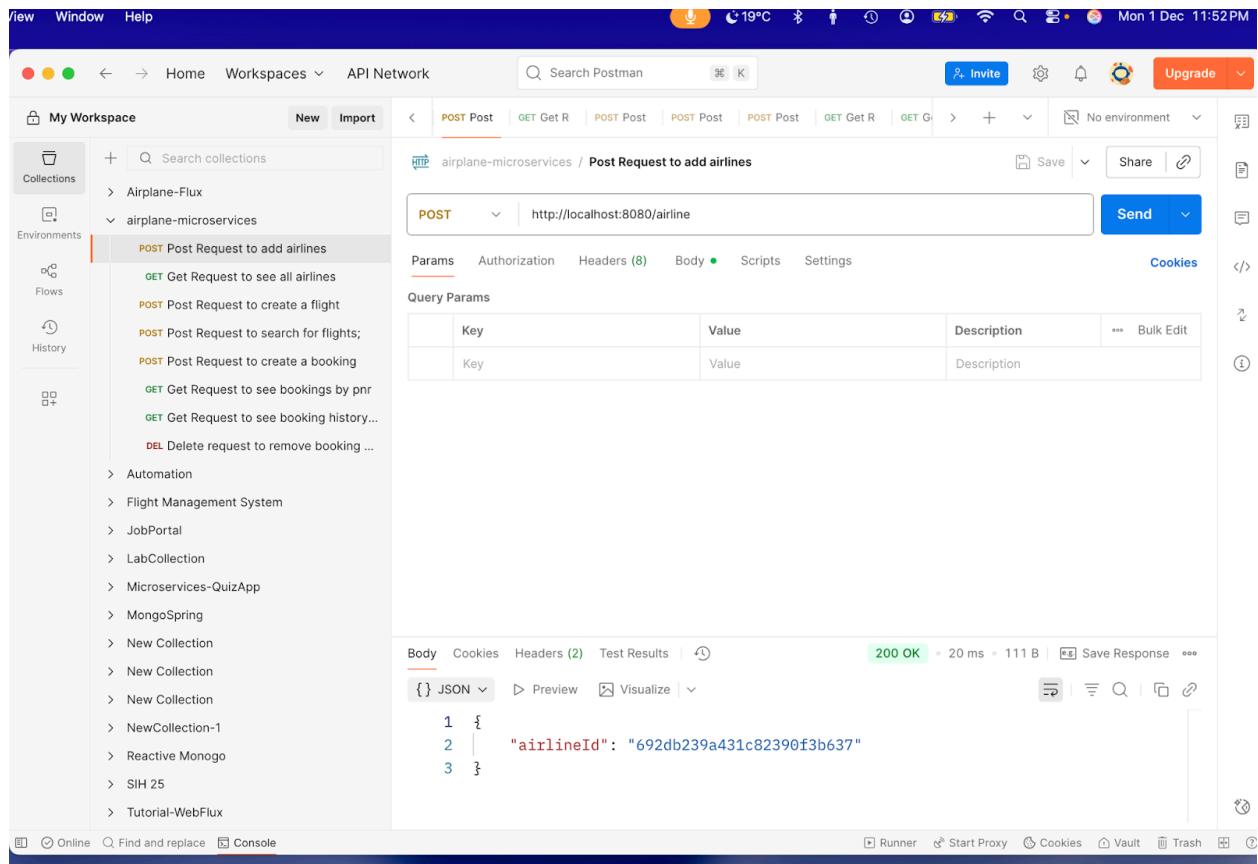


Postman Testing



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

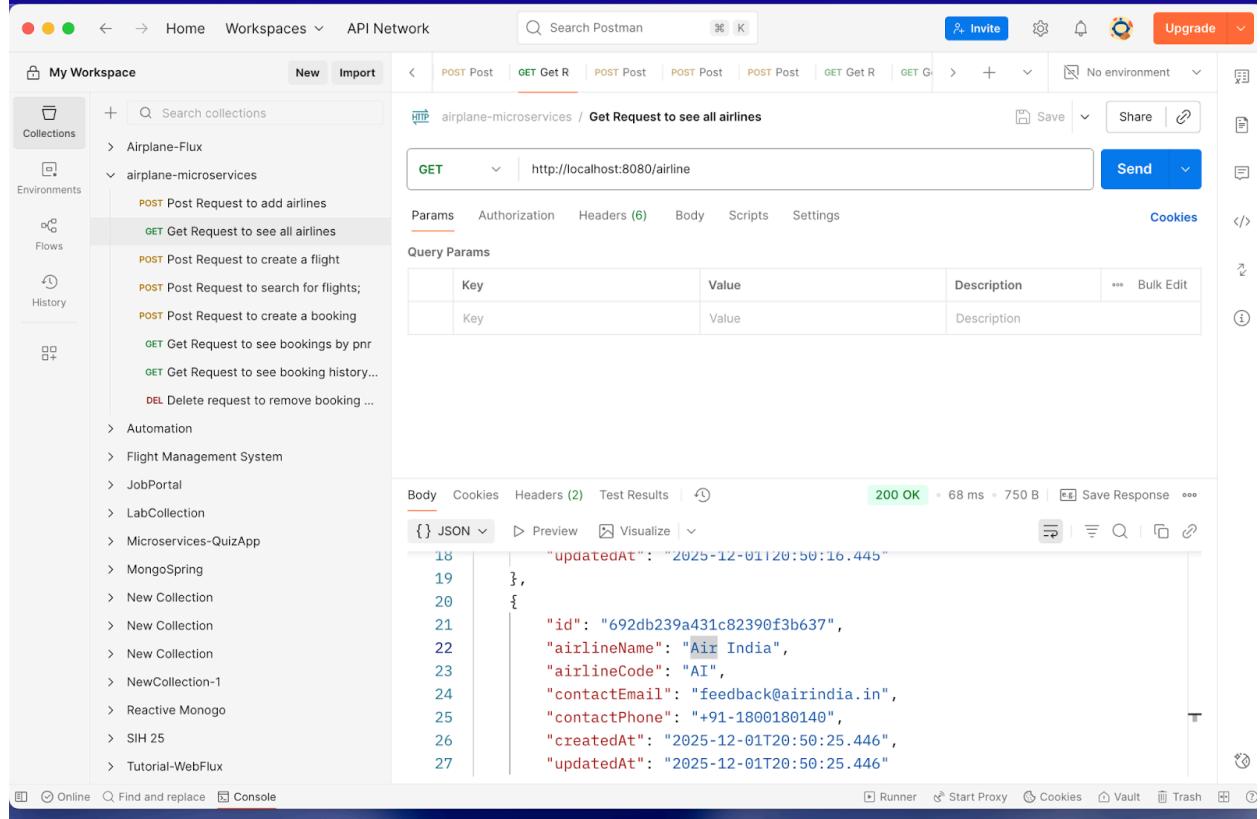
Key	Value	Description
Key	Value	Description

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 1 Mon 1 Dec 11:53PM

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

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)

```
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 }
```

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

{ } JSON Preview Visualize

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

Online Find and replace Console Runner Start Proxy Cookies Vault Trash

The screenshot shows the Postman application interface. The left sidebar displays 'My Workspace' with various collections like 'Airplane-Flux' and 'airplane-microservices'. The main area shows a 'Post Request to create a flight' under the 'airplane-microservices' collection. The request method is 'POST' to 'http://localhost:8080/flight/airline/inventory'. The 'Body' tab is selected, showing a JSON payload with flight details. The response status is '200 OK' with a response time of 19 ms and a body size of 110 B. The response JSON includes a flight ID.

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 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 pane shows the method as POST, the URL, and various tabs for Params, Authorization, Headers, Body, Scripts, and Settings. Below the request details, the response pane displays a 200 OK status with a response time of 38 ms and a body size of 859 B. The response body is shown as JSON, containing flight information with IDs, airline IDs, airline names, flight numbers, from places, and to places. At the bottom, there are navigation links like Online, Find and replace, and Console, along with toolbars for Runner, Start Proxy, Cookies, Vault, and Trash.

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' with several requests listed under it. 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 the request details is a 'Query Params' table. The response panel shows a status of '200 OK' with response time and size information. The response body is displayed as JSON, showing a single object with four properties: id, pnr, flightId, and email. At the bottom, there are navigation links like Online, Find and replace, and Console, along with a toolbar for Runner, Start Proxy, Cookies, Vault, and Trash.

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

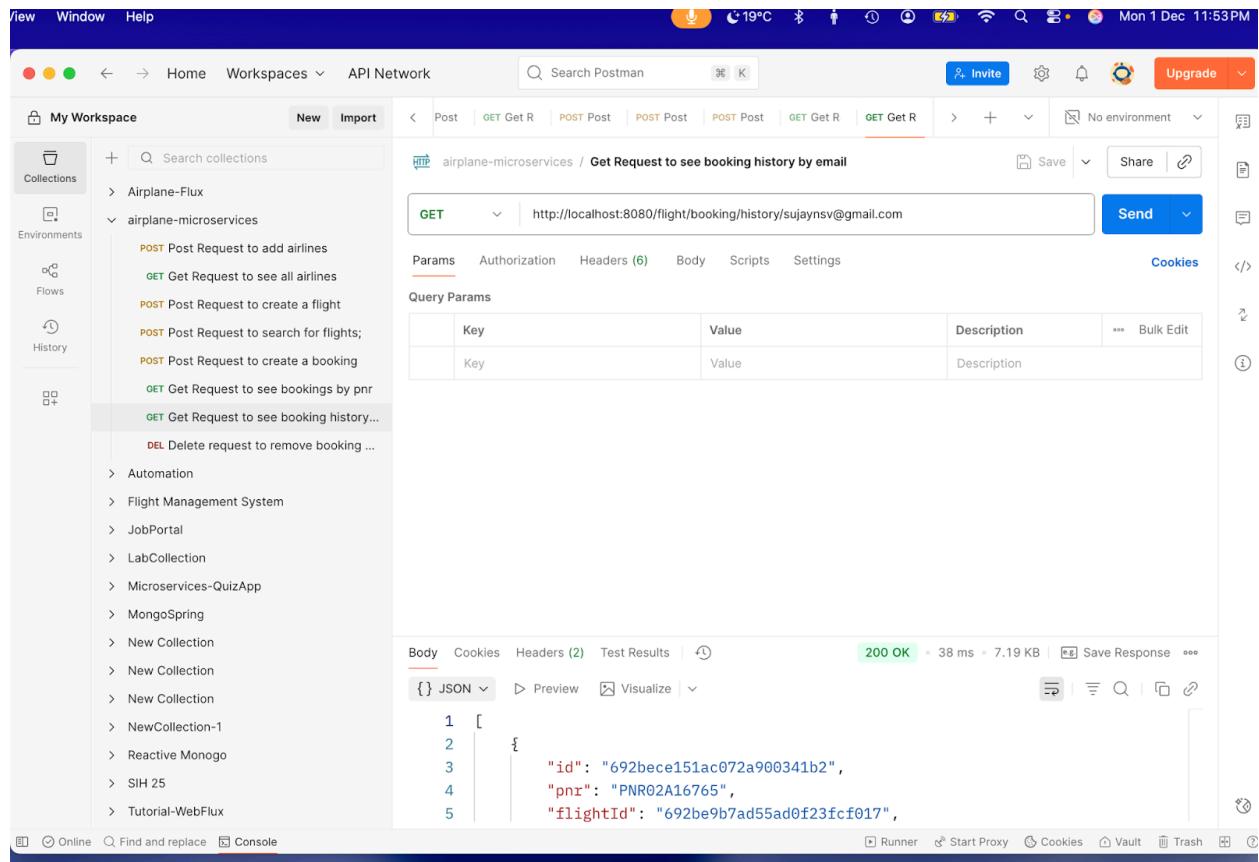
Key	Value	Description
Key	Value	Description

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 [Share](#) [Edit](#)

Collections Environments Flows History

+

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	

Automation Flight Management System JobPortal LabCollection Microservices-QuizApp MongoSpring New Collection New Collection New Collection NewCollection-1 Reactive Monogo SIH 25 Tutorial-WebFlux

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. The main workspace shows a collection named 'airplane-microservices' with a sub-item 'Delete request to remove booking of flight'. This item is highlighted in red. Below it, there's a list of other items like 'Automation', 'Flight Management System', etc. The central area displays a DELETE request to 'http://localhost:8080/flight/booking/cancel/PNR41BCE254'. The 'Headers' tab shows six headers. The 'Body' tab contains the number '1'. At the bottom, there are tabs for 'Raw', 'Preview', and 'Visualize', along with a status bar showing '200 OK', '32 ms', and '38 B'. The bottom navigation bar includes links for 'Online', 'Find and replace', 'Console', 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and a help icon.

Jacoco Report

RabbitMQ: Queue %2Fbooking.email.queue

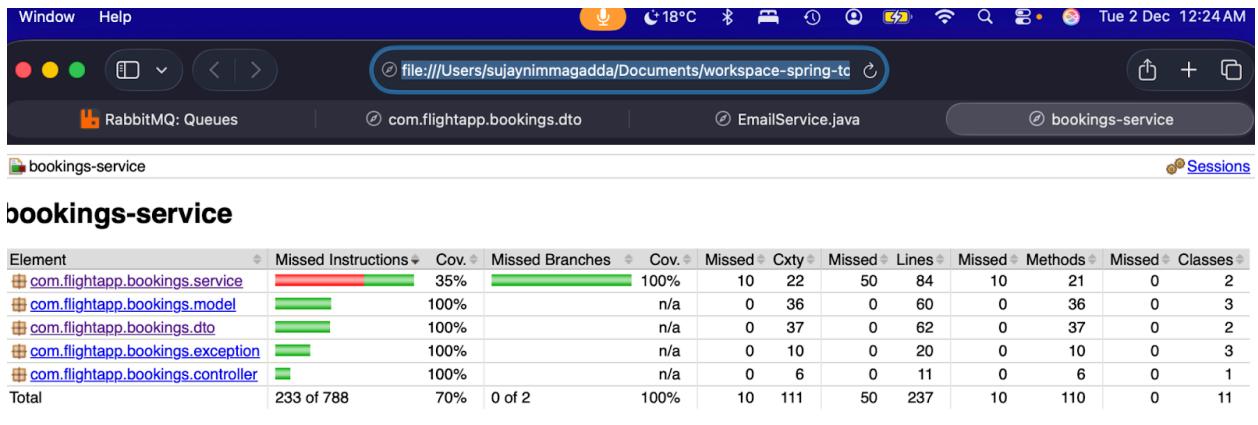
flights-service

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 browser window titled "RabbitMQ: Queue %2F/booking.email.queue". The interface is dark-themed and displays various metrics and configurations for a queue named "flights-service".

Overview: Shows the number of queued messages over the last minute. A chart indicates 23 messages in total, with 0 unacked and 0 ready.

Message rates: Shows message rates per second over the last minute. Rates for Deliver (manual ack), Deliver (auto ack), Consumer ack, Redelivered, Get (manual ack), Get (auto ack), and Get (empty) are all at 0.00/s.

Details: Provides 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 are links for **Add a new queue**, **HTTP API**, **Documentation**, **Tutorials**, **New releases**, **Commercial edition**, **Commercial support**, **Discussions**, **Discord**, **Plugins**, and **GitHub**.

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

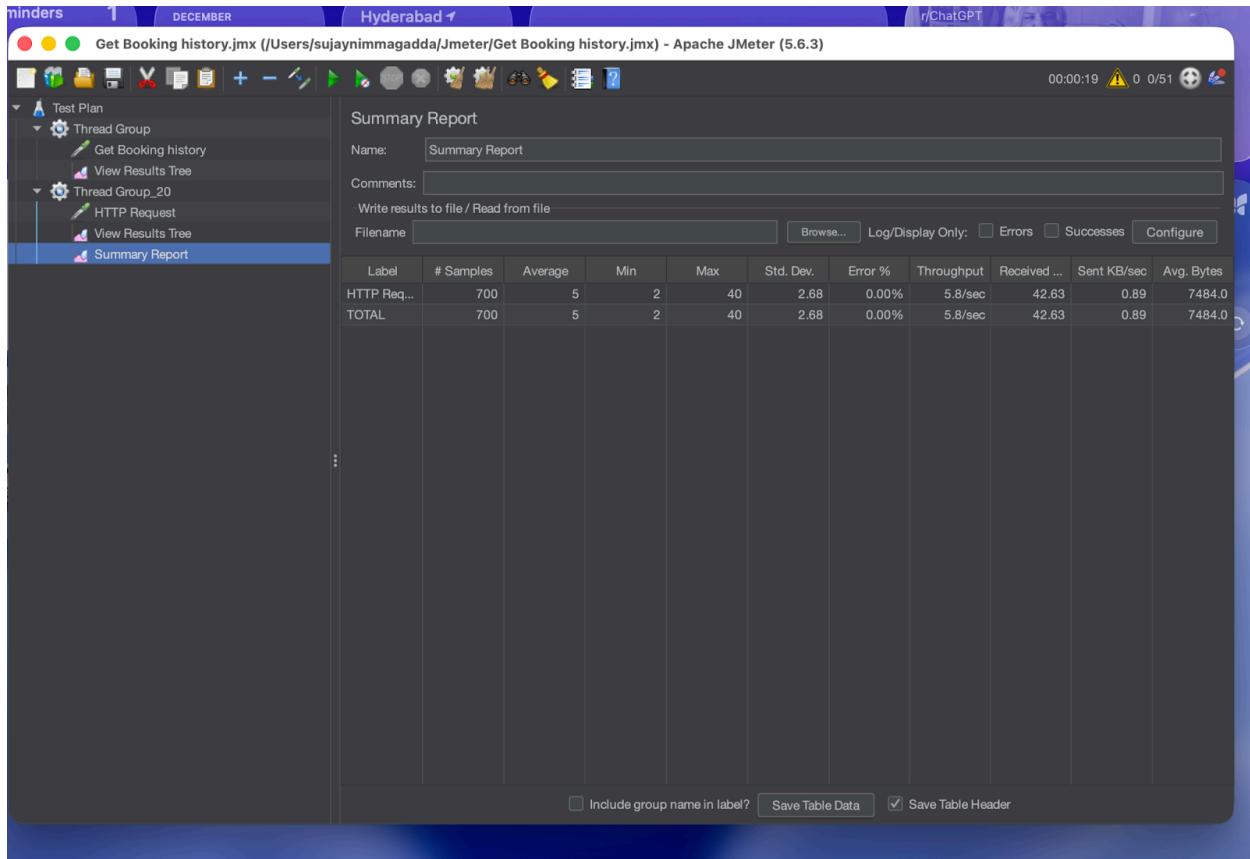
Jmeter

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 dashboard for the project "Sujay Nimmagadda > flight-booking-microservices > main". The dashboard indicates a "Passed" quality gate. Key metrics shown include:

- 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% (No conditions set on 2.4k Lines)

Other visible elements include "New Code" and "Overall Code" tabs, and a "Security Hotspots" section at the bottom.

