

Chubb Assignment 3 Report

Newman report (HTML report of the same can be found in the “newman_report” folder)

```
vishnu fedora ➜ ./newman ➜ newman run Chubb.postman_collection.json
newman
Default
Chubb
vishnu fedora ➜ ➜ jmeter -n -t "/home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/Jmeter
☐ Flight Service.mxml" -l "/home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/results.csv" -e -o
↳ Add flight schedule /home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/HTMLReport"
POST http://localhost:9000/flightservice/api/flight/airline/inventory [201 Created, 484B, 37ms] will be removed in a future
release
↳ Search Flight ConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future
POST http://localhost:9000/flightservice/api/flight/search [200 OK, 2.66kB, 8ms]
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future
release
☐ Booking Service
↳ Search by pnr ConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future
GET http://localhost:9000/bookingservice/api/flight/ticket/PNR17645920003131 [200 OK, 515B, 10ms]
Creating summariser <summary>
↳ Cancel a ticket successfully using /home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/Jmeter te
DELETE http://localhost:9000/bookingservice/api/flight/booking/cancel/PNR17645214757551 [204 No Content, 64B, 21ms]
Starting standalone test @ 2025 Dec 1 19:28:39 IST (1764597519892)
↳ Get details using email Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
GET http://localhost:9000/bookingservice/api/flight/booking/history/vishnuvaradhankr@gmail.com [200 OK, 13.52kB, 13ms]
Tidying up ... @ 2025 Dec 1 19:28:40 IST (1764597520971)
↳ Book ticket of run
POST http://localhost:9000/bookingservice/api/flight/booking/3 [201 Created, 518B, 25ms]
```

Default	executed	failed
↳ test plan iterations	1	0
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release	6	0
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release	0	0
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release	0	0
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release	0	0
total run duration: 174ms		
Created the tree successfully using /home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/Jmeter		
total data received: 17.06kB (approx)		
Starting standalone test @ 2025 Dec 1 19:28:39 IST (1764597519892)		
average response time: 19ms [min: 8ms, max: 37ms, s.d.: 10ms]		

Jmeter Report

Jmeter test using cli Test plan and Full report available on “Jmeter” folder

```
vishnu fedora ➤ ..../Jmeter ➤ jmeter -n -t "/home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/TestPlan.jmx" -l "/home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/results.jtl"
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 /home/vishnu/Documents/Chubb Assignments/FlightBookingMicroservice/Jmeter/TestPlan.jmx
Starting standalone test @ 2025 Dec 1 21:08:20 IST (1764603500402)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary = 120 in 00:00:01 = 114.3/s Avg: 27 Min: 7 Max: 122 Err: 36 (30.00%)
Tidying up ... @ 2025 Dec 1 21:08:21 IST (1764603501515)
... end of run
```



Test and Report information	
Source file	"results.jtl"
Start Time	"12/1/25, 9:08 PM"
End Time	"12/1/25, 9:08 PM"
Filter for display	""

APDEX (Application Performance Index)			
Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.700	500 ms	1 sec 500 ms	Total
0.050	500 ms	1 sec 500 ms	Add Inventory
0.150	500 ms	1 sec 500 ms	Book Ticket
1.000	500 ms	1 sec 500 ms	Search Flight
1.000	500 ms	1 sec 500 ms	Cancel ticket
1.000	500 ms	1 sec 500 ms	Get Ticket history by email
1.000	500 ms	1 sec 500 ms	Get Ticket Details by PNR



Jmeter test using GUI

20 Threads

The screenshot shows the JMeter Test Plan interface. On the left, there's a tree view of the test plan with nodes like 'Test Plan', 'FlightBooking Microservice', 'Add Inventory', 'HTTP Header Manager', 'Search Flight', 'Book Ticket', 'Summary Report', and 'View Results Tree'. The 'Summary Report' node is selected and highlighted in blue. To the right is a 'Summary Report' configuration panel and a detailed summary table.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K...	Sent KB/sec	Avg. Bytes
Add Invent...	20	8	7	26	4.02	95.00%	20.9/sec	3.97	8.46	194.8
Search Flight	20	8	5	12	1.57	0.00%	21.3/sec	55.54	5.82	2673.1
Book Ticket	20	15	11	50	8.15	95.00%	21.3/sec	3.33	13.37	160.4
Get Ticket ...	20	5	5	8	0.83	0.00%	22.2/sec	11.49	3.61	529.0
Get Ticket ...	20	26	14	38	5.42	0.00%	21.7/sec	337.57	3.90	15935.3
Cancel ticket	20	11	9	15	1.45	0.00%	22.3/sec	1.39	5.66	64.0
TOTAL	120	12	5	50	8.13	31.67%	116.8/sec	371.92	37.07	3259.4

50 Threads

The screenshot shows the JMeter Test Plan interface for 50 threads. The tree view and configuration panel are identical to the 20 threads screenshot. The summary table shows the following results:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K...	Sent KB/sec	Avg. Bytes
Add Invent...	50	7	5	26	2.83	98.00%	50.8/sec	9.18	20.57	185.3
Search Flight	50	7	6	12	1.12	0.00%	51.8/sec	135.08	14.15	2672.4
Book Ticket	50	12	8	55	9.27	98.00%	51.8/sec	7.96	32.59	157.3
Get Ticket ...	50	4	3	14	2.15	0.00%	54.5/sec	28.20	8.84	529.6
Get Ticket ...	50	31	26	49	4.17	0.00%	53.4/sec	956.02	9.59	18345.7
Cancel ticket	50	9	7	17	1.51	0.00%	54.7/sec	3.42	13.89	64.0
TOTAL	300	12	3	55	10.09	32.67%	286.8/sec	1024.84	90.98	3659.1

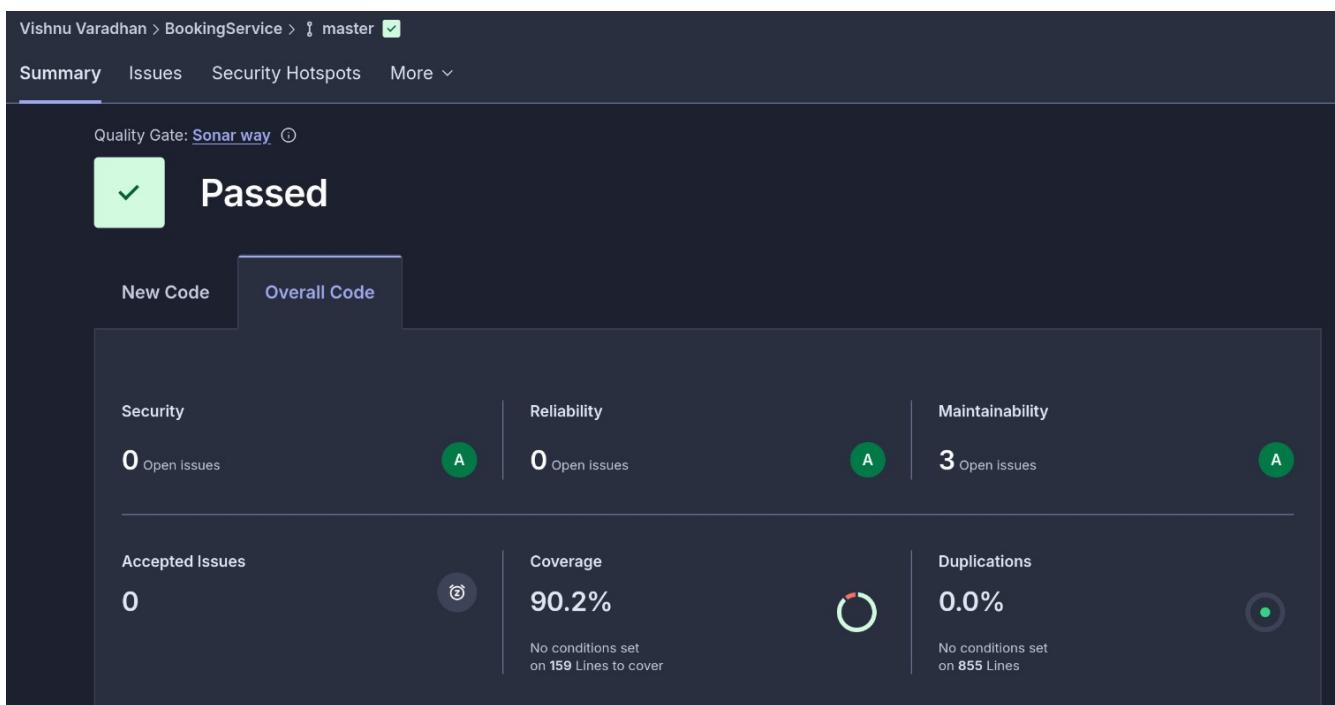
100 Threads

The screenshot shows the JMeter Test Plan interface for 100 threads. The tree view and configuration panel are identical to the previous screenshots. The summary table shows the following results:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K...	Sent KB/sec	Avg. Bytes
Add Invent...	100	7	5	23	2.33	99.00%	100.4/sec	17.86	40.69	182.1
Search Flight	100	8	6	24	1.91	0.00%	101.1/sec	263.93	27.65	2672.9
Book Ticket	100	12	9	46	5.11	98.00%	101.0/sec	14.68	63.53	148.8
Get Ticket ...	100	4	3	10	0.92	0.00%	104.5/sec	54.00	16.94	529.2
Get Ticket ...	100	47	35	59	3.88	0.00%	101.7/sec	2377.69	18.28	23933.7
Cancel ticket	100	9	7	23	1.86	0.00%	106.0/sec	6.63	26.93	64.0
TOTAL	600	14	3	59	14.96	32.83%	565.5/sec	2533.97	179.39	4588.4

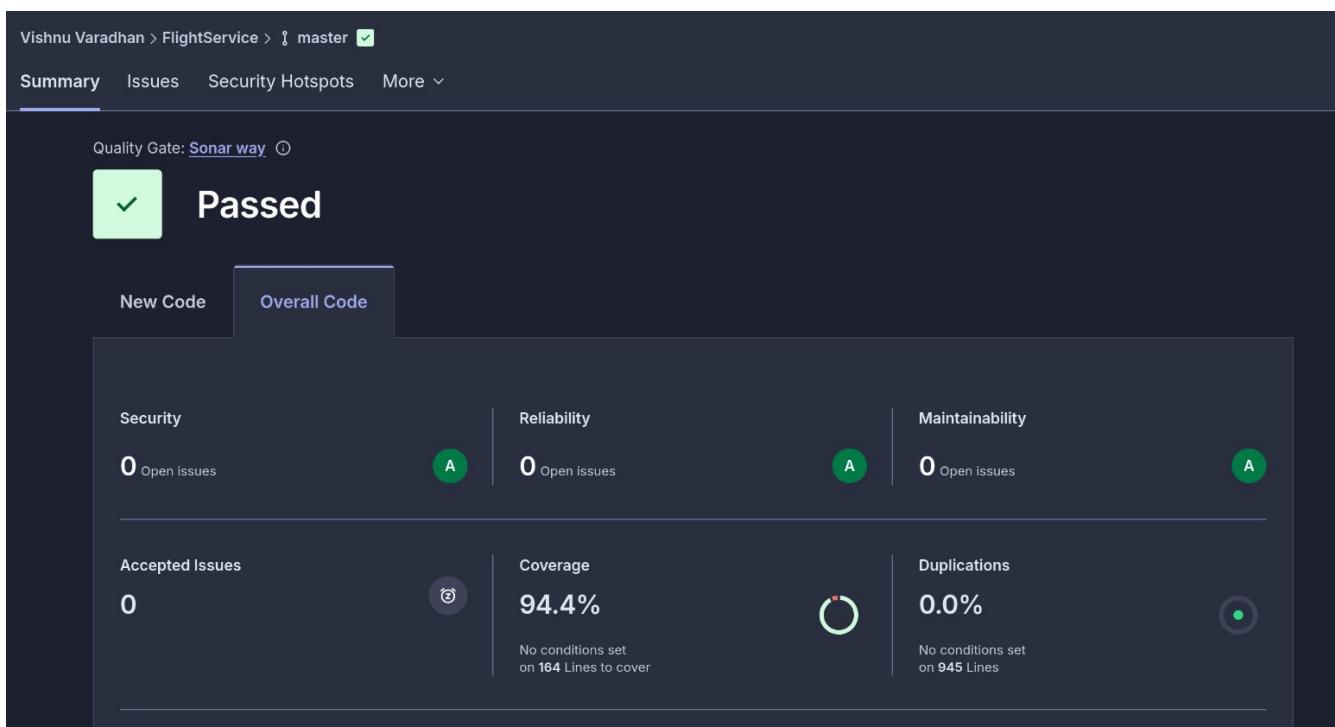
SonarQube Report

Booking Service Report:



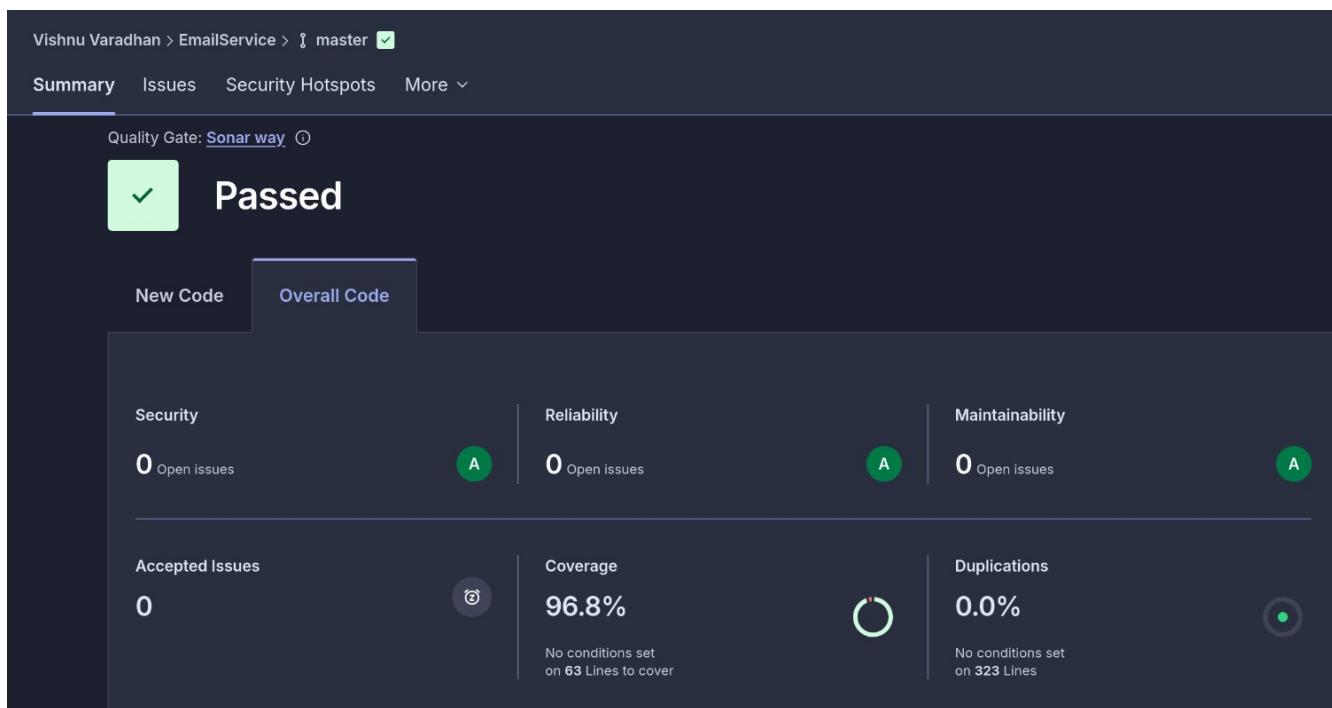
Link: https://sonarcloud.io/summary/overall?id=vi2hnu-1_bookingservice&branch=master

Flight Service Report:



Link: https://sonarcloud.io/summary/overall?id=vi2hnu-1_flightservice&branch=master

Email Service Report:



Link: https://sonarcloud.io/summary/overall?id=vi2hnu-1_emailservice&branch=master

Jacoco Report:

FlightService

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.example.flightservice.controller		82%		n/a	2	11	3	19	2	11	0	3
org.example.flightservice.exception		92%		50%	1	15	2	29	0	14	0	6
org.example.flightservice		37%		n/a	1	2	2	3	1	2	0	1
org.example.flightservice.service.implementation		99%		87%	3	29	0	89	0	17	0	3
org.example.flightservice.dto		100%		100%	0	8	0	15	0	5	0	4
org.example.flightservice.model.enums		100%		n/a	0	2	0	4	0	2	0	2
org.example.flightservice.model.entity		100%		n/a	0	2	0	5	0	2	0	1
Total	26 of 726	96%	4 of 32	87%	7	69	7	164	3	53	0	20



BookingService

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.example.bookingservice.exception		55%		0%	3	13	12	26	2	12	0	5
org.example.bookingservice.service.implementation		98%		95%	1	24	1	98	0	13	0	2
org.example.bookingservice		37%		n/a	1	2	2	3	1	2	0	1
org.example.bookingservice.dto		100%		n/a	0	6	0	11	0	6	0	5
org.example.bookingservice.controller		100%		n/a	0	7	0	15	0	7	0	2
org.example.bookingservice.model.enums		100%		n/a	0	3	0	6	0	3	0	3
Total	55 of 722	92%	3 of 24	87%	5	55	15	159	3	43	0	18

EmailService

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.example.emailservice		37%		n/a	1	2	2	3	1	2	0	1
org.example.emailservice.service.implementation		100%		n/a	0	5	0	44	0	5	0	1
org.example.emailservice.dto		100%		n/a	0	3	0	3	0	3	0	3
org.example.emailservice.model.enums		100%		n/a	0	2	0	4	0	2	0	2
org.example.emailservice.kafka		100%		n/a	0	4	0	9	0	4	0	1
Total	5 of 270	98%	0 of 0	n/a	1	16	2	63	1	16	0	8

Postman (Check full request body and response data at the end of this document)

1) <http://localhost:9000/flightservice/api/flight/airline/inventory>

The screenshot shows the Postman interface for a POST request to <http://localhost:9000/flightservice/api/flight/airline/inventory>. The request body is a JSON object with the following structure:

```
1 {  
2   "flightId": 4,  
3   "fromCityId": 4,  
4   "toCityId": 2,  
5   "departureDate": "2026-07-13",  
6   "departureTime": "2026-07-13T09:30:00",  
7   "price": 4500.50,  
8   "seatsAvailable": 120,  
9   "duration": 180  
10 }
```

The response status is 201 Created, and the response body is a detailed JSON object representing the flight record:

```
1 {  
2   "departureDate": "2026-07-13",  
3   "departureTime": "2026-07-13T09:30:00",  
4   "duration": 180,  
5   "flight": {  
6     "airline": {  
7       "airlineName": "Spice Jet",  
8       "id": 3  
9     },  
10    "columns": 6,  
11    "id": 4,  
12    "name": "SJ404",  
13    "rows": 34,  
14    "year": 2025  
15  },  
16   "fromCity": {  
17     "airportCode": "BLR",  
18     "cityName": "Bengaluru",  
19     "id": 4  
20   },  
21   "id": 60017,  
22   "price": 4500.5,  
23   "seatsAvailable": 204,  
24   "toCity": {
```

2) <http://localhost:9000/flightservice/api/flight/search>

HTTP Chubb / Flight Service / **Search Flight**

POST <http://localhost:9000/flightservice/api/flight/search>

Docs Params Authorization Headers (9) **Body** Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL [JSON](#)

```
1 {  
2   "fromCityCode": "BLR",  
3   "toCityCode": "BOM",  
4   "date": "2026-01-22"  
5 }
```

Body Cookies Headers (3) Test Results | [↻](#) 200 OK

{ } JSON ▾ ▶ Preview  Visualize ▾

```
1 [  
2   {  
3     "departureDate": "2026-01-22",  
4     "departureTime": "2026-01-21T09:30:00",  
5     "duration": 180,  
6     "flight": {  
7       "airline": {  
8         "airlineName": "Spice Jet",  
9         "id": 3  
10      },  
11      "columns": 6,  
12      "id": 5,  
13      "name": "SJ505",  
14      "rows": 26,  
15      "year": 2025  
16    },  
17    "fromCity": {  
18      "airportCode": "BLR",  
19      "cityName": "Bengaluru",  
20      "id": 4  
21    },  
22    "id": 10004,  
23    "price": 4500.5,  
24    "seatsAvailable": 120,
```

3) <http://localhost:9000/bookingservice/api/flight/booking/3>

HTTP Chubb / Booking Service / Book ticket

POST <http://localhost:9000/bookingservice/api/flight/booking/3>

Docs Params Authorization Headers (9) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** ▾

```
1  {
2    "user": {
3      "id": 1,
4      "name": "Vinod Patel",
5      "email": "vishnuvaradhankr@gmail.com",
6      "gender": "MALE"
7    },
8    "scheduleId": 8,
9    "returnTripId": null,
10   "passengers": [
11     {
12       "name": "Ramesh Kumar",
13       "gender": "MALE",
14       "meal": "VEG",
15       "seatPos": "2C"
16     },
17     {
18       "name": "Anita Singh",
19       "gender": "FEMALE",
20       "meal": "NONVEG",
21       "seatPos": "2A"
22     }
23   ]
24 }
```

{ } JSON ▾ ▶ Preview 🖼 Visualize | ▾

```
2   "bookedByUsers": {
3     "email": "vishnuvaradhankr@gmail.com",
4     "gender": "MALE",
5     "id": 1,
6     "name": "Vinod Patel"
7   },
8   "id": 20017,
9   "passengers": [
10    {
11      "gender": "MALE",
12      "id": 20032,
13      "mealOption": "VEG",
14      "name": "Ramesh Kumar",
15      "seatPosition": "2C"
16    },
17    {
18      "gender": "FEMALE",
19      "id": 20033,
20      "mealOption": "NONVEG",
21      "name": "Anita Singh",
22      "seatPosition": "2A"
23    }
24  ],
25  "pnr": "PNR17645937139851",
```

4) <http://localhost:9000/bookingservice/api/flight/ticket/PNR17645920003131>

GET <http://localhost:9000/bookingservice/api/flight/ticket/PNR17645920003131>

☰ Docs Params Authorization Headers (7) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL **Text** ▾

1 Context Add + New Alert AT

Body Cookies Headers (3) Test Results | ⏪ 200 OK

{ } JSON ▾ ▶ Preview ⌘ Visualize ▾

```
1 {
2   "bookedByUsers": {
3     "email": "vishnuvaradhankr@gmail.com",
4     "gender": "MALE",
5     "id": 1,
6     "name": "Vinod Patel"
7   },
8   "id": 20013,
9   "passengers": [
10    {
11      "gender": "MALE",
12      "id": 20024,
13      "mealOption": "VEG",
14      "name": "Ramesh Kumar",
15      "seatPosition": "2C"
16    },
17    {
18      "gender": "FEMALE",
19      "id": 20025,
20      "mealOption": "NONVEG",
21      "name": "Anita Singh"
22    }
23  ]
24}
```

5) <http://localhost:9000/bookingservice/api/flight/booking/history/vishnuvaradhankr@gmail.com>

HTTP Chubb / Booking Service / Get details using email

GET http://localhost:9000/bookingservice/api/flight/booking/history/vishnuvaradhankr@gmail.com

☰ Docs Params Authorization Headers (7) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL **Text** ▾

1 Click to add a comment

Body Cookies Headers (4) Test Results | ⏱ 200 OK

{ } JSON ▾ ▷ Preview 📈 Visualize ▾

```
1 [  
2 {  
3     "bookedByUsers": {  
4         "email": "vishnuvaradhankr@gmail.com",  
5         "gender": "MALE",  
6         "id": 1,  
7         "name": "Vinod Patel"  
8     },  
9     "id": 2,  
10    "passengers": [  
11        {  
12            "gender": "MALE",  
13            "id": 1,  
14            "mealOption": null,  
15            "name": "Ramesh Kumar",  
16            "seatPosition": "4A"  
17        },  
18        {  
19            "gender": "FEMALE",  
20            "id": 2,  
21            "mealOption": "NONVEG"
```

6) <http://localhost:9000/bookingservice/api/flight/booking/cancel/PNR17645214757551>

HTTP Chubb / Booking Service / Cancel a ticket

DELETE ▾ http://localhost:9000/bookingservice/api/flight/booking/cancel/PNR17645214757551

☰ Docs Params Authorization Headers (7) Body Scripts Settings

Body Cookies Headers (1) Test Results | ⏱ 204 No Content • 19 ms

Raw ▾ ▷ Preview ⚡ Visualize ▾

1 |

A screenshot of the Postman application interface. At the top, there's a header bar with the text "HTTP Chubb / Booking Service / Cancel a ticket". Below it is a search bar containing the URL "http://localhost:9000/bookingservice/api/flight/booking/cancel/PNR17645214757551". The main area shows a "DELETE" method selected. There are tabs for "Docs", "Params", "Authorization", "Headers (7)", "Body", "Scripts", and "Settings". The "Headers" tab is active, showing one entry. Below the tabs, there are buttons for "Raw", "Preview", and "Visualize". On the right side, the status is shown as "204 No Content" with a timestamp of "19 ms". At the bottom, there's a text input field with the number "1" followed by a cursor.

API Request body and example response:

1) <http://localhost:9000/flightservice/api/flight/airline/inventory> (add flight)

Request:

```
{  
  "flightId": 5,  
  "fromCityId": 4,  
  "toCityId": 2,  
  "departureDate": "2026-07-18",  
  "departureTime": "2026-07-18T09:30:00",  
  "price": 4500.50,  
  "seatsAvailable": 120,  
  "duration": 180  
}
```

Response:

```
{  
  "departureDate": "2026-07-19",  
  "departureTime": "2026-07-19T09:30:00",  
  "duration": 180,  
  "flight": {  
    "airline": {  
      "airlineName": "Spice Jet",  
      "id": 3  
    },  
    "columns": 6,  
    "id": 5,  
    "name": "SJ505",  
    "rows": 26,  
    "year": 2025  
  },  
  "fromCity": {  
    "airportCode": "BLR",  
    "cityName": "Bengaluru",  
    "id": 4  
  },  
  "id": 70015,  
  "price": 4500.5,  
  "seatsAvailable": 156,  
  "toCity": {  
    "airportCode": "BOM",  
    "cityName": "Mumbai",  
    "id": 2  
  }  
}
```

2) <http://localhost:9000/flightservice/api/flight/search> (Search for flight)

Request:

```
{  
  "fromCityCode": "BLR",  
  "toCityCode": "BOM",  
  "date": "2026-01-22"  
}
```

Response:

```
[  
  {  
    "departureDate": "2026-01-22",  
    "departureTime": "2026-01-21T09:30:00",  
    "duration": 180,  
    "flight": {  
      "airline": {  
        "airlineName": "Spice Jet",  
        "id": 3  
      },  
      "columns": 6,  
      "id": 5,  
      "name": "SJ505",  
      "rows": 26,  
      "year": 2025  
    },  
    "fromCity": {  
      "airportCode": "BLR",  
      "cityName": "Bengaluru",  
      "id": 4  
    },  
    "id": 10004,  
    "price": 4500.5,  
    "seatsAvailable": 120,  
    "toCity": {  
      "airportCode": "BOM",  
      "cityName": "Mumbai",  
      "id": 2  
    }  
  }  
]
```

3) <http://localhost:9000/bookingservice/api/flight/booking/3> (book ticket)

Request:

```
{  
  "user": {  
    "id": 1,  
    "name": "Vinod Patel",  
    "email": "vishnuvaradhankr@gmail.com",  
    "gender": "MALE"  
  },  
  "scheduleId": 10,  
  "returnTripId": null,  
  "passengers": [  
    {  
      "name": "Ramesh Kumar",  
      "gender": "MALE",  
      "meal": "VEG",  
      "seatPos": "13C"  
    }  
  ]  
}
```

Response:

```
{  
  "bookedByUsers": {  
    "email": "vishnuvaradhankr@gmail.com",  
    "gender": "MALE",  
    "id": 1,  
    "name": "Vinod Patel"  
  },  
  "id": 30023,  
  "passengers": [  
    {  
      "gender": "MALE",  
      "id": 30004,  
      "mealOption": "VEG",  
      "name": "Ramesh Kumar",  
      "seatPosition": "13C"  
    }  
  ],  
  "pnr": "PNR17646382318681",  
  "returnTripScheduleId": null,  
  "scheduleId": 10,  
  "status": "BOOKED"  
}
```

4) <http://localhost:9000/bookingservice/api/flight/ticket/PNR17645920003131> (search using pnr)

Response:

```
{  
  "bookedByUsers": {  
    "email": "vishnuvaradhankr@gmail.com",  
    "gender": "MALE",  
    "id": 1,  
    "name": "Vinod Patel"  
  },  
  "id": 30023,  
  "passengers": [  
    {  
      "gender": "MALE",  
      "id": 30004,  
      "mealOption": "VEG",  
      "name": "Ramesh Kumar",  
      "seatPosition": "13C"  
    }  
  ],  
  "pnr": "PNR17646382318681",  
  "returnTripScheduleId": null,  
  "scheduleId": 10,  
  "status": "BOOKED"  
}
```

5)

<http://localhost:9000/bookingservice/api/flight/booking/history/vishnuvaradhankr@gmail.com>
(Search history using email)

Response:

```
[{
  "bookedByUsers": {
    "email": "vishnuvaradhankr@gmail.com",
    "gender": "MALE",
    "id": 1,
    "name": "Vinod Patel",
    "id": 2,
    "passengers": [
      {"gender": "FEMALE",
       "id": 2,
       "mealOption": "NONVEG",
       "name": "Anita Singh",
       "seatPosition": "5B"
     }
    ],
    "pnr": "PNR17643280656201",
    "returnTripScheduleId": null,
    "scheduleId": 1,
    "status": "CANCELED"
  },
  {
    "bookedByUsers": {
      "email": "vishnuvaradhankr@gmail.com",
      "gender": "MALE",
      "id": 1,
      "name": "Vinod Patel"
    },
    "id": 4,
    "passengers": [
      {
        "gender": "MALE",
        "id": 3,
        "mealOption": null,
        "name": "Ramesh Kumar",
        "seatPosition": null
      }
    ],
    "pnr": "PNR17643286947351",
    "returnTripScheduleId": null,
    "scheduleId": 2,
    "status": "BOOKED"
  }
}]
```

6) <http://localhost:9000/bookingservice/api/flight/booking/cancel/PNR17646373940441>
(cancellation of ticket)

Response: 204 No content