

# **FLIGHT BOOKING SYSTEM**

**OUTPUTS, REPORTS AND ANALYSIS**

**BY ANANYA NAYAK**

# SonarQube Analysis Reports

## Running analysis :

```
C:\Windows\System32\cmd.e <--> + - X
[INFO] 22:26:30.815 Architecture analysis is enabled with the following features: legacy
[INFO] 22:26:30.815 * Protobuf reading starting | memory total=1384 | free=1001 | used=382 (MB)
[INFO] 22:26:30.816 * Reading SonarArchitecture IR data from directory "D:\CHUBB_Training\FlightBookingSystem_WebFlux\FlightBookingSystem_WebFlux\target\sonar\architecture.java"
[INFO] 22:26:30.794 * Files successfully loaded: "45" out of "45"
[INFO] 22:26:30.795 * Protobuf reading complete | memory total=1384 | free=982 | used=401 (MB)
[INFO] 22:26:30.827 * Build architecture.graph.java.namespace_graph.default_perspective hierarchy graph complete (filtered=false) | memory total=1384 | free =974 | used=406 (MB)
[INFO] 22:26:30.857 Sensor JavaArchitectureSensor [architecture] (done) | time=875ms
[INFO] 22:26:30.857 Sensor Zero Coverage Sensor
[INFO] 22:26:30.861 Sensor Zero Coverage Sensor (done) | time=4ms
[INFO] 22:26:30.862 Sensor Java CPD Block Indexer
[INFO] 22:26:30.958 Sensor Java CPD Block Indexer (done) | time=88ms
[INFO] 22:26:30.968 Inferred api base url 'https://api.sonarcloud.io' from host url 'https://sonarcloud.io'.
[INFO] 22:26:31.635 ----- Gather SCA dependencies on project
[INFO] 22:26:31.635 Checking if SCA is enabled for organization ananya
[INFO] 22:26:32.014 Dependency analysis skipped
[INFO] 22:26:32.017 CPD Executor 17 files had no CPD blocks
[INFO] 22:26:32.017 CPD Executor Calculating CPD for 22 files
[INFO] 22:26:32.034 CPD Executor CPD calculation finished (done) | time=16ms
[INFO] 22:26:32.056 SCM Publisher SCM provider for this project is: git
[INFO] 22:26:32.057 SCM Publisher 8 source files to be analyzed
[INFO] 22:26:32.735 SCM Publisher 8/8 source files have been analyzed (done) | time=677ms
[INFO] 22:26:33.822 Analysis report generated in 244ms, dir size=468 KB
[INFO] 22:26:34.080 Analysis report compressed in 224ms, zip size=168 KB
[INFO] 22:26:35.063 Analysis report uploaded in 983ms
[INFO] 22:26:35.066 ANALYSIS SUCCESSFUL, you can find the results at: https://sonarcloud.io/dashboard?id=isananya_FlightBookingSystem_WebFlux
[INFO] 22:26:35.066 Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
[INFO] 22:26:35.066 More about the report processing at https://sonarcloud.io/api/ce/task?id=AZq2y7JYj9z-iXtxRZ0
[INFO] 22:26:35.069 ----- Upload SCA dependency files
[INFO] 22:26:36.632 Sensor cache published successfully
[INFO] 22:26:36.786 Analysis total time: 31.803 s
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:15 min
[INFO] Finished at: 2025-11-24T22:26:36+05:30
[INFO] -----
```

D:\CHUBB\_Training\FlightBookingSystem\_WebFlux\FlightBookingSystem\_WebFlux>

## Before Improvement :

https://sonarcloud.io/summary/overall?id=isananya\_FlightBookingSystem\_WebFlux&branch=main

Main Branch Summary

1.7k Lines of Code · Version 0.0.1-SNAPSHOT · Last analysis 6 minutes ago · 26da1c55

Quality Gate: Sonar way

Not computed

Next scan will generate a Quality Gate.

Security	Reliability	Maintainability
2 Open issues	17 Open issues	101 Open issues

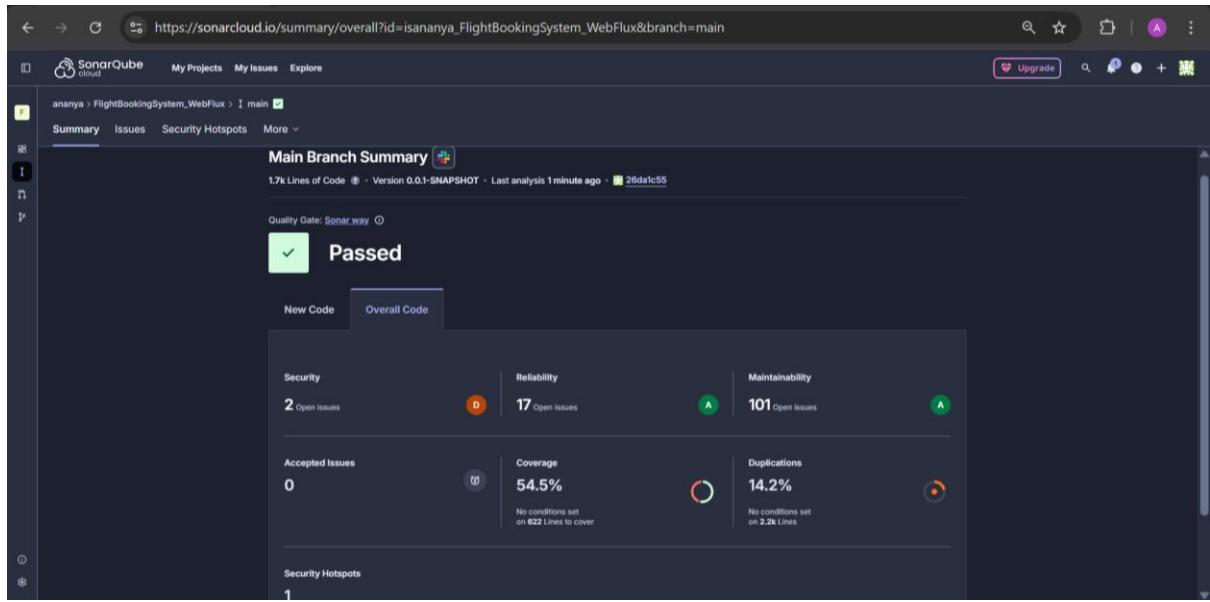
Accepted Issues	Coverage	Duplications
0	0.0%	14.4%

No conditions set on 564 Lines to cover

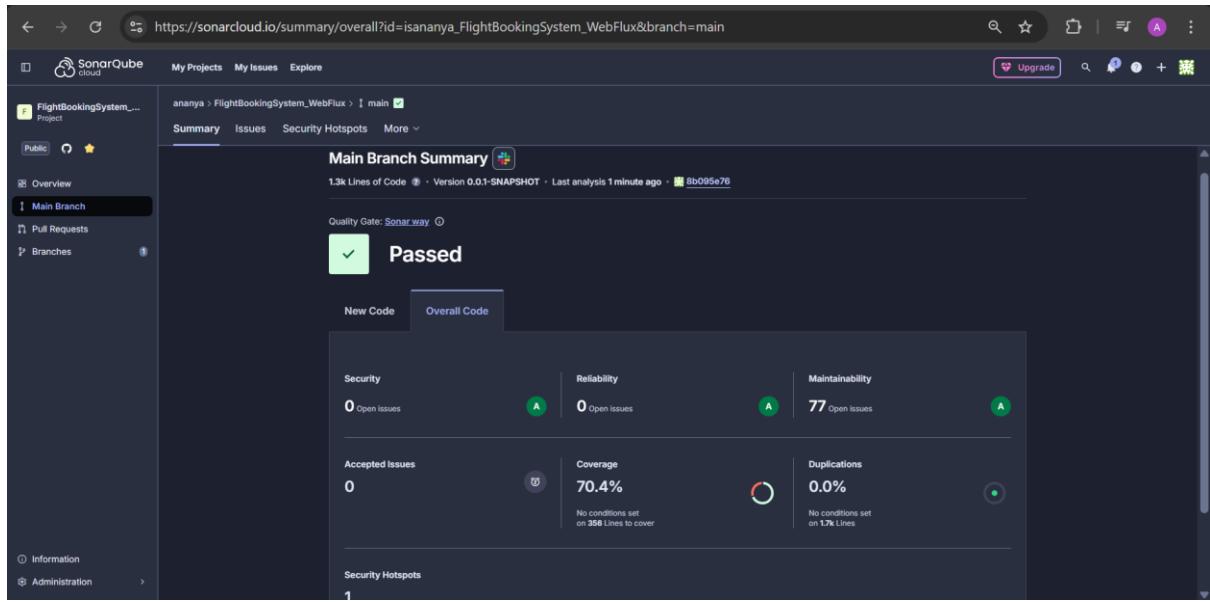
Security Hotspots

1

After setting up coverage (jacoco) :



## Final Analysis



# JMeter Summary Reports

## 20 samples :

Summary Report										
Name:	Summary Report									
Comments:										
Write results to file / Read from file										
Filename	<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
Add User	20	5	4	16	2.55	0.00%	20.9/sec	2.21	7.70	108.0
Add Flight	20	3	2	5	1.02	0.00%	21.2/sec	1.83	7.99	88.5
Add Schedule	20	4	2	8	1.20	100.00%	21.2/sec	244.19	7.69	11815.0
Search Flights	20	2	2	4	0.62	0.00%	21.2/sec	8.16	7.34	395.0
Add Booking	20	4	3	6	0.81	0.00%	21.1/sec	0.78	14.81	38.0
Get ticket usi...	20	6	4	10	1.32	0.00%	21.1/sec	11.41	3.68	555.0
Booking His...	20	7	5	10	1.26	0.00%	21.1/sec	11.41	4.03	555.0
Cancel Booki...	20	2	1	4	0.62	100.00%	21.2/sec	0.97	4.32	47.0
TOTAL	160	4	1	16	2.14	25.00%	162.3/sec	269.43	55.30	1700.2

## 50 samples:

Summary Report										
Name:	Summary Report									
Comments:										
Write results to file / Read from file										
Filename	<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
Add User	50	5	3	17	1.88	0.00%	50.8/sec	5.35	18.72	108.0
Add Flight	50	4	3	12	1.49	0.00%	51.4/sec	4.46	19.38	88.5
Add Schedule	50	5	4	8	1.21	100.00%	51.5/sec	594.13	18.71	11815.0
Search Flights	50	3	2	5	0.75	0.00%	51.7/sec	19.92	17.91	395.0
Add Booking	50	5	4	8	1.04	0.00%	51.7/sec	1.92	36.22	38.0
Get ticket usi...	50	7	5	14	1.55	0.00%	51.7/sec	28.00	9.03	555.0
Booking His...	50	7	5	12	1.47	0.00%	51.8/sec	28.05	9.91	555.0
Cancel Booki...	50	2	1	6	0.89	100.00%	52.2/sec	2.40	10.66	47.0
TOTAL	400	5	1	17	2.24	25.00%	395.3/sec	656.28	134.73	1700.2

## 100 Samples :

Summary Report										
Name:	Summary Report									
Comments:										
Write results to file / Read from file										
Filename	<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
Add User	100	17	3	63	12.71	1.00%	97.3/sec	21.31	35.89	224.4
Add Flight	100	15	3	58	11.80	0.00%	95.8/sec	8.32	36.10	88.9
Add Schedule	100	16	3	39	10.63	100.00%	93.0/sec	1073.31	33.79	11815.0
Search Flights	100	10	2	51	8.14	0.00%	92.5/sec	35.68	32.07	395.0
Add Booking	100	23	4	61	15.03	0.00%	90.3/sec	3.35	63.34	38.0
Get ticket usi...	100	28	6	79	18.09	0.00%	87.3/sec	47.29	15.25	555.0
Booking His...	100	27	6	96	18.98	0.00%	86.7/sec	46.97	16.59	555.0
Cancel Booki...	100	8	2	46	6.96	100.00%	86.8/sec	3.98	17.72	47.0
TOTAL	800	18	2	96	15.10	25.12%	664.5/sec	1112.69	226.52	1714.8

# Postman Screenshots

## Add User:

The screenshot shows a Postman request to `http://localhost:9090/signup` using a POST method. The request body is a JSON object with fields: name, emailId, mobileNo, gender, age, and password. The response status is 201 Created, and the response body contains a unique identifier: `692467c9c1155f4c5682eae8`.

```
1 {
2   "name": "ananya",
3   "emailId": "ananya@example.com",
4   "mobileNo": "1234567890",
5   "gender": "Female",
6   "age": 21,
7   "password": "SecurePass123!"
8 }
```

## Add User fail:

The screenshot shows a Postman request to `http://localhost:9090/signup` using a POST method. The request body is identical to the successful one above. The response status is 409 Conflict, and the response body indicates that a user with the email `ananya@example.com` already exists.

```
1 {
2   "name": "ananya",
3   "emailId": "ananya@example.com",
4   "mobileNo": "1234567890",
5   "gender": "Female",
6   "age": 21,
7   "password": "SecurePass123!"
8 }
```

## Add Flight:

The screenshot shows a POST request to `http://localhost:9090/airline/route/add`. The request body is a JSON object with the following fields:

```
1 {
2   "flightNumber": "AI102",
3   "sourceAirport": "BOM",
4   "destinationAirport": "DEL",
5   "departureTime": "08:30:00",
6   "arrivalTime": "10:45:00",
7   "duration": "PT2H15M"
8 }
```

The response status is **201 Created**, with a response time of 16 ms and a response size of 88 B. The response body contains the flight number `AI102`.

## Add flight fail:

The screenshot shows a POST request to `http://localhost:9090/airline/route/add`. The request body is identical to the successful one above. The response status is **409 Conflict**, with a response time of 31 ms and a response size of 119 B. The response body contains the message `Flight Number AI102 already exists`.

## Add schedule:

The screenshot shows the Postman interface with the following details:

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

```
1 {
2   "airlineName": "Air India Express",
3   "departureDate": "2025-11-27",
4   "basePrice": 4500.0,
5   "totalSeats": 180,
6   "availableSeats": 180,
7   "flightNumber": "AI102"
8 }
```
- Response Status:** 201 Created
- Response Time:** 7 ms
- Response Size:** 108 B
- Response Body:** 69246880c1155f4c5682eae9

## Add schedule fail:

The screenshot shows the Postman interface with the following details:

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

```
1 {
2   "airlineName": "Air India Express",
3   "departureDate": "2025-11-27",
4   "basePrice": 4500.0,
5   "totalSeats": 180,
6   "availableSeats": 180,
7   "flightNumber": "AI102"
8 }
```
- Response Status:** 409 Conflict
- Response Time:** 15 ms
- Response Size:** 149 B
- Response Body:** Schedule with Flight Number AI102 already scheduled on 2025-11-27

## Search flight:

The screenshot shows a Postman request for searching flights. The URL is `http://localhost:9090/flights/search`. The request method is POST. The body is a JSON object:

```
1 {
2     "sourceAirport": "BOM",
3     "destinationAirport": "DEL",
4     "roundTrip": false,
5     "departureDate": "2025-11-27",
6     "passengerCount": 2
7 }
```

The response status is 200 OK, with a response time of 5 ms and a response size of 395 B. The response body is:

```
1 {
2     "departureCount": 1,
3     "returnCount": 0,
4     "departure": [
5         {
6             "airlineName": "Air India Express",
7             "arrivalTime": "10:45:00",
8             "basePrice": 4500.0,
9         }
10    ]
11 }
```

## Booking:

The screenshot shows a Postman request for booking a flight. The URL is `http://localhost:9090/booking`. The request method is POST. The body is a JSON object:

```
1 {
2     "emailId": "user@gmail.com",
3     "departureScheduleId": "692466ecc1155f4c5682ea81",
4     //   "returnScheduleId" : "6923f9cb4a5569b042ffaa45",
5     "roundTrip": false,
6     "passengerCount": 2,
7     "passengers": [
8         {
9             "firstName": "ananya",
10            "lastName": "nayak",
11            "age": 21,
12            "departureSeatNumber": "2B",
13            //   "returnSeatNumber": "2B",
14            "gender": "FEMALE",
15            "otherDetails": "none"
16        }
17    ]
18 }
```

The response status is 201 Created, with a response time of 11 ms and a response size of 116 B. The response body is:

```
1 Booking Successful! PNR : XUg5FM
```

## Seat not available while booking:

The screenshot shows a POST request to `http://localhost:9090/booking`. The request body is a JSON object with fields: `firstName: "Riya"`, `lastName: "Rao"`, `age: 22`, `departureSeatNumber: "2D"`, `gender: "FEMALE"`, and `mealOption: "NONVEG"`. The response status is **400 Bad Request** with the message `Seat Number 2D not available on flight 692466ecc1155f4c5682ea81.`.

```
16  },
17  {
18    "firstName": "Riya",
19    "lastName": "Rao",
20    "age": 22,
21    "departureSeatNumber": "2D",
22    // "returnSeatNumber": "2D",
23    "gender": "FEMALE",
24    "mealOption": "NONVEG"
```

## Get tickets by pnr:

The screenshot shows a GET request to `http://localhost:9090/ticket/2ubYPL`. The response status is **200 OK** with a response body containing a single ticket record for a passenger named `ananya nayak` with seat number `2A`.

```
1  [
2    {
3      "firstName": "ananya",
4      "lastName": "nayak",
5      "age": 21,
6      "gender": "FEMALE",
7      "seatNumber": "2A",
8      "mealOption": "VEG",
9      "date": "2025-11-29",
10     "fromAirport": "BOM",
11     "toAirport": "DEL",
12     "fromTime": "08:30:00",
13     "toTime": "10:45:00",
14     "status": "CONFIRMED"
15   },
```

## Booking not found (pnr doesn't exist)

The screenshot shows the Postman interface with the following details:

- Request Method:** GET
- Request URL:** <http://localhost:9090/ticket/2uXYZZ>
- Body:** None
- Response Status:** 404 Not Found
- Response Body:** 1 Booking not found with 2uXYZZ

## Booking history by email:

The screenshot shows the Postman interface with the following details:

- Request Method:** GET
- Request URL:** <http://localhost:9090/booking/history/user@gmail.com>
- Body:** JSON
- Response Status:** 200 OK
- Response Body:** (JSON Data)

```
[{"id": 1, "user_email": "user@gmail.com", "first_name": "Riya", "last_name": "Rao", "age": 22, "gender": "FEMALE", "seat_number": "2D", "meal_option": "NONVEG", "date": "2025-11-29", "from_airport": "BOM", "to_airport": "DEL", "from_time": "08:30:00", "to_time": "10:45:00", "status": "CONFIRMED"}, {"id": 2, "user_email": "user@gmail.com", "first_name": "ananya", "last_name": "nayak", "age": 21, "gender": "FEMALE", "seat_number": "2D", "meal_option": "NONVEG", "date": "2025-11-29", "from_airport": "BOM", "to_airport": "DEL", "from_time": "08:30:00", "to_time": "10:45:00", "status": "CONFIRMED"}]
```

## Boooking not found using email:

The screenshot shows a Postman request for a GET operation at `http://localhost:9090/booking/history/user123@gmail.com`. The response status is 404 Not Found, with a message: "Booking not found with user123@gmail.com".

## Delete booking :

The screenshot shows a DELETE operation at `http://localhost:9090/booking/cancel/XUg5FM`. The response status is 200 OK, with a message: "1".