

**CHUBB®**

## **WEEK-7 ASSIGNMENT**

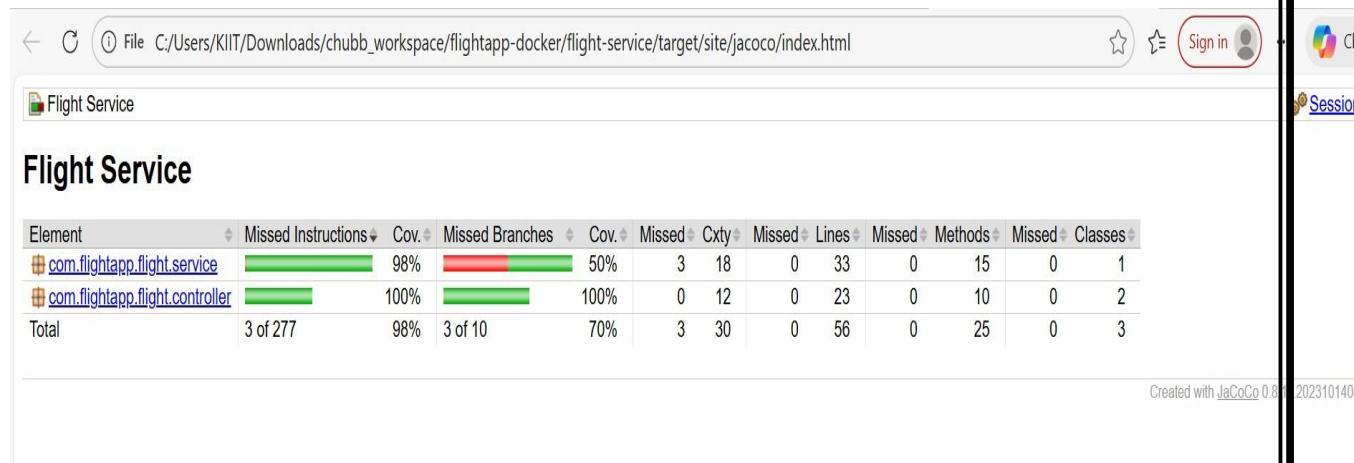
-Riddhima Bhanja  
Kalinga Institute of Industrial Technology  
Bhubaneswar(Java Track)

# INDEX

Content	Page No.
<b>1. JACOCO Code Coverage Report</b>	1
<b>2. SonarQube Report &amp; Issues</b>	3
<b>3. System Architecture, ER Diagram</b>	5
<b>4. JMeter &amp; RabbitMQ dashboard</b>	6
— JMeter Result Tree	6
— JMeter Summary Report	6
— Apache JMETER Dashboard	7
<b>5. Logs</b>	9
— RabbitMQ Dashboard	9
— Eureka server, Booking Service	10
— Flight Service, API Gateway, Notification service	10
<b>6. MongoDB Screenshots</b>	11
<b>7. Postman Screenshots</b>	11
— Circuit Breaker, Message broker	11
— — JWT security through API Gateway	12
— — Status,Events	12
— — Health Checks	13
— — API Gateway Health	13
— — Booking Service Health	13
— — Eureka Server Health	14
— — Flight Service Health	14
— Add Flight	15
— Book Flight, get flight by PNR, get flight by ID	15
— Cancel Booking, FALBACK CASE	15
— Get Booking by PNR	16
— Get Booking History	16
<b>8. Email</b>	17
— With PDF	17
— Without PDF	17
— Fallback Case	17
<b>9. Eureka Dashboard, MySQL Workbench</b>	17

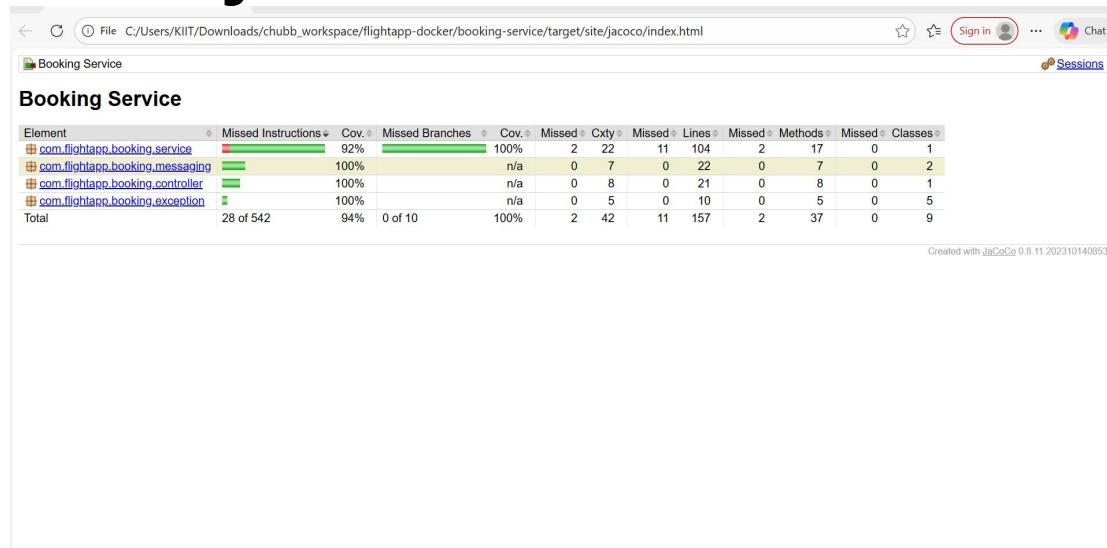
# JACOCO REPORTS

## FLIGHT SERVICE: 98% COVERAGE

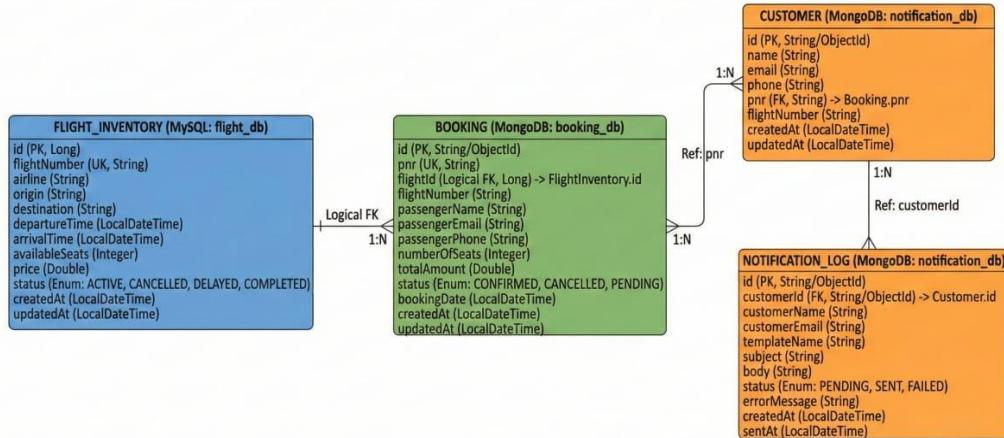


## BOOKING SERVICE

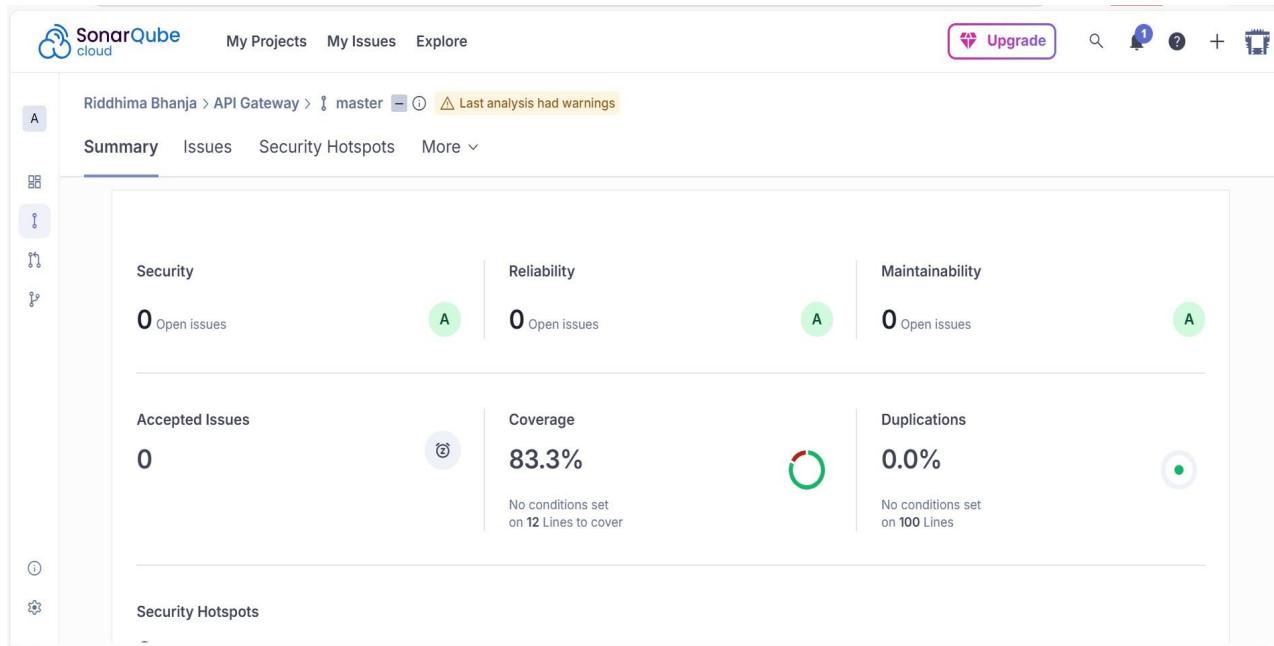
### 94% coverage



## ER DIAGRAM



## 1. SonarQUBE Code Coverage



## 2. SonarQube Issues

### Before fixing:

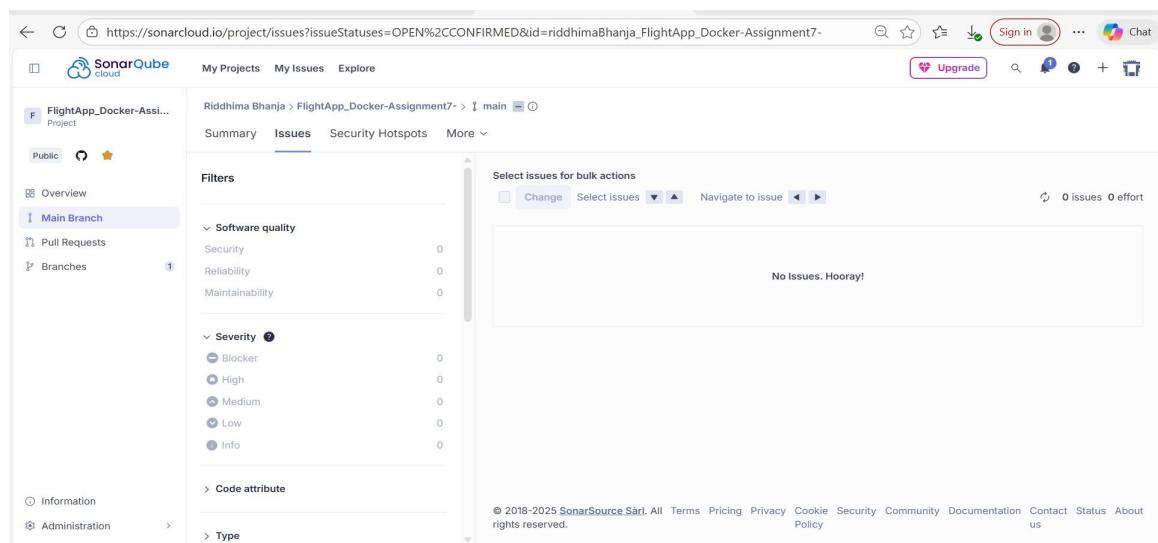
The screenshot shows the SonarQube Issues page for the project 'Riddhima Bhanja > docker > main'. The left sidebar includes sections for Overview, Main Branch (selected), Pull Requests, Branches, Information, and Administration. The main area displays a list of 32 issues under the 'Issues' tab. The issues are categorized by severity: Blocker (1), High (5), Medium (15), Low (11), and Info (0). The 'Software quality' filter is applied, showing 1 issue related to 'Remove this field injection and use constructor injection instead.' Other filters include 'Reliability' (4 issues) and 'Maintainability' (29 issues). The right side of the interface shows navigation controls for bulk actions and a summary of the total effort required.

The screenshot shows the SonarQube Overall Summary page for the project 'Riddhima Bhanja > docker > main'. The left sidebar includes sections for Overview, Main Branch (selected), Pull Requests, Branches, Information, and Administration. The main area displays a summary of project health across several metrics:

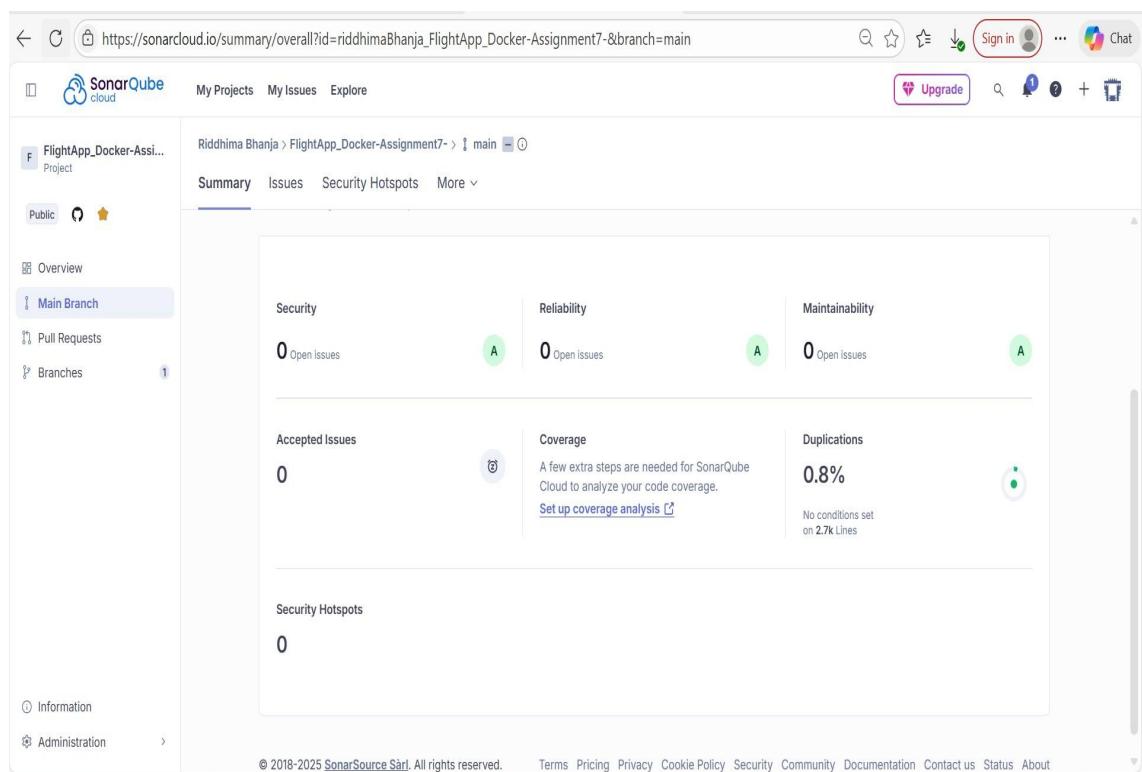
Category	Value	Color
Security	1 Open issues	E
Reliability	4 Open issues	C
Maintainability	29 Open issues	A
Accepted Issues	0	B
Coverage	A few extra steps are needed for SonarQube Cloud to analyze your code coverage. Set up coverage analysis	D
Duplications	0.7%	G
Security Hotspots	0	F

The summary also notes that no conditions have been set on 3k Lines.

## After fixing:

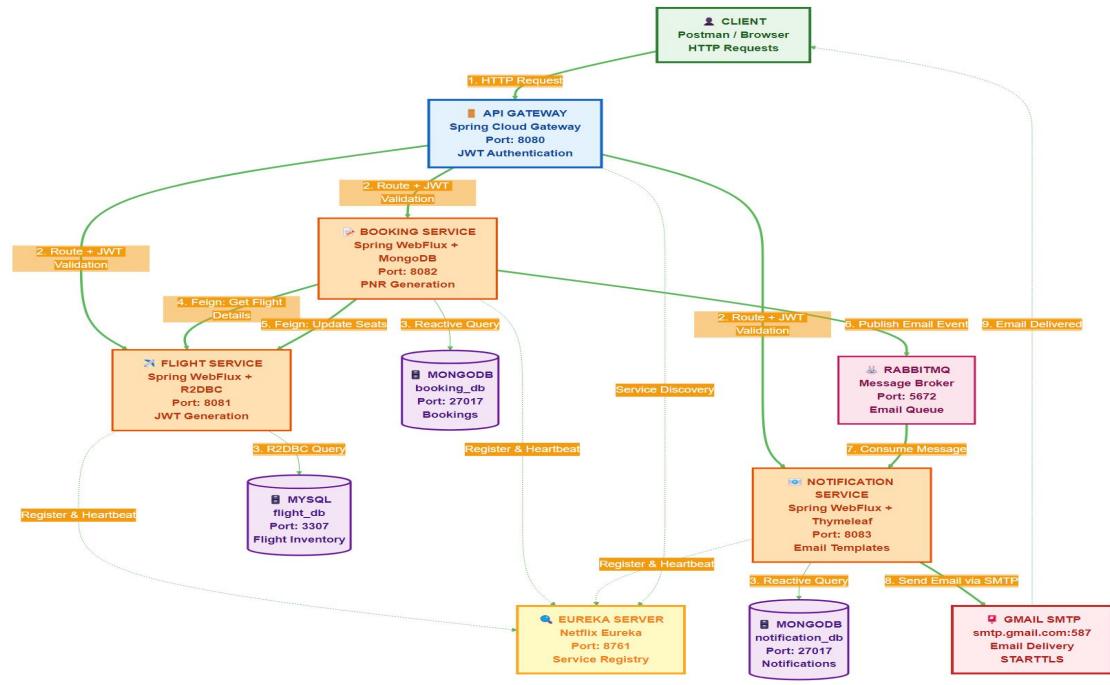


The screenshot shows the SonarCloud Issues page for the project 'FlightApp\_Docker-Assignment7'. The URL is [https://sonarcloud.io/project/issues?issueStatuses=OPEN%2CCONFIRMED&id=riddhimaBhanja\\_FlightApp\\_Docker-Assignment7-](https://sonarcloud.io/project/issues?issueStatuses=OPEN%2CCONFIRMED&id=riddhimaBhanja_FlightApp_Docker-Assignment7-). The page title is 'Riddhima Bhanja > FlightApp\_Docker-Assignment7 - main'. The left sidebar shows 'Main Branch' selected. The main content area displays a table with columns 'Select issues for bulk actions', 'Change', 'Select issues', and 'Navigate to issue'. A message at the bottom right says 'No Issues. Hooray!'. The footer includes links for 'Terms', 'Pricing', 'Privacy', 'Cookie Policy', 'Security', 'Community', 'Documentation', 'Contact us', 'Status', and 'About'.



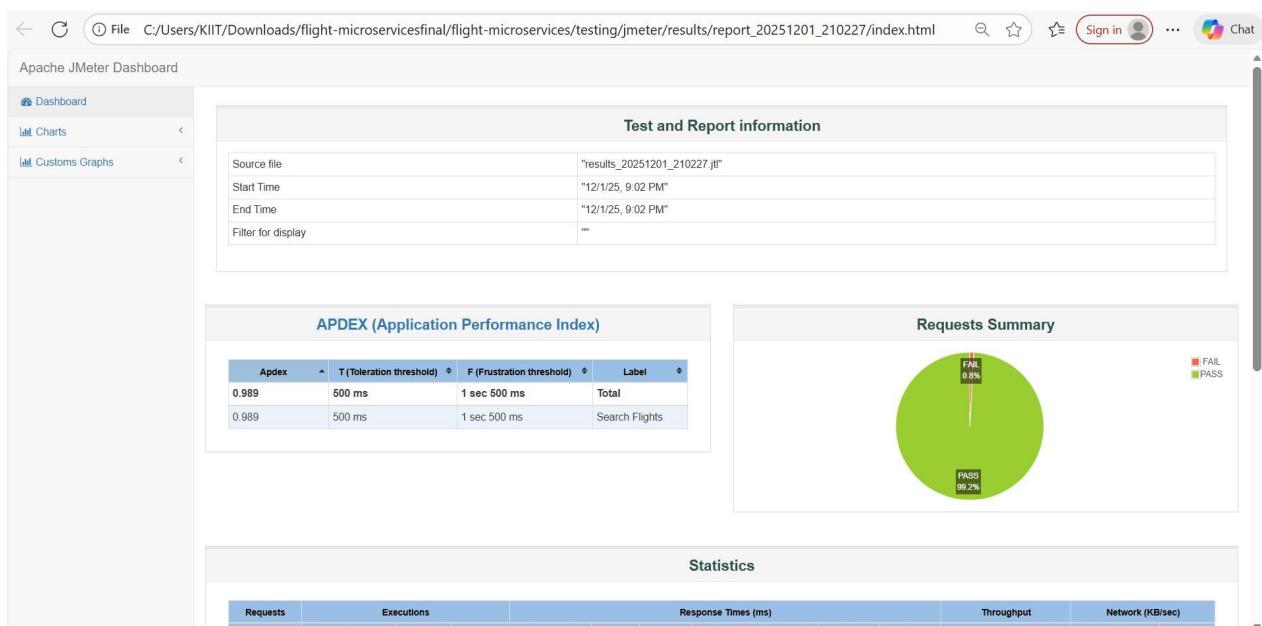
The screenshot shows the SonarCloud Overall Summary page for the project 'FlightApp\_Docker-Assignment7'. The URL is [https://sonarcloud.io/summary/overall?id=riddhimaBhanja\\_FlightApp\\_Docker-Assignment7-&branch=main](https://sonarcloud.io/summary/overall?id=riddhimaBhanja_FlightApp_Docker-Assignment7-&branch=main). The page title is 'Riddhima Bhanja > FlightApp\_Docker-Assignment7 - main'. The left sidebar shows 'Main Branch' selected. The main content area displays various metrics: Security (0 Open issues), Reliability (0 Open issues), Maintainability (0 Open issues), Accepted Issues (0), Coverage (0.8%), and Duplications (0.8%). A note says 'A few extra steps are needed for SonarQube Cloud to analyze your code coverage. Set up coverage analysis'. The footer includes links for 'Terms', 'Pricing', 'Privacy', 'Cookie Policy', 'Security', 'Community', 'Documentation', 'Contact us', 'Status', and 'About'.

### **3. System Architecture**



### **4. Jmeter**

- Apache Jmeter Dashboard**



**Statistics**

Label	#Samples	Requests			Executions						Response Times (ms)						Throughput		Network (KB/sec)	
		#FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent							
Total	500	4	0.80%	46.39	9	1264	16.00	88.90	127.85	989.02	50.97	22.16	13.59							
Search Flights	500	4	0.80%	46.39	9	1264	16.00	88.90	127.85	989.02	50.97	22.16	13.59							

**Errors**

Type of error	Number of errors	% in errors	% in all samples
405/Method Not Allowed	4	100.00%	0.80%

**Top 5 Errors by sampler**

Sample	#Samples	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error
Total	500	4	405/Method Not Allowed	4									
Search Flights	500	4	405/Method Not Allowed	4									

## LOGS SCREENSHOTS

## DASHBOARD

**docker desktop PERSONAL**

- Ask Gordon BETA
- Containers
- Images
- Volumes
- Kubernetes
- Builds
- Models
- MCP Toolkit BETA
- Docker Hub
- Docker Scout
- Extensions

**flightapp-docker** C:\Users\KLT\Downloads\chubb\_workspace\flightapp-docker

View configurations

Search

Ctrl+K

MySQL

rabbitmq

mongodb

mailhog

flight

booking

api-gate...

eureka...

notificat...

mysql:8.0

rabbitmq:3.1

mongo:7.0

mailhog/ma...

flightapp-dor...

flightapp-dor...

flightapp-dor...

flightapp-dor...

flightapp-dor...

flightapp-dor...

flightapp-dor...

flightapp-dor...

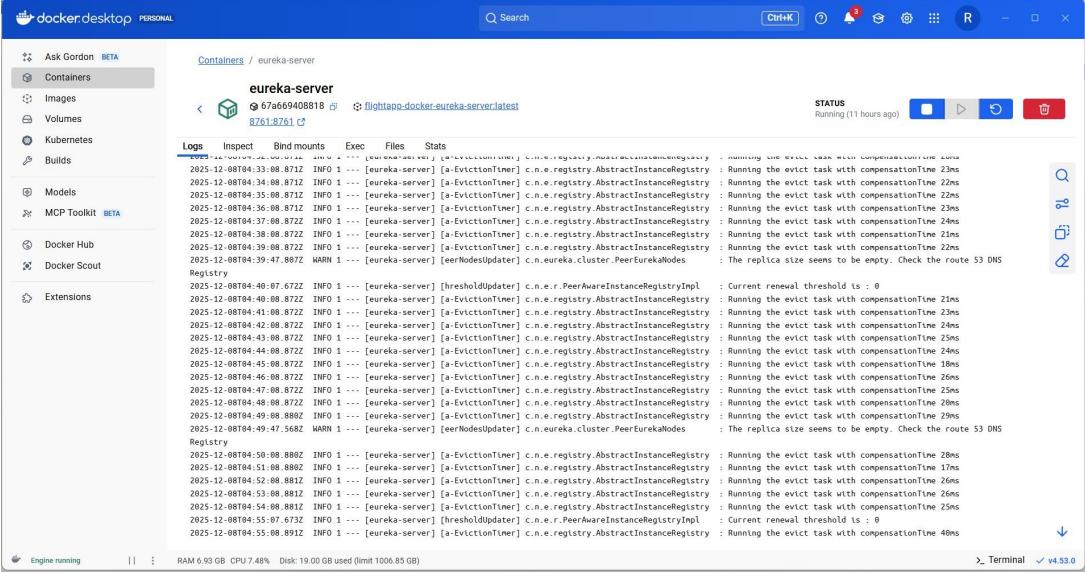
conn35080", "msg": "client metadata", "attr": {"remote": "127.0.0.1:43872", "client": "conn35080", "negotiatedCompressors": [], "doc": {"application": {"name": "mongosh 2.5.9"}, "driver": {"name": "nodejs|mongosh", "version": "6.19.0|2.5.9"}, "platform": "Node.js v20.19.5, LE", "os": {"name": "linux", "architecture": "x64", "version": "3.19.8-327.22.2.el7.x86\_64"}, "type": "Linux"}, "env": {"container": [{"runTime": "docker"}]}}, {"t": {"\$date": "2025-12-08T04:51:51.461+00:00"}, "s": "I", "c": "ACCESS", "id": 10483960, "ctx": "conn35080", "msg": "Connection not authenticating", "attr": {"client": "127.0.0.1:43872", "doc": {"application": {"name": "mongosh 2.5.9"}, "driver": {"name": "nodejs|mongosh", "version": "6.19.0|2.5.9"}, "platform": "Node.js v20.19.5, LE", "os": {"name": "linux", "architecture": "x64", "version": "3.19.8-327.22.2.el7.x86\_64"}, "type": "Linux"}, "env": {"container": [{"runTime": "docker"}]}}, {"t": {"\$date": "2025-12-08T04:51:51.461+00:00"}, "s": "I", "c": "NETWORK", "id": 6788700, "ctx": "conn35080", "msg": "Received first command on ingress connection since session start or auth handshake", "attr": {"elapsdMillis": 2}}, {"t": {"\$date": "2025-12-08T04:51:51.487+00:00"}, "s": "I", "c": "NETWORK", "id": 22944, "ctx": "conn35080", "msg": "Connection ended", "attr": {"remote": "127.0.0.1:43872", "isLoadBalanced": false, "uid": {"\$uuid": "4a6ff64d-ec02-4372-a3d3-13a0121bd72b"}, "connectionId": 35080, "connectionCount": 10}}, {"t": {"\$date": "2025-12-08T04:51:51.487+00:00"}, "s": "I", "c": "NETWORK", "id": 22944, "ctx": "conn35087", "msg": "Connection ended", "attr": {"remote": "127.0.0.1:43844", "isLoadBalanced": false, "uid": {"\$uuid": "50324352-493c-42f2-bdc1-a121cb7cf4d7"}, "connectionId": 35087, "connectionCount": 9}}, {"t": {"\$date": "2025-12-08T04:51:51.487+00:00"}, "s": "I", "c": "NETWORK", "id": 22944, "ctx": "conn35079", "msg": "Connection ended", "attr": {"remote": "127.0.0.1:43838", "isLoadBalanced": false, "uid": {"\$uuid": "a2f299f8-d489-4b2e-86e6-7a10422ba93c"}, "connectionId": 35087, "connectionCount": 7}}, {"t": {"\$date": "2025-12-08T04:51:51.487+00:00"}, "s": "I", "c": "NETWORK", "id": 22944, "ctx": "conn35078", "msg": "Connection ended", "attr": {"remote": "127.0.0.1:43848", "isLoadBalanced": false, "uid": {"\$uuid": "779b3f9f-6d10-4bc9-9ed1-8bb7a7c365a9"}, "connectionId": 35078, "connectionCount": 6}}

Engine running

RAM: 7.21 GB CPU: 5.84% Disk: 19.00 GB used (limit 1006.85 GB)

Terminal v4.53.0

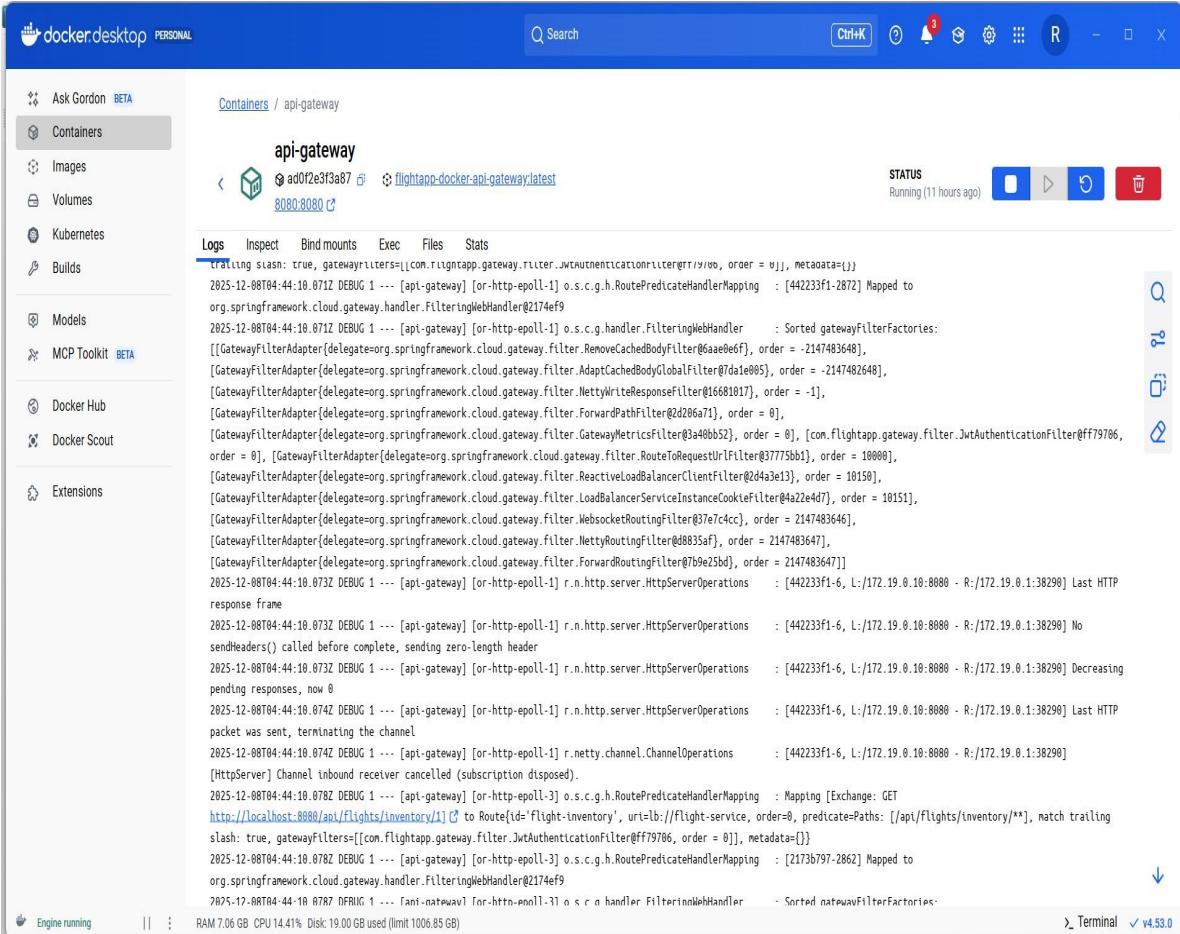
## EUREKA SERVER LOGS



Docker Desktop interface showing the logs for the eureka-server container. The logs output information about the eviction task running every 20ms, with compensationTime set to 20ms. It also shows periodic tasks for threshold updating and replica size checks.

```
2025-12-08T04:33:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:34:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:35:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:36:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:37:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:38:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:39:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 20ms
2025-12-08T04:39:47.897Z MARN 1 --- [eureka-server] [eerNodesUpdater] c.n.eureka.cluster.PeerEurekaNodes : The replica size seems to be empty. Check the route 53 DNS Registry
2025-12-08T04:40:47.568Z INFO 1 --- [eureka-server] [thresholdUpdater] c.n.e.PeerAwareInstanceRegistryImpl : Current renewal threshold is : 8
2025-12-08T04:40:48.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 21ms
2025-12-08T04:41:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 23ms
2025-12-08T04:42:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 24ms
2025-12-08T04:43:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 25ms
2025-12-08T04:44:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 24ms
2025-12-08T04:45:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 18ms
2025-12-08T04:46:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:47:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 25ms
2025-12-08T04:48:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:49:08.888Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:49:47.568Z MARN 1 --- [eureka-server] [eerNodesUpdater] c.n.eureka.cluster.PeerEurekaNodes : The replica size seems to be empty. Check the route 53 DNS Registry
2025-12-08T04:50:08.888Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 28ms
2025-12-08T04:51:08.888Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 17ms
2025-12-08T04:52:08.881Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:53:08.881Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:54:08.881Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 25ms
2025-12-08T04:55:07.673Z INFO 1 --- [eureka-server] [thresholdUpdater] c.n.e.PeerAwareInstanceRegistryImpl : Current renewal threshold is : 8
2025-12-08T04:55:08.891Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 40ms
```

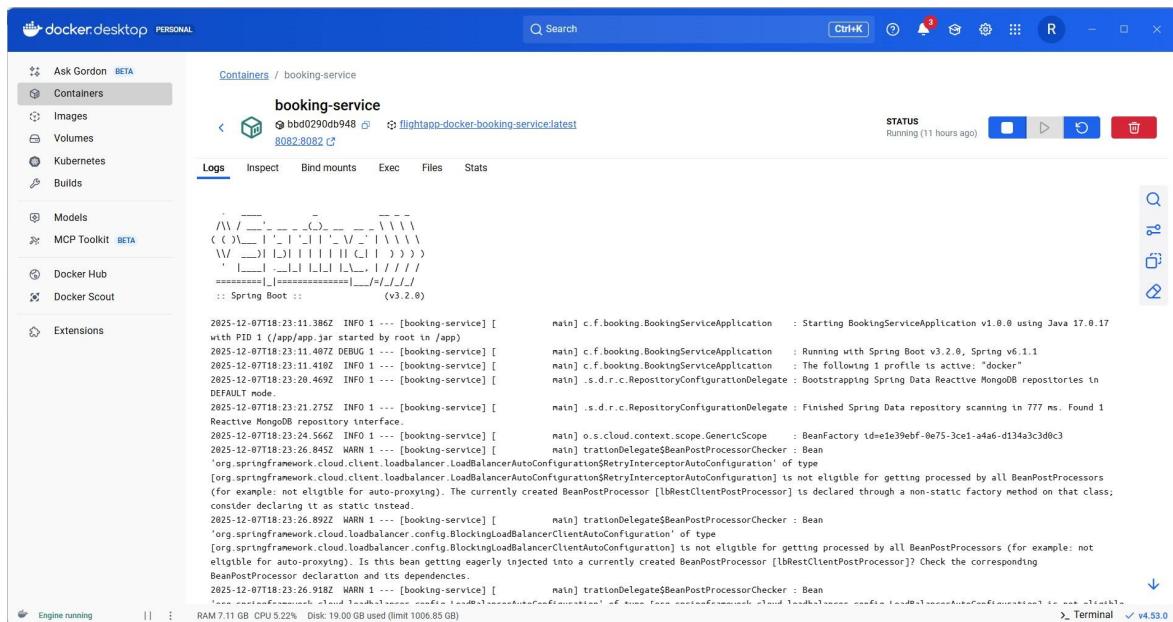
## API GATEWAY



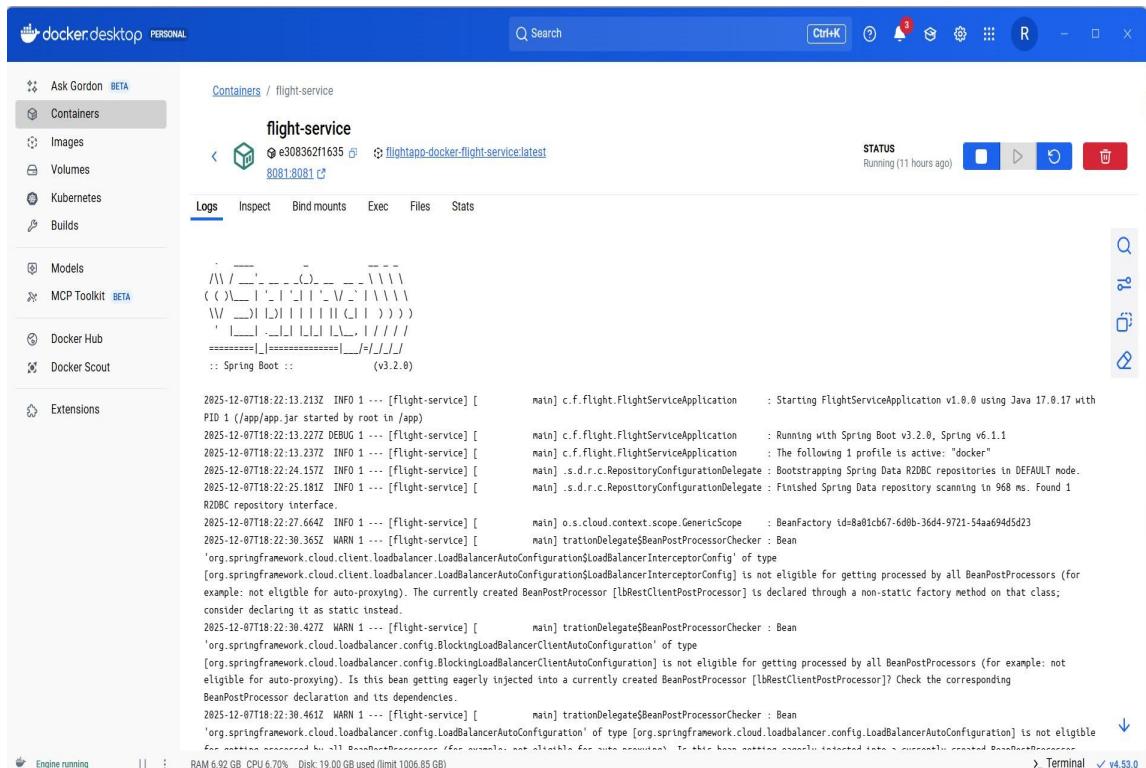
Docker Desktop interface showing the logs for the api-gateway container. The logs detail the configuration of gateway filters, including JwtAuthenticationFilter, NettyWriteResponseFilter, and various metrics and route filters. It also shows the handling of HTTP requests and responses.

```
2025-12-08T04:44:10.071Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] o.s.c.g.h.RoutePredicateHandlerMapping : [442233f1-2872] Mapped to org.springframework.cloud.gateway.handler.FilteringWebHandler@2174ef9
2025-12-08T04:44:10.071Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] o.s.c.g.h.RoutePredicateHandlerMapping : Sorted gatewayFilterFactories: [[GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.Filter.AdaptCachedBodyGlobalFilter@7d1e005], order = -2147482648], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.AdaptCachedBodyGlobalFilter@16681017], order = -1], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.NettyWriteResponseFilter@16681017], order = -1], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.ForwardPathFilter@2d206a71], order = 0], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.GatewayMetricsFilter@34a0bb52], order = 0], [com.flighapp.gateway.filter.JwtAuthenticationFilter@ff79786], order = 0], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.RouteToRequestUrlFilter@37775bb1], order = 100000], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.ReactiveLoadBalancerClientFilter@2d4a3e13], order = 10150], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.LoadBalancerServiceInstanceCookieFilter@422e4d0], order = 10151], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.WebsocketRoutingFilter@37e7c4cc], order = 2147483646], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.NettyRoutingFilter@08835af], order = 2147483647], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.ForwardRoutingFilter@709e25bd], order = 2147483647]]
2025-12-08T04:44:10.073Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] r.n.http.server.HttpServerOperations : [442233f1-6, L:/172.19.0.10:8080 - R:/172.19.0.1:38298] Last HTTP response frame
2025-12-08T04:44:10.073Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] r.n.http.server.HttpServerOperations : [442233f1-6, L:/172.19.0.10:8080 - R:/172.19.0.1:38298] No sendHeaders() called before complete, sending zero-length header
2025-12-08T04:44:10.073Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] r.n.http.server.HttpServerOperations : [442233f1-6, L:/172.19.0.10:8080 - R:/172.19.0.1:38298] Decreasing pending responses, now 0
2025-12-08T04:44:10.074Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] r.n.http.server.HttpServerOperations : [442233f1-6, L:/172.19.0.10:8080 - R:/172.19.0.1:38298] Last HTTP packet was sent, terminating the channel
2025-12-08T04:44:10.074Z DEBUG 1 --- [apt-gateway] [or-http-epoll-1] r.netty.channel.ChannelOperations : [442233f1-6, L:/172.19.0.10:8080 - R:/172.19.0.1:38298] [HttpServer] Channel inbound receiver cancelled (subscription disposed).
2025-12-08T04:44:10.078Z DEBUG 1 --- [apt-gateway] [or-http-epoll-3] o.s.c.g.h.RoutePredicateHandlerMapping : Mapping [Exchange: GET http://localhost:8080/api/flights/inventory/] To Route{id='flight-inventory', uri=/b//flight-service, order=0, predicatePaths: [/api/flights/inventory/**], match trailing slash=true, gatewayFilters=[com.flighapp.gateway.filter.JwtAuthenticationFilter@ff79786, order = 0]}, metadata={}
2025-12-08T04:44:10.078Z DEBUG 1 --- [apt-gateway] [or-http-epoll-3] o.s.c.g.h.RoutePredicateHandlerMapping : [2173b797-2862] Mapped to org.springframework.cloud.gateway.handler.FilteringWebHandler@2174ef9
2025-12-08T04:44:10.078Z DEBUG 1 --- [apt-gateway] [or-http-epoll-3] o.s.c.g.h.RoutePredicateHandlerMapping : Sorted gatewayFilterFactories:
```

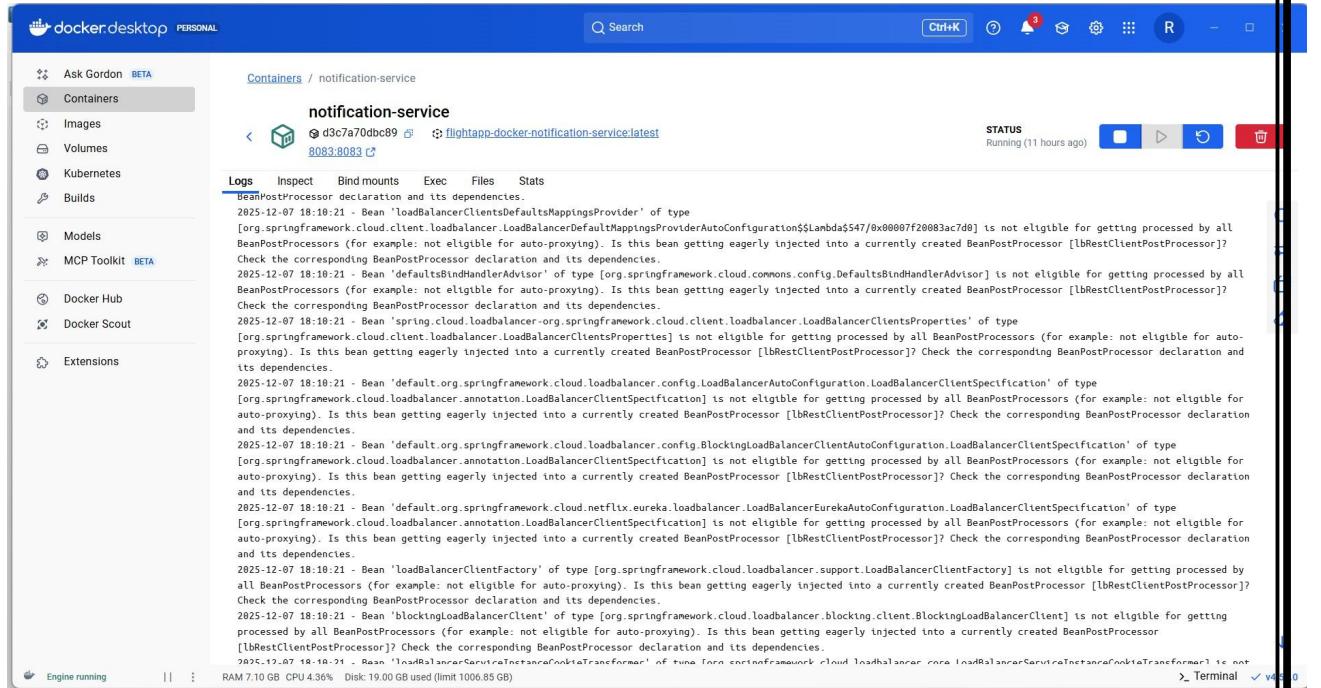
## **BOOKING SERVICE LOGS**



# FLIGHT SERVICE LOGS



## NOTIFICATION SERVICE LOGS



Docker Desktop PERSONAL

Containers / notification-service

**notification-service**

Status: Running (11 hours ago)

Logs Inspect Bind mounts Exec Files Stats

```
2025-12-07 18:19:21 - Bean 'loadBalancerClientsDefaultsMappingsProvider' of type [org.springframework.cloud.client.loadbalancer.LoadBalancerDefaultMappingsProviderAutoConfiguration$$Lambda$547/0x00007f20083ac7d0] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'defaultBindHandlerAdvisor' of type [org.springframework.cloud.commons.config.DefaultsBindHandlerAdvisor] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'spring.cloud.loadbalancer.org.springframework.cloud.client.loadbalancer.LoadBalancerClientsProperties' of type [org.springframework.cloud.client.loadbalancer.LoadBalancerClientsProperties] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'default.org.springframework.cloud.loadbalancer.config.LoadBalancerAutoConfiguration.LoadBalancerClientSpecification' of type [org.springframework.cloud.loadbalancer.annotation.LoadBalancerClientSpecification] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'default.org.springframework.cloud.loadbalancer.config.BlockingLoadBalancerClientAutoConfiguration.LoadBalancerClientSpecification' of type [org.springframework.cloud.loadbalancer.annotation.LoadBalancerClientSpecification] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'default.org.springframework.cloud.netflix.eureka.loadbalancer.LoadBalancerEurekaAutoConfiguration.LoadBalancerClientSpecification' of type [org.springframework.cloud.loadbalancer.annotation.LoadBalancerClientSpecification] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

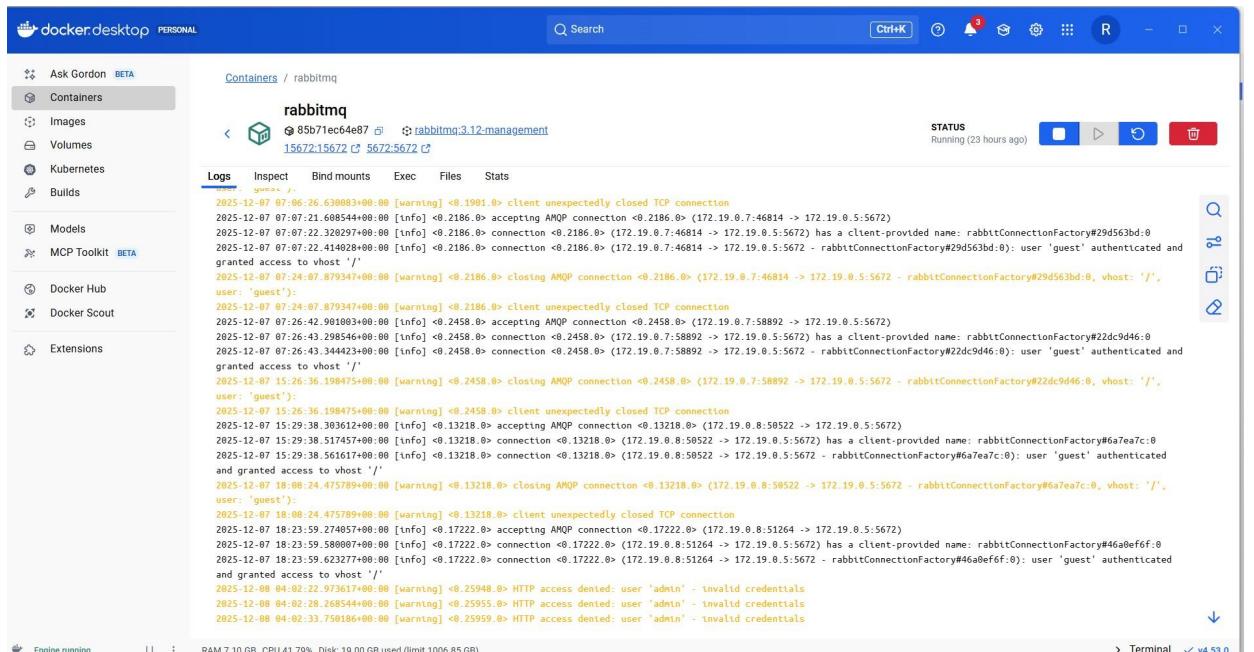
```
2025-12-07 18:19:21 - Bean 'loadBalancerClientFactory' of type [org.springframework.cloud.loadbalancer.support.LoadBalancerClientFactory] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'blockingLoadBalancerClient' of type [org.springframework.cloud.loadbalancer.blocking.client.BlockingLoadBalancerClient] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

```
2025-12-07 18:19:21 - Bean 'loadBalancerServiceTransformer' of type [org.springframework.cloud.loadbalancer.core.loadBalancerServiceTransformer] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently created BeanPostProcessor [lbRestClientPostProcessor]? Check the corresponding BeanPostProcessor declaration and its dependencies.
```

Logs Engine running RAM 7.10 GB CPU 4.36% Disk: 19.00 GB used (limit 1006.85 GB) Terminal v4.53.0

## RabbitMQ logs



Docker Desktop PERSONAL

Containers / rabbitmq

**rabbitmq**

Status: Running (23 hours ago)

Logs Inspect Bind mounts Exec Files Stats

```
2025-12-07 07:07:06.630083+00:00 [warning] <0.1901.0> client unexpectedly closed TCP connection
```

```
2025-12-07 07:21.608544+00:00 [info] <0.2186.0> accepting AMQP connection <0.2186.0> (172.19.0.7:46814 -> 172.19.0.5:5672)
```

```
2025-12-07 07:22.320297+00:00 [info] <0.2186.0> connection <0.2186.0> (172.19.0.7:46814 -> 172.19.0.5:5672) has a client-provided name: rabbitConnectionFactory#29d563bd:0 user: 'guest' authenticated and granted access to vhost '/'
```

```
2025-12-07 07:24:07.879347+00:00 [warning] <0.2186.0> closing AMQP connection <0.2186.0> (172.19.0.7:46814 -> 172.19.0.5:5672 - rabbitConnectionFactory#29d563bd:0, vhost: '/', user: 'guest')
```

```
2025-12-07 07:24:07.879347+00:00 [warning] <0.2186.0> client unexpectedly closed TCP connection
```

```
2025-12-07 07:26:42.981083+00:00 [info] <0.2458.0> accepting AMQP connection <0.2458.0> (172.19.0.7:58892 -> 172.19.0.5:5672)
```

```
2025-12-07 07:26:43.298546+00:00 [info] <0.2458.0> connection <0.2458.0> (172.19.0.7:58892 -> 172.19.0.5:5672 - rabbitConnectionFactory#22dc9d46:0) user: 'guest' authenticated and granted access to vhost '/'
```

```
2025-12-07 07:26:43.344423+00:00 [info] <0.2458.0> connection <0.2458.0> (172.19.0.7:58892 -> 172.19.0.5:5672 - rabbitConnectionFactory#22dc9d46:0) user: 'guest' authenticated and granted access to vhost '/'
```

```
2025-12-07 07:26:36.198475+00:00 [warning] <0.2458.0> closing AMQP connection <0.2458.0> (172.19.0.7:58892 -> 172.19.0.5:5672 - rabbitConnectionFactory#22dc9d46:0, vhost: '/', user: 'guest')
```

```
2025-12-07 07:26:36.198475+00:00 [warning] <0.2458.0> client unexpectedly closed TCP connection
```

```
2025-12-07 07:29:38.303612+00:00 [info] <0.13218.0> accepting AMQP connection <0.13218.0> (172.19.0.8:50522 -> 172.19.0.5:5672)
```

```
2025-12-07 07:29:38.517457+00:00 [info] <0.13218.0> connection <0.13218.0> (172.19.0.8:50522 -> 172.19.0.5:5672) has a client-provided name: rabbitConnectionFactory#6a7ea7c:0
```

```
2025-12-07 07:29:38.561617+00:00 [info] <0.13218.0> connection <0.13218.0> (172.19.0.8:50522 -> 172.19.0.5:5672 - rabbitConnectionFactory#6a7ea7c:0) user: 'guest' authenticated and granted access to vhost '/'
```

```
2025-12-07 08:24.475789+00:00 [warning] <0.13218.0> closing AMQP connection <0.13218.0> (172.19.0.8:50522 -> 172.19.0.5:5672 - rabbitConnectionFactory#6a7ea7c:0, vhost: '/', user: 'guest')
```

```
2025-12-07 08:24.475789+00:00 [warning] <0.13218.0> client unexpectedly closed TCP connection
```

```
2025-12-07 08:23:59.274857+00:00 [info] <0.17222.0> accepting AMQP connection <0.17222.0> (172.19.0.8:51264 -> 172.19.0.5:5672)
```

```
2025-12-07 08:23:59.580087+00:00 [info] <0.17222.0> connection <0.17222.0> (172.19.0.8:51264 -> 172.19.0.5:5672) has a client-provided name: rabbitConnectionFactory#46a0ef6f:0
```

```
2025-12-07 08:23:59.623277+00:00 [info] <0.17222.0> connection <0.17222.0> (172.19.0.8:51264 -> 172.19.0.5:5672 - rabbitConnectionFactory#46a0ef6f:0) user: 'guest' authenticated and granted access to vhost '/'
```

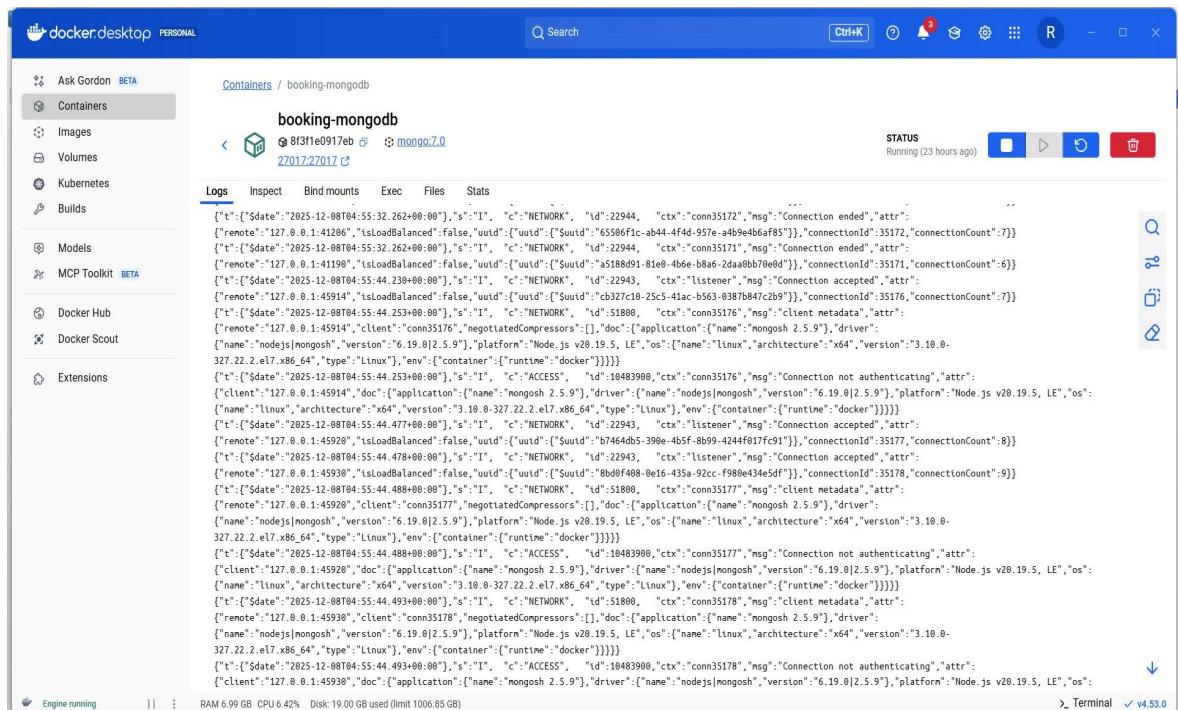
```
2025-12-08 04:02:22.973617+00:00 [warning] <0.25948.0> HTTP access denied: user 'admin' - invalid credentials
```

```
2025-12-08 04:02:28.268544+00:00 [warning] <0.25955.0> HTTP access denied: user 'admin' - invalid credentials
```

```
2025-12-08 04:02:33.758186+00:00 [warning] <0.25959.0> HTTP access denied: user 'admin' - invalid credentials
```

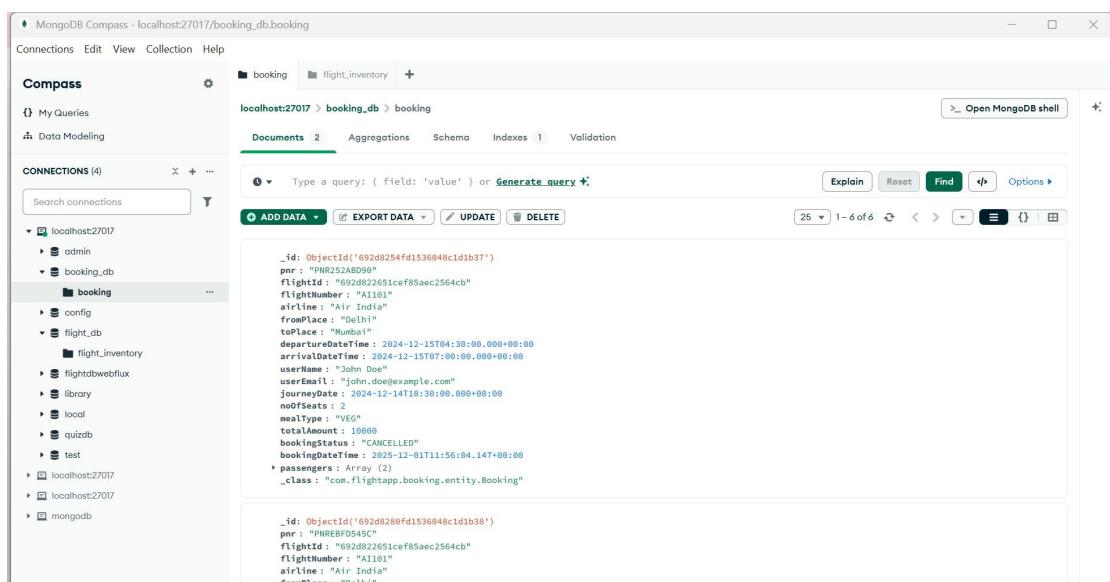
Logs Engine running RAM 7.10 GB CPU 41.79% Disk: 19.00 GB used (limit 1006.85 GB) Terminal v4.53.0

## Mongodb logs



## **5. Mongodb screenshots**

- Booking db



- Flight db

MongoDB Compass - localhost:27017/flight\_db/flight\_inventory

Connections Edit View Collection Help

**Compass**

My Queries Data Modeling

CONNECTIONS (4)

- localhost:27017
  - admin
  - booking\_db
    - booking
  - config
  - flight\_db
    - flight\_inventory
  - flightdbwebflux
  - library
  - local
  - quizdb
  - test
- localhost:27017
- localhost:27017
- mongodb

localhost:27017 > **flight\_db** > **flight\_inventory**

Documents 1 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#) +:

[ADD DATA](#) [EXPORT DATA](#) [UPDATE](#) [DELETE](#)

`_id: ObjectId('692d822651cef85aec2564cb')
airline: "Air India"
flightNumber: "AI101"
fromPlace: "Delhi"
toPlace: "Mumbai"
departureDateTime: 2024-12-15T04:30:00.000+00:00
arrivalDateTime: 2024-12-15T07:00:00.000+00:00
totalSeats: 180
availableSeats: 174
ticketPrice: 5000
flightStatus: "ACTIVE"
oneWayPrice: 5000
roundTripPrice: 9000
mealAvailable: true
_class: "com.flightapp.flight.entity.FlightInventory"`

25 1-1 of 1

- Notification logs

MongoDB Compass - localhost:27017/flightdbwebflux.notification\_log

Connections Edit View Collection Help

**Compass**

My Queries Data Modeling

CONNECTIONS (4)

- localhost:27017
  - admin
  - booking\_db
    - booking
  - config
  - flight\_db
    - flight\_inventory
  - flightdbwebflux
    - booking
    - flight\_inventory
    - notification\_log
  - library
  - local
  - quizdb
  - test
- localhost:27017
- localhost:27017
- mongodb

localhost:27017 > **flightdbwebflux** > **notification\_log**

Documents 3 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#) +:

[ADD DATA](#) [EXPORT DATA](#) [UPDATE](#) [DELETE](#)

`status: "SUCCESS"
details: "Email sent"
createdAt: 2025-11-29T19:16:22.209+00:00
_class: "com.flightapp.entity.NotificationLog"`

`_id: ObjectId('692d85338195fb6e6811e1466')
passengerId: "692d828c54e03f4e58599e436"
email: "ridhimbhanjaj2003@gmail.com"
subject: "Your e-ticket and booking details"
type: "TICKET"
status: "SUCCESS"
details: "Email sent"
createdAt: 2025-12-01T12:08:19.565+00:00
_class: "com.flightapp.entity.NotificationLog"`

`_id: ObjectId('693880f65a1c847745c44c2')
passengerId: "692d828c54e03f4e58599e436"
email: "ridhimbhanjaj2003@gmail.com"
subject: "Your e-ticket and booking details"
type: "TICKET"
status: "SUCCESS"
details: "Email sent"
createdAt: 2025-12-03T18:27:02.142+00:00
_class: "com.flightapp.entity.NotificationLog"`

25 1-3 of 3

## 5. Email

- Welcome Email

The screenshot shows a Gmail inbox with a single email from 'riddhimabhanja2003@gmail.com'. The subject is 'Welcome Test'. The email content is as follows:

**Welcome Aboard!**

Dear Test User,

Welcome to our Flight Booking Service! We're excited to have you with us.

Your Customer ID: CUST789

With our service, you can:

- Search and book flights to destinations worldwide
- Manage your bookings easily
- Receive instant email confirmations
- Access 24/7 customer support

Start exploring our flight options and book your next journey today!

Best regards,  
Flight Booking Team

© 2025 Flight Booking Service. All rights reserved.

- Confirm Booking Email

The screenshot shows a Gmail inbox with a single email from 'riddhimabhanja2003@gmail.com'. The subject is 'Flight Booking Confirmation - PNR12345678'. The email content is as follows:

**Flight Booking Confirmed!**

Dear riddhima bhanja,

Your flight booking has been successfully confirmed. Below are your booking details:

**Booking Information**

PNR:PNR12345678  
Flight Number:AI101  
Number of Seats:2  
Total Amount:Rs. 10000  
Booking Date:2025-12-07

**Important Notes:**

- Please arrive at the airport at least 2 hours before departure
- Carry a valid photo ID for check-in
- Check-in opens 2 hours before departure

Thank you for choosing our service!

Best regards,  
Flight Booking Team

- Flight cancel email

Flight Booking Cancelled

riddhimabhanja2003@gmail.com  
to john.doe ▾  
Dear John Doe,  
  
Your flight booking has been cancelled.  
  
Booking Details:  
PNR: PNR45910D76  
Flight: AI101  
Route: Delhi to Mumbai  
Departure: 2024-12-15T10:00  
Total Amount: \$10000.0  
  
Thank you for choosing our service!  
  
Best regards,  
Flight Booking Team

## RabbitMQ email-queue dashboard

localhost:15672/#/queues/%2F/email.queue

RabbitMQ 3.12.14 Erlang 25.3.2.15

Overview Connections Channels Exchanges Queues and streams Admin

### Queue email.queue

Overview

Queued messages (last minute) ?

Ready: 0 Unacked: 0 Total: 0

Message rates (last minute) ?

Publish: 0.00/s Consumer: 0.00/s Get (acks): 0.00/s  
Deliver (memory ACK): 0.00/s Redelivered: 0.00/s Get (empty): 0.00/s  
Deliver (trans ACK): 0.00/s Get (manual ACK): 0.00/s

Details

Features	durable	auto-delete	State	Consumers	Messages	Total	Ready	Unacked	In memory	Persistent	Transient	Paged Out
Policy					Message body bytes	0	0	0	0	0	0	0
Operator policy					Process memory	0	0	0	0	0	0	0
Effective policy definition						10 KB						

Consumers (1)

Bindings (2)

Publish message

Get messages

Move messages

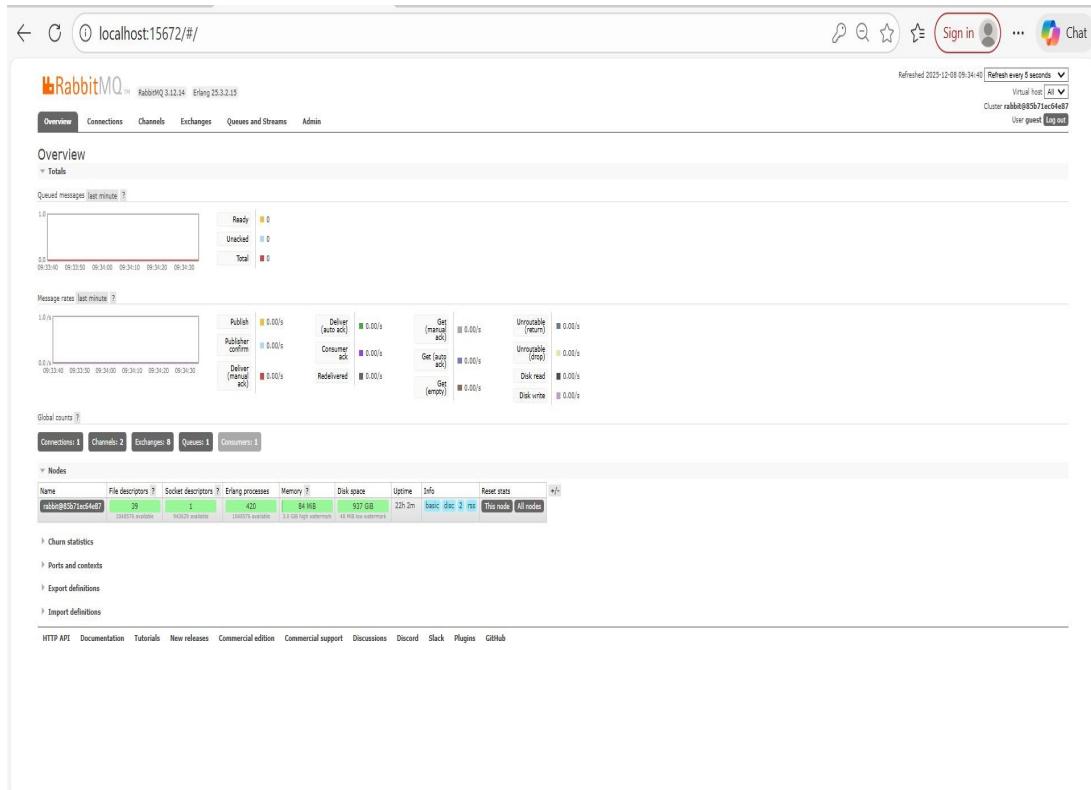
Delete

Purge

Runtime Metrics (Advanced)

HTTP API Documentation Tutorials New releases Commercial edition Commercial support Discussions Discord Slack Plugins GitHub

## RabbitMQ overview dashboard



## 6. Postman screenshots

### Obtain JWT TOKEN

The screenshot shows the Postman application interface. On the left, the sidebar lists workspaces, collections, environments, flows, and files. The current workspace is "riddhima bhanja's Workspace".

The main panel shows a collection named "flightService API(docker)". A specific POST request titled "obtain JWT token" is selected. The request URL is `http://localhost:8080/api/auth/login`. The request body is set to "raw" JSON:

```
1 {
2   "username": "admin",
3   "password": "password"
4 }
```

The response status is 200 OK, with a response time of 148 ms and a response size of 395 B. The response body is displayed as:

```
1 "token": "eyJhbGciOiJIUzIwMjQ9.eyJzdWIiOiJhZG1pbkiIsImhdCI6MTc2NTEzNDM5MiwiZXhwIjoxNjY1MjIwNzkyfQ.
2 8FY9CwGdLFuUh0Qo0zfZvk8LuHsCdUhvJhm1k3FRVxrtor66TBWEAIM4nWso4t;01vDzhKnVTQFUfTbk1_87w",
3 "username": "admin",
4 "message": "Login successful."
```

## Unauthorized access

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** flightService API(docker)
- Request:** GET /api/flights/search
- Body:** Raw JSON (copied below)
- Response Status:** 401 Unauthorized
- Response Body:** An empty JSON object {}

```
1 {
2   "origin": "DEL",
3   "destination": "BOM",
4   "travelDate": "2025-12-15"
5 }
```

## Same endpoint with auth bearer token

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** flightService API(docker)
- Request:** POST /api/flights/search
- Authorization:** Bearer Token (with a redacted token value)
- Response Status:** 200 OK
- Response Body:** A JSON object containing flight details.

```
3 {
4   "id": 1,
5   "flightNumber": "AI101",
6   "airline": "Air India",
7   "origin": "DEL",
8   "destination": "BOM",
9   "departureTime": "2025-12-15T08:00:00",
10  "arrivalTime": "2025-12-15T10:30:00",
11  "availableSeats": 134,
12  "price": 5000.0,
13  "status": "ACTIVE",
14  "createdAt": "2025-12-07T06:02:42",
```

- Circuit Breaker

## Events:

The screenshot shows the Postman interface with a collection named "riddhima bhanja's Workspace". The left sidebar lists various API endpoints under "circuit breaker". The main panel shows a GET request to `http://localhost:8080/actuator/circuitbreakerevents`. The response status is 200 OK, and the response body is a JSON object containing two entries for circuit breaker events. The first entry is for the "bookingServiceCircuitBreaker" and the second for the "flightServiceCircuitBreaker". Both entries include details like creation time, error message, duration in ms, and state transition.

```
1  {
2    "circuitBreakerEvents": [
3      {
4        "circuitBreakerName": "bookingServiceCircuitBreaker",
5        "type": "ERROR",
6        "creationTime": "2025-12-01T21:55:37.209614500+05:30[Asia/Calcutta]",
7        "errorMessage": "java.util.concurrent.TimeoutException: Did not observe any item or terminal signal within 1000ms in 'circuitBreaker' (and no fallback has been configured)",
8        "durationInMs": 1000,
9        "stateTransition": null
10      },
11      {
12        "circuitBreakerName": "flightServiceCircuitBreaker",
13        "type": "ERROR",
14        "creationTime": "2025-12-01T21:55:45.179704400+05:30[Asia/Calcutta]",
15        "errorMessage": "java.util.concurrent.TimeoutException: Did not observe any item or terminal signal within 1000ms in 'circuitBreaker' (and no fallback has been configured)"
16      }
17    ]
18  }
```

## Status:

The screenshot shows the Postman interface with the same collection "riddhima bhanja's Workspace". The left sidebar shows the "status" endpoint under "circuit breaker" is selected. The main panel shows a GET request to `http://localhost:8080/actuator/circuitbreakers`. The response status is 200 OK, and the response body is a JSON object containing information about the "bookingServiceCircuitBreaker" and "flightServiceCircuitBreaker". The "bookingServiceCircuitBreaker" is in a "CLOSED" state with failure rate and slow call rate thresholds. The "flightServiceCircuitBreaker" is also in a "CLOSED" state with its own specific metrics.

```
1  {
2    "circuitBreakers": [
3      {
4        "bookingServiceCircuitBreaker": {
5          "failureRate": "+1.0%",
6          "slowCallRate": "+1.0%",
7          "failureRateThreshold": "50.0%",
8          "slowCallRateThreshold": "100.0%",
9          "bufferedCalls": 3,
10         "failedCalls": 1,
11         "slowCalls": 0,
12         "slowFailedCalls": 0,
13         "notPermittedCalls": 0,
14         "state": "CLOSED"
15       },
16      {
17        "flightServiceCircuitBreaker": {
18        }
19      }
20    ]
21  }
```

- Health Check

### API gateway health:

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** Flight Booking Microservices
- Request:** GET /actuator/health
- Response Status:** 200 OK
- Response Body (JSON):**

```
{
  "status": "UP",
  "components": {
    "discoveryComposite": {
      "status": "UP",
      "components": {
        "discoveryClient": {
          "status": "UP",
          "details": {
            "services": [
              "api-gateway",
              "flight-service",
              "booking-service"
            ]
          }
        }
      }
    }
  }
}
```

### Booking Service Health:

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** Flight Booking Microservices
- Request:** GET /actuator/health
- Response Status:** 200 OK
- Response Body (JSON):**

```
{
  "status": "UP",
  "components": {
    "circuitBreakers": {
      "status": "UP",
      "details": {
        "flightService": {
          "status": "UP",
          "details": {
            "failureRate": "-1.0%",
            "failureRateThreshold": "50.0%",
            "slowCallRate": "-1.0%",
            "slowCallRateThreshold": "100.0%",
            "bufferedCalls": 0,
            "slowCalls": 0
          }
        }
      }
    }
  }
}
```

## Eureka server:

The screenshot shows the Postman application interface. On the left, the sidebar displays 'riddhima bhanja's Workspace' with various collections, environments, and flows. The main workspace shows a 'Flight Booking Microservices / Health Checks / Eureka Server' collection. A 'GET' request is selected with the URL `http://localhost:8761/`. The response status is 200 OK, with a response time of 24 ms and a size of 5.47 KB. The response body is displayed as HTML, showing the Eureka Server's health check page.

```
<!doctype html>
<html lang="en">
<title>Eureka</title>
<meta name="description" content="">
<meta name="viewport" content="width=device-width">
<base href="/">
<meta charset="utf-8">
<link rel="stylesheet" href="eureka/css/wro.css">
```

## Flight Service Health:

The screenshot shows the Postman application interface. The sidebar is identical to the previous one. The main workspace shows a 'Flight Booking Microservices / Health Checks / Flight Service Health' collection. A 'GET' request is selected with the URL `http://localhost:8081/actuator/health`. The response status is 200 OK, with a response time of 35 ms and a size of 1.03 KB. The response body is displayed as JSON, showing the health status of the discovery composite and its components.

```
{
  "status": "UP",
  "components": {
    "discoveryComposite": {
      "status": "UP",
      "components": {
        "discoveryClient": {
          "status": "UP",
          "details": {
            "services": [
              "api-gateway",
              "flight-service",
              "booking-service"
            ]
          }
        }
      }
    }
}
```

## ADD FLIGHT

The screenshot shows the Postman interface with the following details:

- Collection:** riddhima bhanja's Workspace
- API:** POST Add Flight Inventory
- URL:** http://localhost:8080/api/v1/flight/inventory
- Body (raw JSON):**

```
1 {
2   "airline": "Air India",
3   "flightNumber": "AI101",
4   "fromPlace": "Delhi",
5   "toPlace": "Mumbai",
6   "departureDateTime": "2024-12-15T10:00:00",
```

- Response:** 201 Created (88 ms, 425 B)

## BOOK FLIGHT

The screenshot shows the Postman interface with the following details:

- Collection:** riddhima bhanja's Workspace
- API:** POST Book Flight
- URL:** http://localhost:8080/api/v1/booking/book/:flightId
- Path Variables:**

Key	Value	Description
flightId	692d822651cef85aec2564cb	Description

- Response:** 200 OK (92 ms, 717 B)

## SEARCH FLIGHTS

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Request Type:** POST
- URL:** http://localhost:8080/api/v1/flight/search
- Body (JSON):** (Empty)
- Response:** 200 OK (360 ms, 429 B)
  - Preview: A JSON object representing a flight search result.

```
1 [  
2 {  
3   "id": "692d022651cef05aec2564cb",  
4   "airline": "Air India",  
5   "flightNumber": "AI101",  
6   "fromPlace": "Delhi",  
7   "toPlace": "Mumbai",  
8   "departureDateTime": "2024-12-15T10:00:00",  
9   "arrivalDateTime": "2024-12-15T12:30:00",  
10  "totalSeats": 180,  
11  "availableSeats": 180,  
12  "ticketPrice": 5000.0,  
13  "flightStatus": "ACTIVE",  
14  "oneWayPrice": 5000.0,  
15  "roundTripPrice": 9000.0,
```

## CANCEL BOOKING

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Request Type:** DELETE
- URL:** http://localhost:8080/api/v1/booking/cancel/:pnr
- Headers (8):** (Hidden)
- Body (JSON):** (Empty)
- Response:** 200 OK (201 ms, 712 B)
  - Preview: A JSON object representing a booking cancellation result.

```
1 {  
2   "pnr": "PNR45010D76",  
3   "flightId": "692d022651cef05aec2564cb",  
4   "flightNumber": "AI101",  
5   "airline": "Air India",  
6   "fromPlace": "Delhi",  
7   "toPlace": "Mumbai",  
8   "departureDateTime": "2024-12-15T10:00:00",  
9   "arrivalDateTime": "2024-12-15T12:30:00",  
10  "userName": "John Doe",  
11  "userEmail": "john.doe@example.com",  
12  "journeyDate": "2024-12-15",  
13  "noOfSeats": 2,  
14  "mealType": "VEG",  
15  "totalAmount": 10000.0,
```

## GET BOOKING BY PNR

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:8080/api/v1/booking/:pnr`. The response body is:

```
1 {  
2   "pnr": "PNR252ABD96",  
3   "flightId": "692d822651ce1f85aec2564cb",  
4   "flightNumber": "AI101",  
5   "airline": "Air India",  
6   "fromPlace": "Delhi",  
7   "toPlace": "Mumbai",  
8   "departureDateTime": "2024-12-15T10:00:00",  
9   "arrivalDateTime": "2024-12-15T12:30:00",  
10  "userName": "John Doe"
```

## GET BOOKING HISTORY

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:8080/api/v1/booking/history/:email`. The response body is:

```
11 {  
12   "userName": "John Doe",  
13   "userEmail": "john.doe@example.com",  
14   "journeyDate": "2024-12-15",  
15   "noOfSeats": 2,  
16   "mealType": "VEG",  
17   "totalAmount": 10000.0,  
18   "bookingStatus": "CANCELLED",  
19   "bookingDateTime": "2025-12-01T17:26:04.147",  
20   "passenger": [
```

## **FALLBACK case**

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** BookingService API(docker)
- Request:** POST /bookings/book
- Body (raw JSON):**

```
1 {
2   "flightId": 1,
3   "passengerName": "riddhima",
4   "passengerEmail": "riddhimabhanja2003@example.com",
5   "passengerPhone": "9876543210".
```
- Response Status:** 201 Created
- Response Body (raw JSON):**

```
1 {
2   "pnr": null,
3   "flightNumber": null,
4   "passengerName": null,
5   "numberOfSeats": null,
6   "totalAmount": null,
7   "status": "FAILED",
8   "bookingDate": null,
9   "message": "Service temporarily unavailable. Please try again later."
10 }
```

## **SEND EMAIL**

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** notification service(docker)
- Request:** POST /api/notifications/send
- Body (raw JSON):**

```
6   "subject": "Welcome Test",
7   "templatedData": [
8     "customerName": "Test User",
9     "customerId": "CUST789"
10   ]
11 }
```
- Response Status:** 201 Created
- Response Body (raw JSON):**

```
1   {
2     "notificationId": "6935c6a955627f03b8b65198",
3     "customerId": "CUST789",
4     "customerEmail": "riddhimabhanja2003@gmail.com",
5     "status": "SENT",
6     "message": "Email sent successfully",
7     "timestamp": "2025-12-07T18:25:49.718726912"
8   }
```

## GET BY ID

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Request:** GET /api/notifications/6935a9a61ee8f9036919f375
- Headers:** (7)
- Query Params:** Key: Key, Value: Value
- Body:** JSON (Preview shows a response object with status: "SENT", errorMessage: null, createdAt: "2025-12-07T16:21:58.004", sentAt: "2025-12-07T16:21:59.059")
- Response:** 200 OK, 85 ms, 2.51 KB

## BOOKING CONFIRMATION EMAIL

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Request:** POST /api/notifications/send
- Headers:** (10)
- Body:** raw (JSON content: {"templateName": "booking-confirmation", "subject": "Flight Booking Confirmation - PNR12345678", "templateData": {"customerName": "riddhima bhanja", "pnr": "PNR12345678", "flightNumber": "AI101"}, "notificationId": "6935c67f55627f03b8b65197", "customerId": "CUST123", "customerEmail": "riddhimbhanja2003@gmail.com", "status": "SENT", "message": "Email sent successfully", "timestamp": "2025-12-07T18:25:07.687277843"})
- Response:** 201 Created, 6.01 s, 288 B

## GET ALL NOTIFICATION

The screenshot shows the Postman interface with the following details:

- Collection:** notification service(docker)
- Request:** GET http://localhost:8083/api/notifications/all
- Status:** 200 OK
- Body:** JSON response (partial view shown)

```
id": "69359bdbca6bb290bcb8712",
"customerId": "CUST123",
"customerName": "riddhima bhanja",
"customerEmail": "riddhimabhanja2003@example.com",
"templateName": "booking-confirmation",
"subject": "Booking Confirmation",
"body": "<!DOCTYPE html><html><head><meta charset=\"UTF-8\"></head><body><h1>Booking Confirmation</h1><p>Your booking confirmation has been sent to your email address.</p></body></html>"}
```

## GET CUSTOMER NOTIFICATION

The screenshot shows the Postman interface with the following details:

- Collection:** notification service(docker)
- Request:** GET http://localhost:8083/api/notifications/customer/CUST123
- Status:** 200 OK
- Body:** JSON response (partial view shown)

```
"status": "FAILED",
"errorMessage": "Failed to send email",
"createdAt": "2025-12-07T15:11:04.1",
"sentAt": null}
```

## CREATE BOOKING

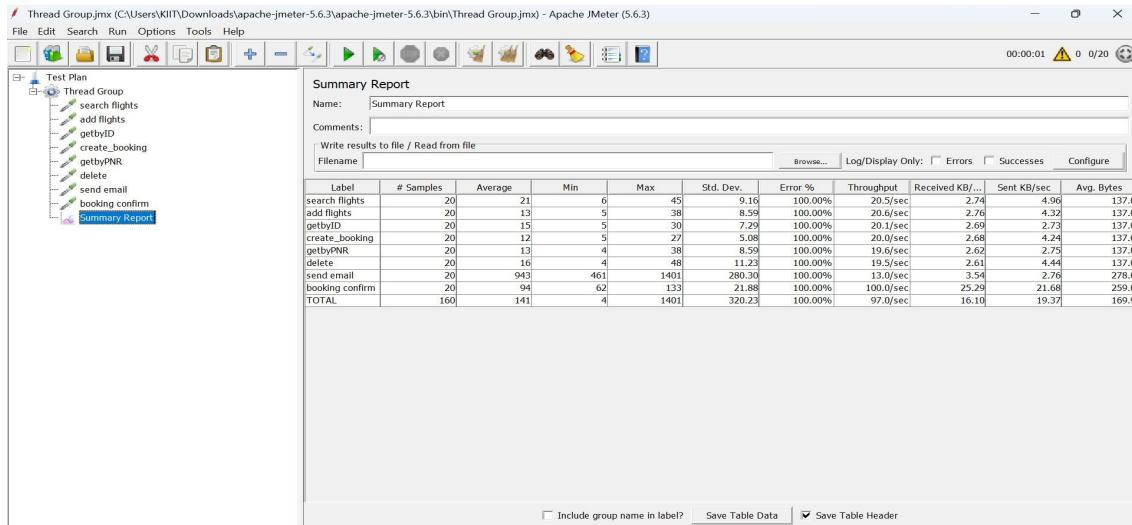
The screenshot shows the Postman interface. On the left, the sidebar lists collections, environments, history, flows, and files. The main area shows a POST request to 'BookingService API(docker) / create booking' with the URL `http://localhost:8080/api/bookings/book`. The 'Headers' tab is selected, showing a Content-Type header set to application/json. The 'Body' tab shows a JSON response with fields like PNR, flightNumber, passengerName, numberofSeats, totalAmount, status, bookingDate, and message. The status bar at the bottom indicates a 201 Created response.

- Jmeter result tree

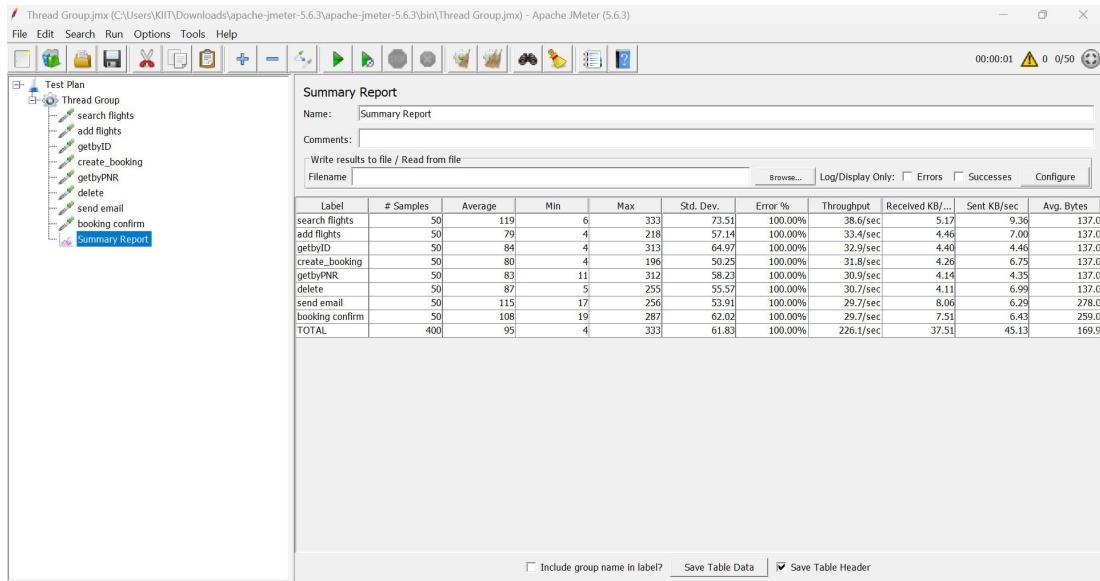
The screenshot shows the Apache JMeter interface with a 'Flight Microservices Load Test' plan. The 'View Results Tree' listener is selected. The results tree displays multiple 'Search Flights' sampler results, each with a green checkmark icon. The 'Text' column shows the sampler name, and the 'Sampler result' column shows the raw response data. The status bar at the top right shows 0 errors and 0 successes.

- Jmeter Summary report

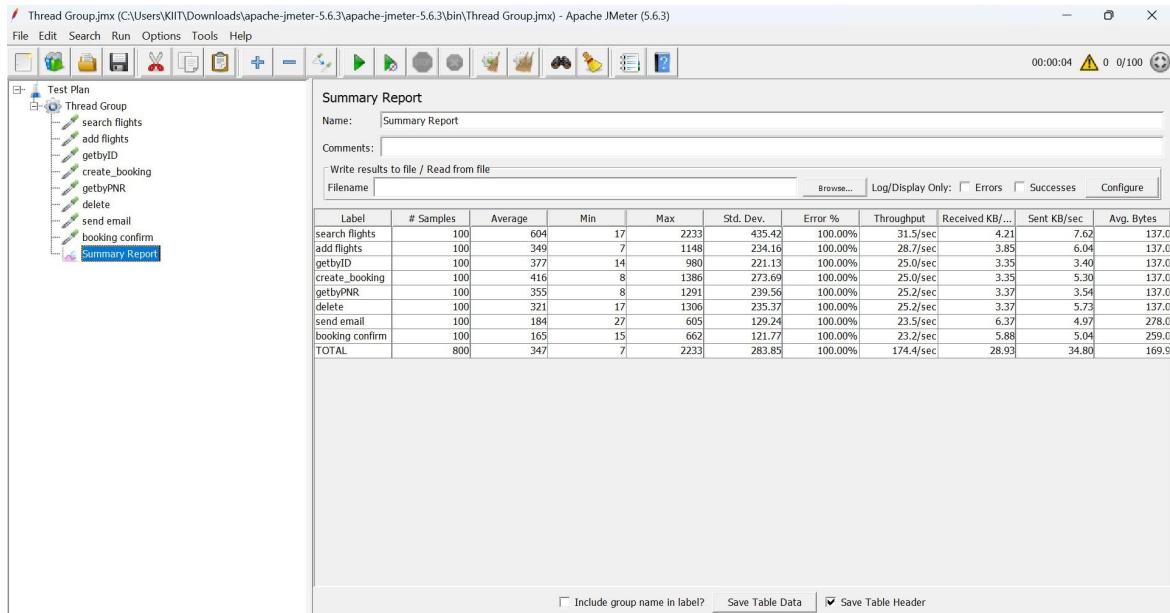
## 20 Requests



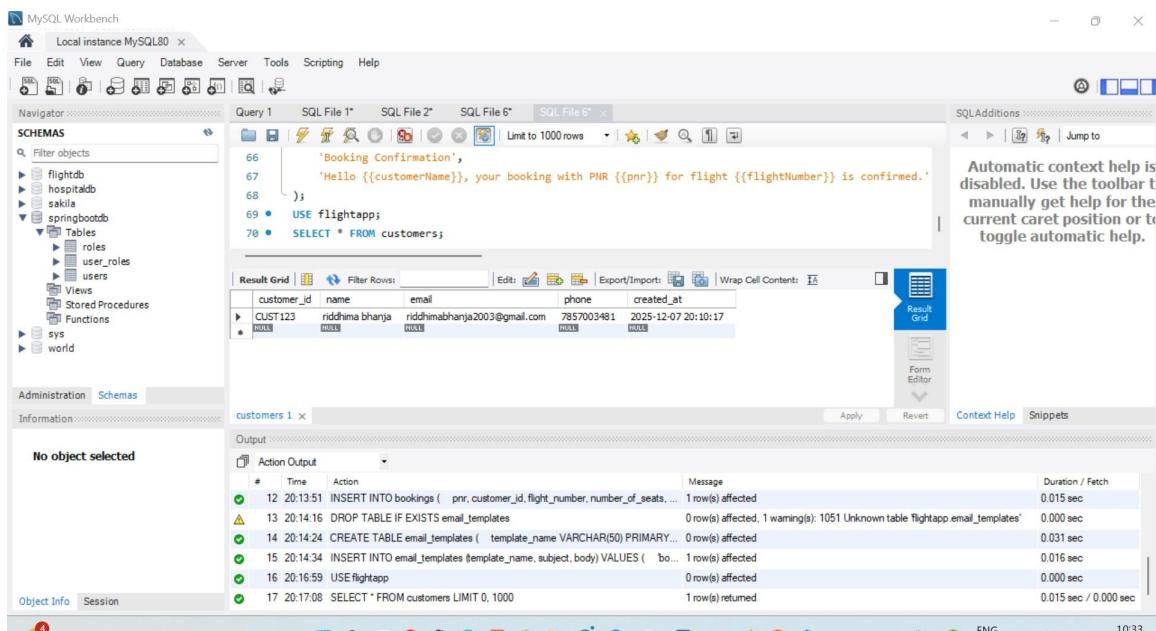
## 50 Requests



## 100 Requests



## MySQL Workbench



# EUREKA DASHBOARD

The screenshot shows the Spring Eureka dashboard interface. At the top, there's a header bar with a back arrow, a search icon, a star icon, a sign-in button, and a chat icon. The main title is "spring Eureka". Below the header, the page is titled "System Status". It displays various system metrics:

Environment	test	Current time	2025-12-07T16:36:21+0000
Data center	default	Uptime	00:21
		Lease expiration enabled	true
		Renew threshold	0
		Renew (last min)	8

A note at the bottom of this section states: "THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS."

Below the System Status, there's a section for "DS Replicas" which lists instances currently registered with Eureka:

Application	AMIs	Availability Zones	Status
API-GATEWAY	n/a (1)	(1)	UP (1) - api-gateway:80a8172b0ef35671c484-c34e-3542-cf
BOOKING-SERVICE	n/a (1)	(1)	UP (1) - booking-service:65d42243511f92ad2efeb060a16a443
FLIGHT-SERVICE	n/a (1)	(1)	UP (1) - flight-service:55125ccb520e35391c750420a1aba2c8
NOTIFICATION-SERVICE	n/a (1)	(1)	UP (1) - 23b5e63ac0d6-notification-service:8083

Finally, there's a "General Info" section with various system metrics:

Name	Value
total-avail-memory	87mb
num-of-cpus	8
current-memory-usage	53mb (60%)
server-uptime	00:21
registered-replicas	
unavailable-replicas	
available-replicas	

At the bottom left, there's a link labeled "Instance Info".