

**CHUBB®**

## **WEEK-7 ASSIGNMENT**

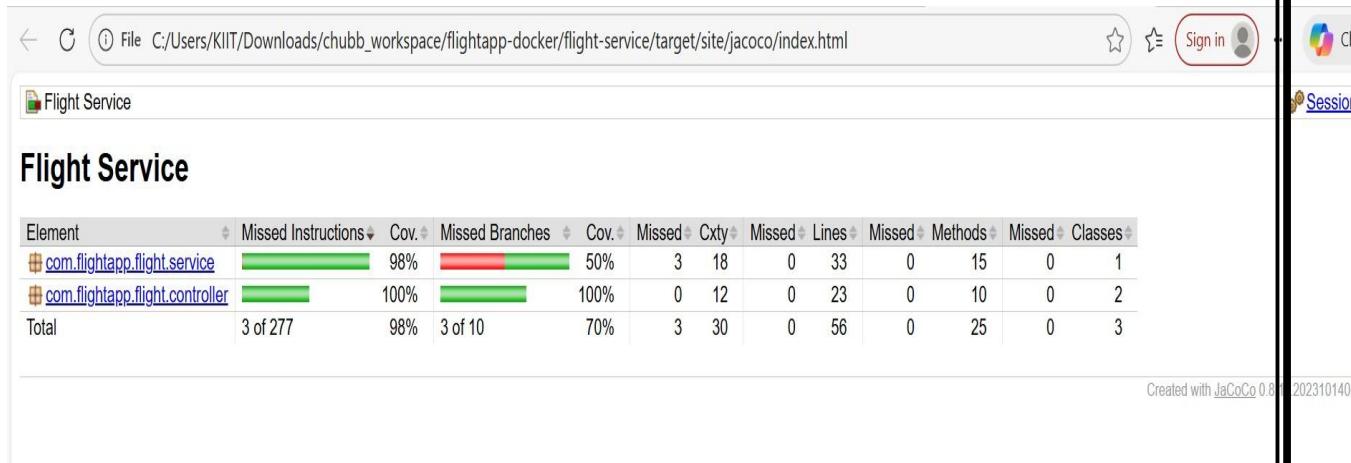
-Riddhima Bhanja  
Kalinga Institute of Industrial Technology  
Bhubaneswar(Java Track)

# 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

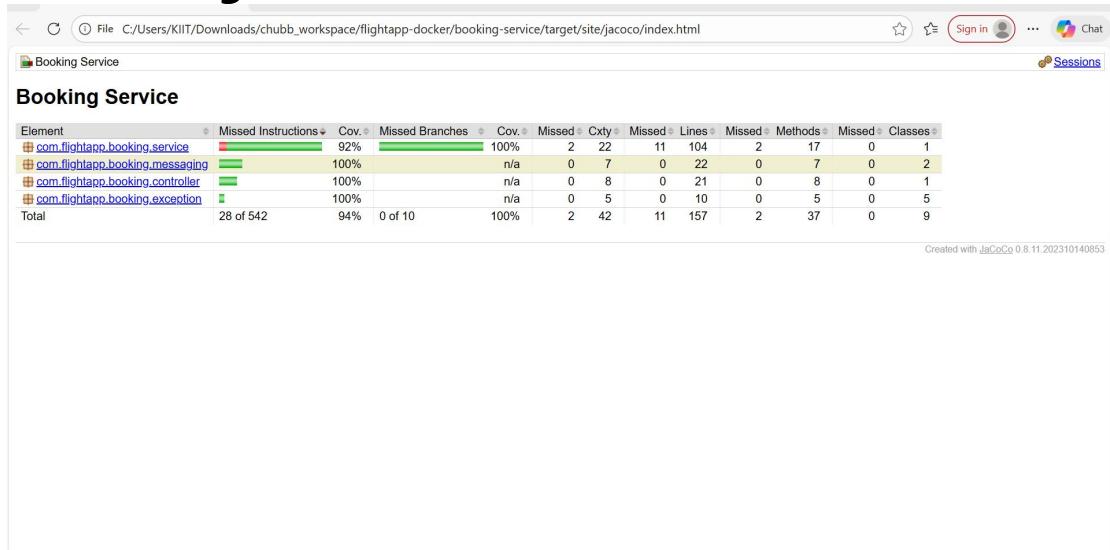
# JACOCO REPORTS

## FLIGHT SERVICE: 98% COVERAGE

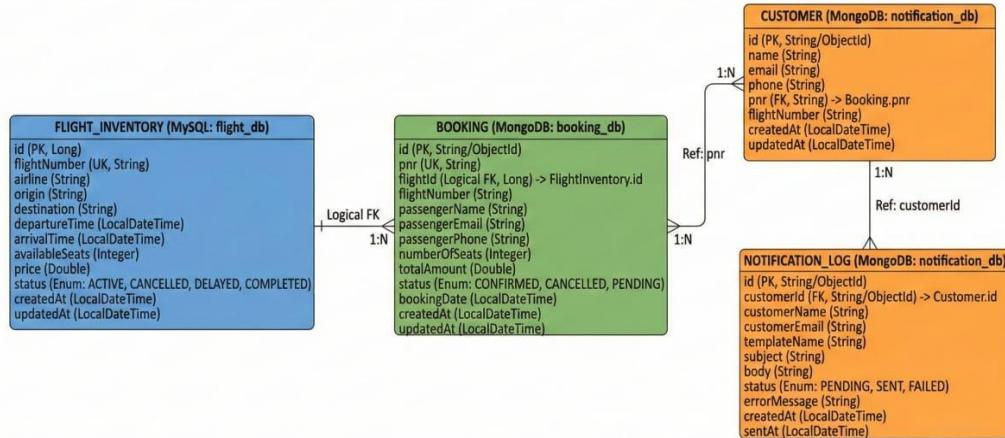


## BOOKING SERVICE

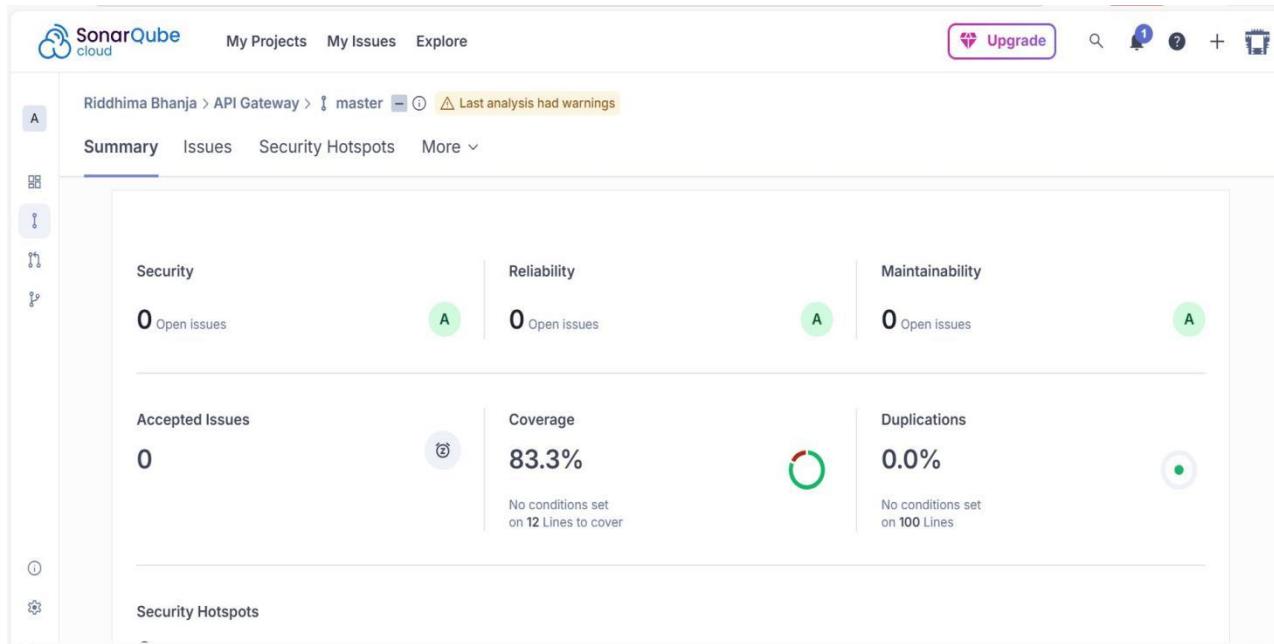
### 94% coverage



## 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 shows a summary of the 32 issues with details like effort (4h 23min), tags (Consistency, Intentionality, Adaptability), and priority (Major, Critical).

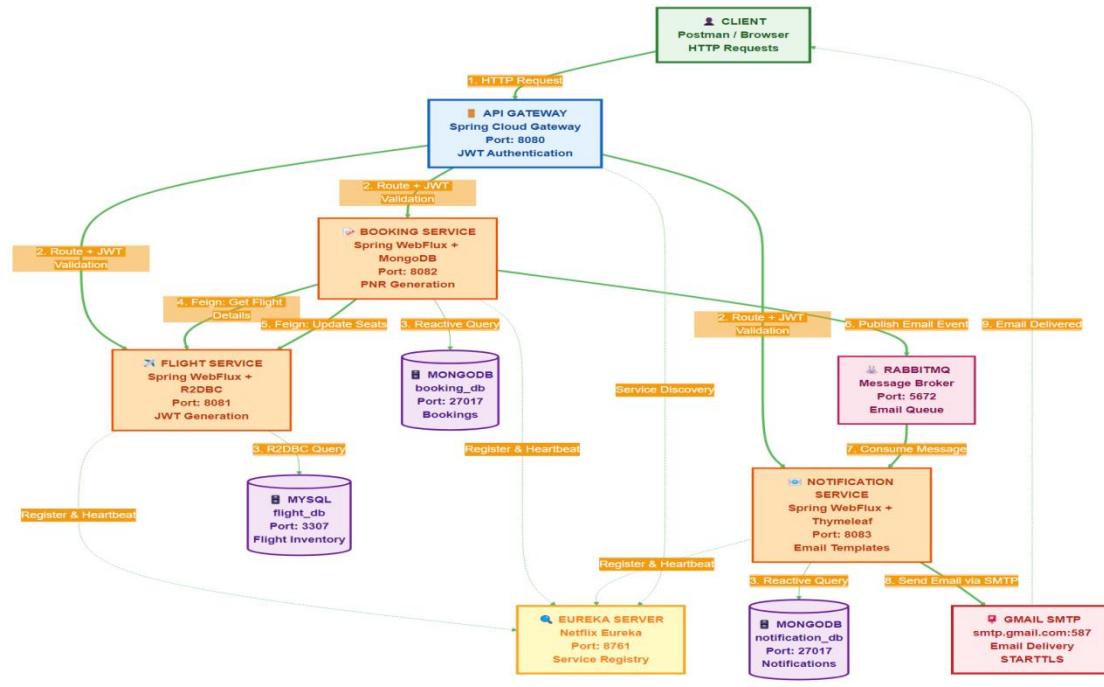
The screenshot shows the SonarQube Overall Summary page for the project 'Riddhima Bhanja > docker > main'. The left sidebar includes sections for Overview, Main Branch (selected), Pull Requests, Branches, Information, and Administration. The main area displays a summary of the project's status. Key metrics shown are: Security (1 Open issue, E), Reliability (4 Open issues, C), Maintainability (29 Open issues, A), Accepted Issues (0), Coverage (0.7%, No conditions set on 3k Lines), and Security Hotspots (0).

## After fixing:

The screenshot shows the SonarCloud Issues page for the 'FlightApp\_Docker-Assignment7' project. The main content area displays a message: 'No Issues. Hooray!' with a small icon. The left sidebar includes sections for Overview, Main Branch (selected), Pull Requests, Branches, Information, and Administration. The top navigation bar has links for My Projects, My Issues, and Explore.

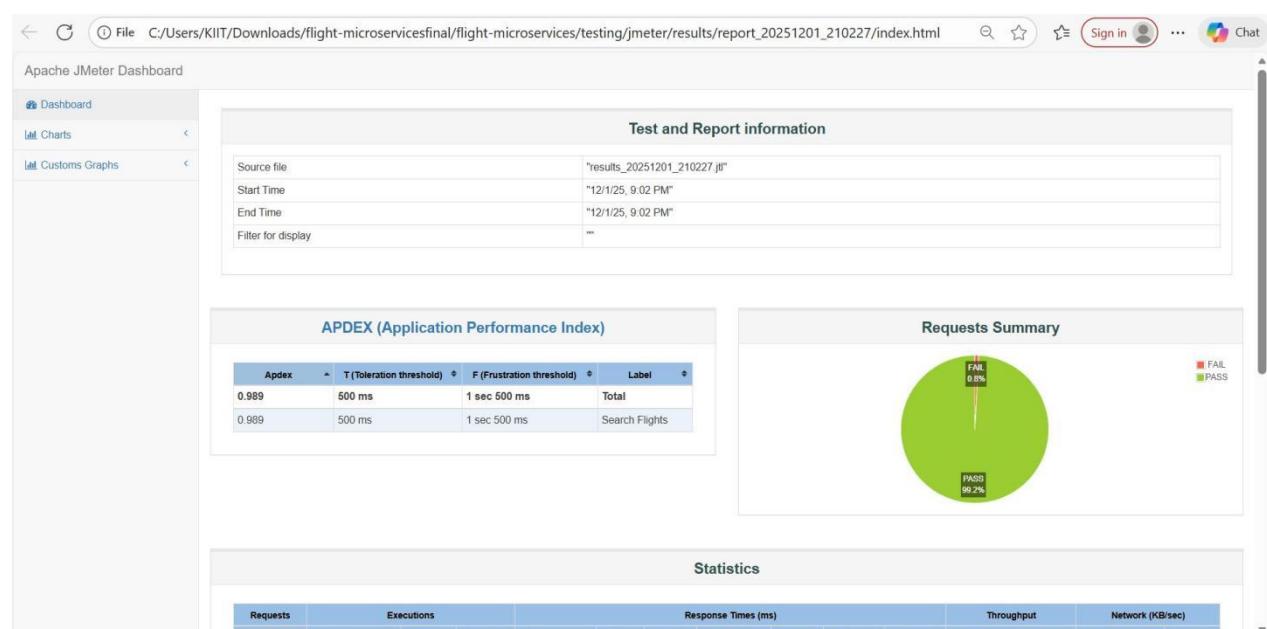
The screenshot shows the SonarCloud Overall Summary page for the 'FlightApp\_Docker-Assignment7' project. It displays various metrics: Security (0 Open issues, A grade), Reliability (0 Open issues, A grade), Maintainability (0 Open issues, A grade), Accepted Issues (0), Coverage (0.8%, Set up coverage analysis link), and Duplications (0.8%). The left sidebar includes sections for Overview, Main Branch (selected), Pull Requests, Branches, Information, and Administration. The top navigation bar has links for My Projects, My Issues, and Explore.

### **3. System Architecture**



### **4. Jmeter**

- Apache Jmeter Dashboard



**Statistics**

Label	Requests				Executions						Response Times (ms)						Throughput			Network (KB/sec)		
	#Samples	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 Delete

Container	Status	IP Address	Port	Action
mysql	Up	mysql:8.0	3307:3306	⋮
rabbitmq	Up	rabbitmq:3.1	15672:5672	Show all port
mongodb	Up	mongo:7.0	27017:27017	⋮
mailhog	Up	mailhog/mailhog:1025:1025	c	Show all port
flight	Up	flightapp-docker:8081	8081:8081	⋮
booking	Up	flightapp-docker:8082	8082:8082	⋮
api-gate...	Up	flightapp-docker:8080	8080:8080	⋮
eureka...	Up	flightapp-docker:8761	8761:8761	⋮
notificat...	Up	...	...	⋮

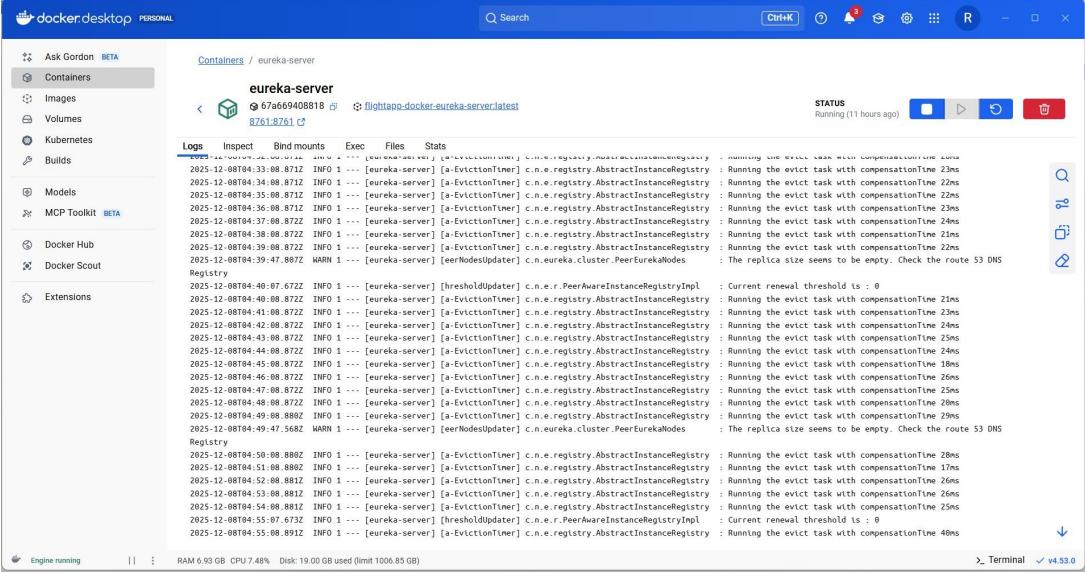
```

"ctx":"conn350880","msg":"client metadata","attr": {"remote":"127.0.0.1:43872","client":"conn350880","negotiatedCompressors":[],"doc":{"application":{"name":"mongosh 2.5.9"},"driver":{"name":"nodejs|mongosh","version":"6.19.0|2.5.9"},"platform":"Node.js v28.19.5, LE","os":{"name":"linux","architecture":"x64","version":"3.19.8-327.22.2.el7.x86_64"},"type":"Linux"},"env":{"container":{"runtime":"docker"}}, [{"t":{"$date":"2025-12-08T04:51:51.461+00:00"}, "s":"I", "c":"ACCESS", "id":10483960,"ctx":"conn350880","msg":"Connection not authenticating","attr": {"client":"127.0.0.1:43872","doc":{"application":{"name":"mongosh 2.5.9"},"driver":{"name":"nodejs|mongosh","version":"6.19.0|2.5.9"},"platform":"Node.js v28.19.5, LE","os":{"name":"linux","architecture":"x64","version":"3.19.8-327.22.2.el7.x86_64"},"type":"Linux"},"env":{"container":{"runtime":"docker"}}, [{"t":{"$date":"2025-12-08T04:51:51.461+00:00"}, "s":"I", "c":"NETWORK", "id":6788700, "ctx":"conn350880","msg":"Received first command on ingress connection since session start or auth handshake","attr": {"elapsMillis":2}}]}, {"t":{"$date":"2025-12-08T04:51:51.487+00:00"}, "s":"I", "c":"NETWORK", "id":22944, "ctx":"conn350880","msg":"Connection ended","attr": {"remote":"127.0.0.1:43872","isLoadBalanced":false,"uid":{"$uuid":"4a6ff64d-ec02-4372-a3d3-13a0121bd72b"}}, "connectionId":350880, "connectionCount":10}], [{"t":{"$date":"2025-12-08T04:51:51.487+00:00"}, "s":"I", "c":"NETWORK", "id":22944, "ctx":"conn350877","msg":"Connection ended","attr": {"remote":"127.0.0.1:43844","isLoadBalanced":false,"uid":{"$uuid":"50324352-493c-42f2-bdc1-a121cb7cf4d7"}, "connectionId":350877, "connectionCount":9}], {"t":{"$date":"2025-12-08T04:51:51.487+00:00"}, "s":"I", "c":"NETWORK", "id":22944, "ctx":"conn350879","msg":"Connection ended","attr": {"remote":"127.0.0.1:43838","isLoadBalanced":false,"uid":{"$uuid":"a2f299f8-d489-4b2e-86e6-7a10422ba93c"}, "connectionId":350876, "connectionCount":7}], {"t":{"$date":"2025-12-08T04:51:51.487+00:00"}, "s":"I", "c":"NETWORK", "id":22944, "ctx":"conn350878","msg":"Connection ended","attr": {"remote":"127.0.0.1:43848","isLoadBalanced":false,"uid":{"$uuid":"779b3f9f-6d10-4bc9-9ed1-8bb7a7c365a9"}, "connectionId":350878, "connectionCount":6}]}

```

Engine running | RAM 7.21 GB CPU 5.84% Disk 19.00 GB used (limit 1006.85 GB) Terminal v4.53.0

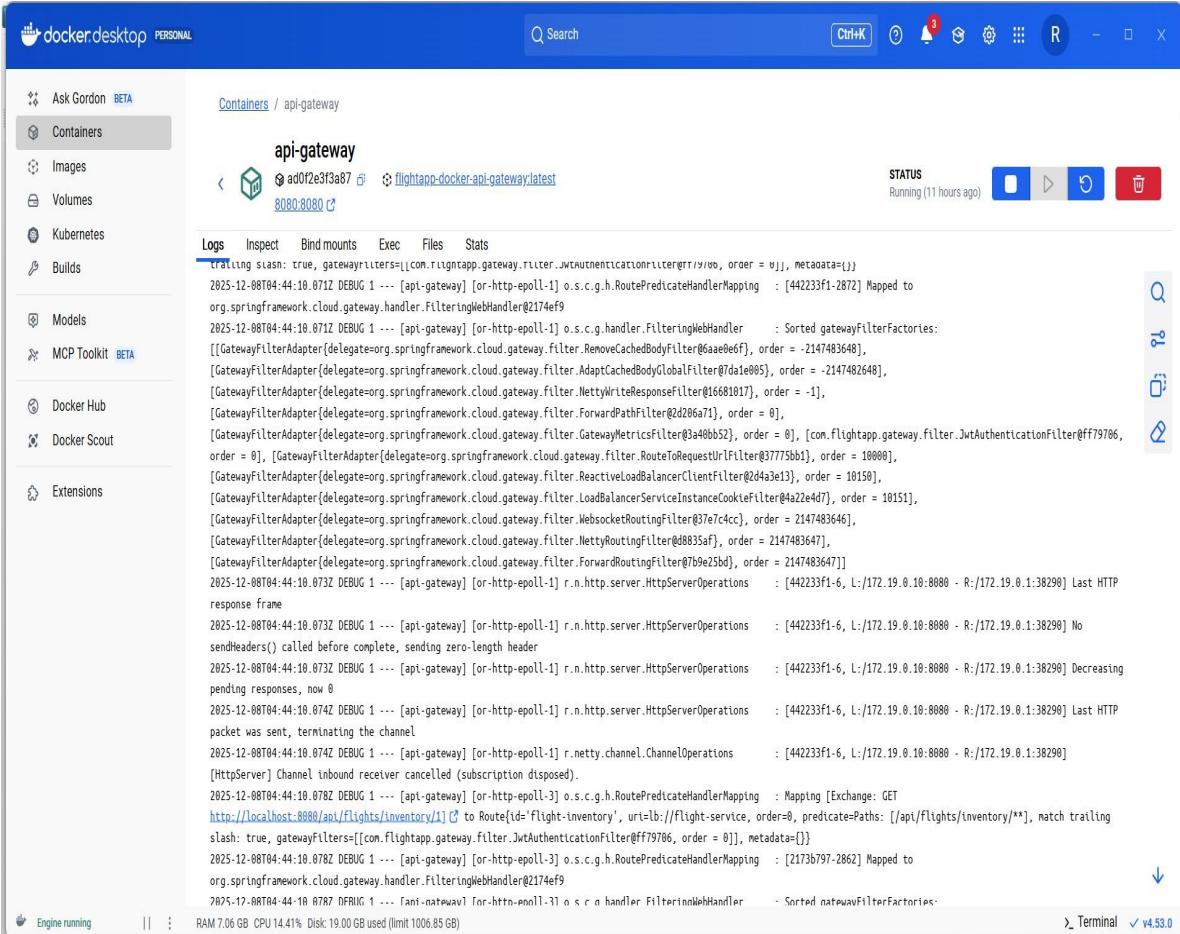
## EUREKA SERVER LOGS



Docker Desktop interface showing the logs for the eureka-server container. The logs output information about the eviction task running every 20ms, with compensation time varying between 23ms and 24ms. It also shows periodic tasks for threshold updates 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 23ms
2025-12-08T04:34:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 22ms
2025-12-08T04:35:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 22ms
2025-12-08T04:36:08.871Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 23ms
2025-12-08T04:37:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 24ms
2025-12-08T04:38:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 23ms
2025-12-08T04:39:08.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 22ms
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
2025-12-08T04:40:09.872Z INFO 1 --- [eureka-server] [thresholdUpdater] c.n.e.PeerAwareInstanceRegistryImpl : Current renewal threshold is : 8
2025-12-08T04:40:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 21ms
2025-12-08T04:41:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 23ms
2025-12-08T04:42:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 24ms
2025-12-08T04:43:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 25ms
2025-12-08T04:44:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 24ms
2025-12-08T04:45:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 18ms
2025-12-08T04:46:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:47:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 25ms
2025-12-08T04:48:09.872Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:49:09.886Z 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
2025-12-08T04:50:09.898Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 28ms
2025-12-08T04:51:09.898Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 17ms
2025-12-08T04:52:09.891Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:53:09.891Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 26ms
2025-12-08T04:54:09.891Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 25ms
2025-12-08T04:55:09.673Z INFO 1 --- [eureka-server] [thresholdUpdater] c.n.e.PeerAwareInstanceRegistryImpl : Current renewal threshold is : 8
2025-12-08T04:55:09.891Z INFO 1 --- [eureka-server] [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 40ms
```

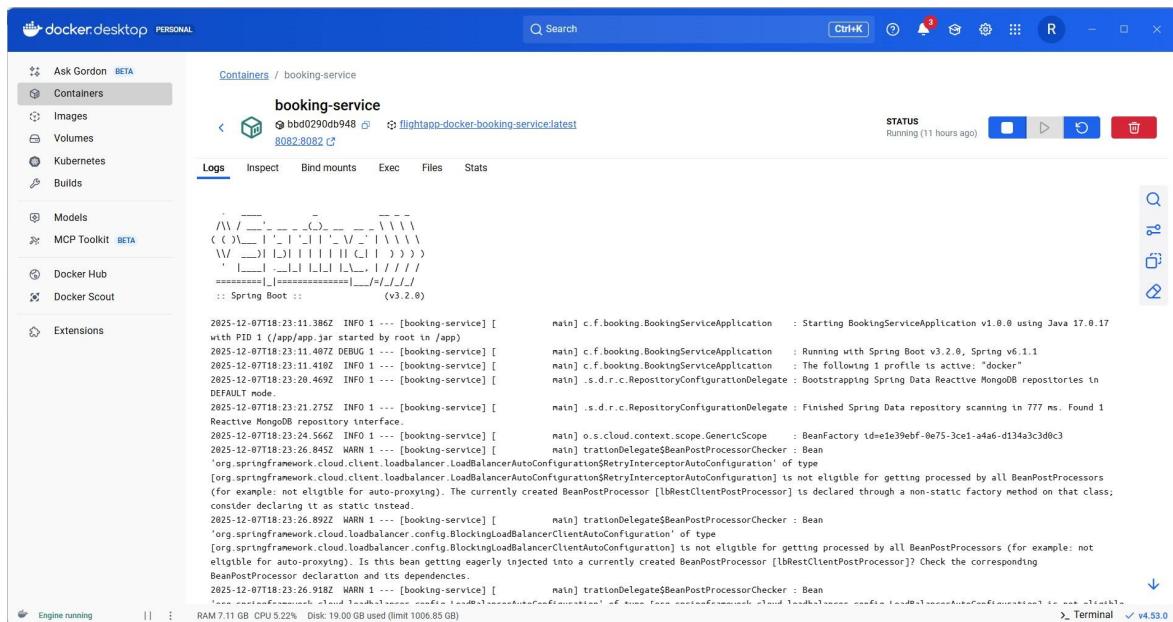
## API GATEWAY



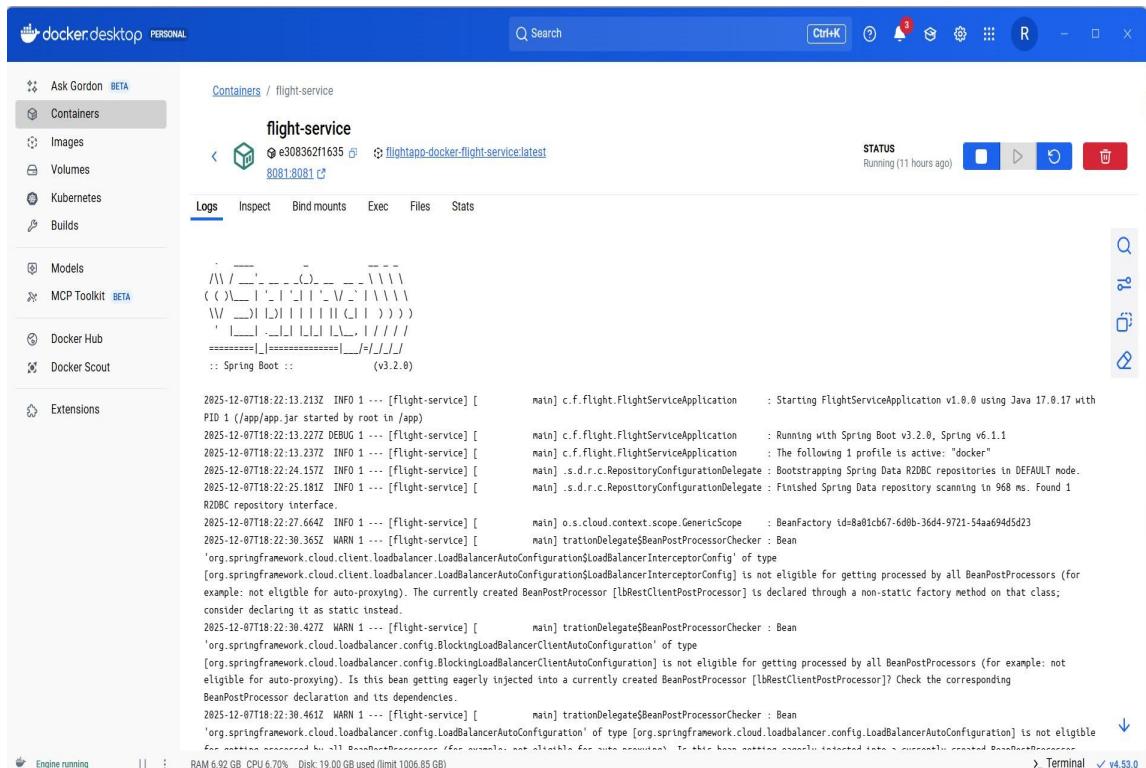
Docker Desktop interface showing the logs for the api-gateway container. The logs detail the configuration of gateway filters, including JwtAuthenticationFilter, NettyWriteResponseFilter, and various metrics and route filters. It also shows the handling of HTTP requests and responses.

```
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.RemoveCachedBodyFilter@6aaee0ef], order = 2147483648], [GatewayFilterAdapter[delegate=org.springframework.cloud.gateway.filter.AdaptCachedBodyGlobalFilter@7da1e005], order = -2147482648], [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@3a40bb52], order = 0], [com.flighatsapp.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 = 18150], [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@88835af], 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.flighatsapp.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:
```

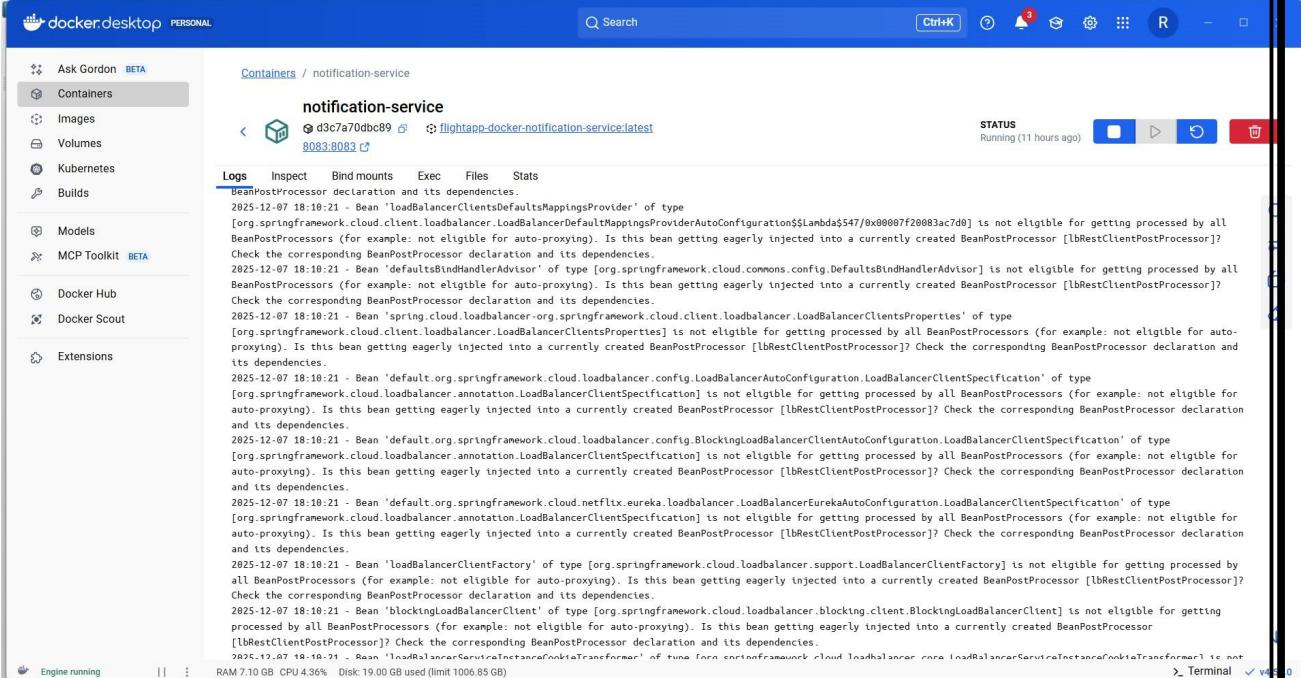
## **BOOKING SERVICE LOGS**



# 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.
```

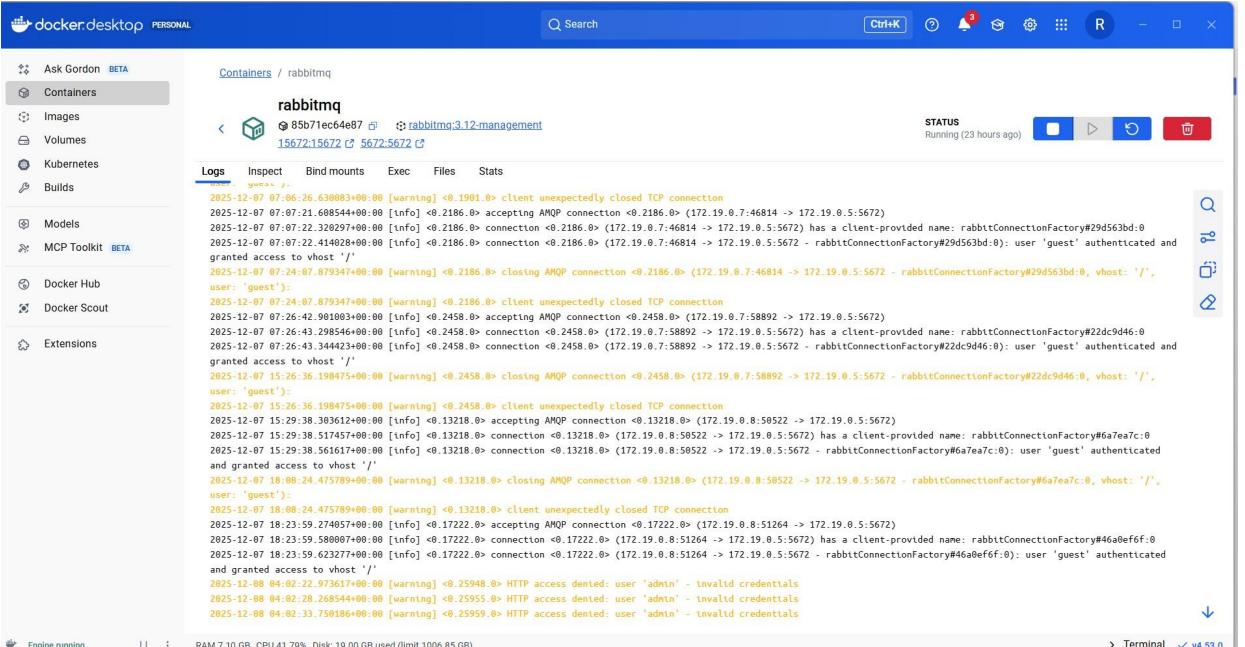
```
2025-12-07 18:19:21 - Bean 'loadBalancerClientFactory' of type [org.springframework.cloud.loadbalancer.support.LoadBalancerClientFactory] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying). Is this bean getting eagerly injected into a currently 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 07:09:26.630083+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 '/'
```

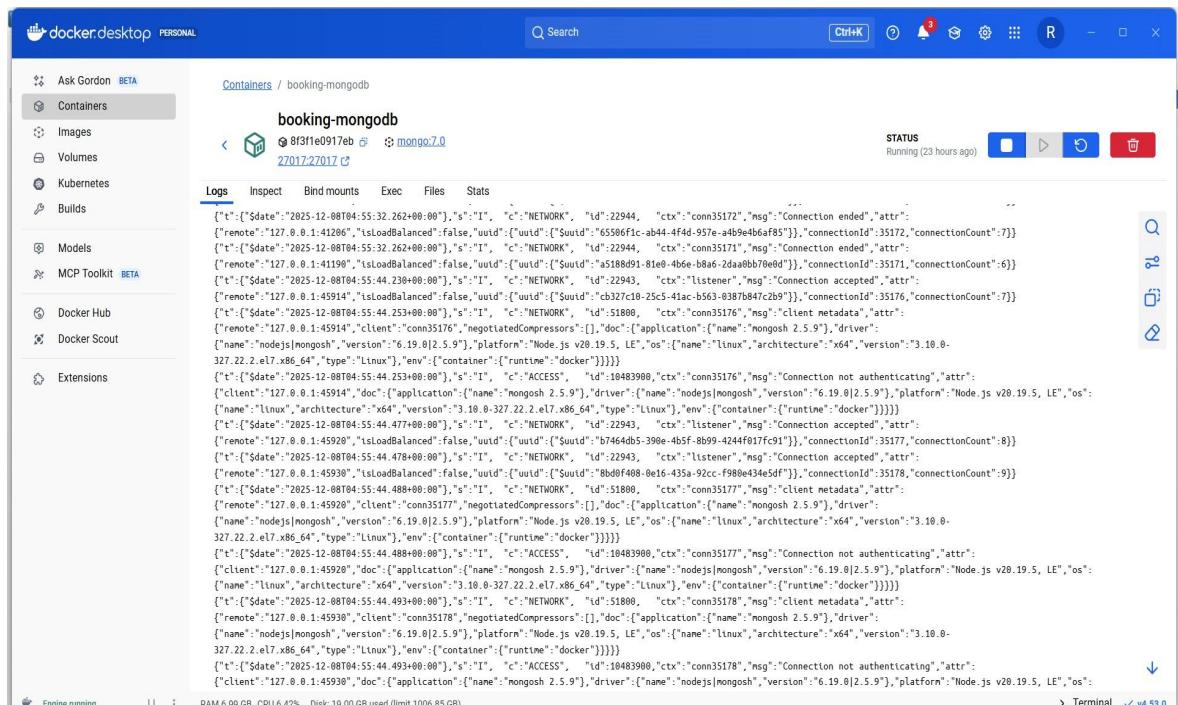
```
2025-12-08 04:02:22.973617+00:00 [warning] <0.25948.0> HTTP access denied: user 'admin' - invalid credentials
```

```
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
```

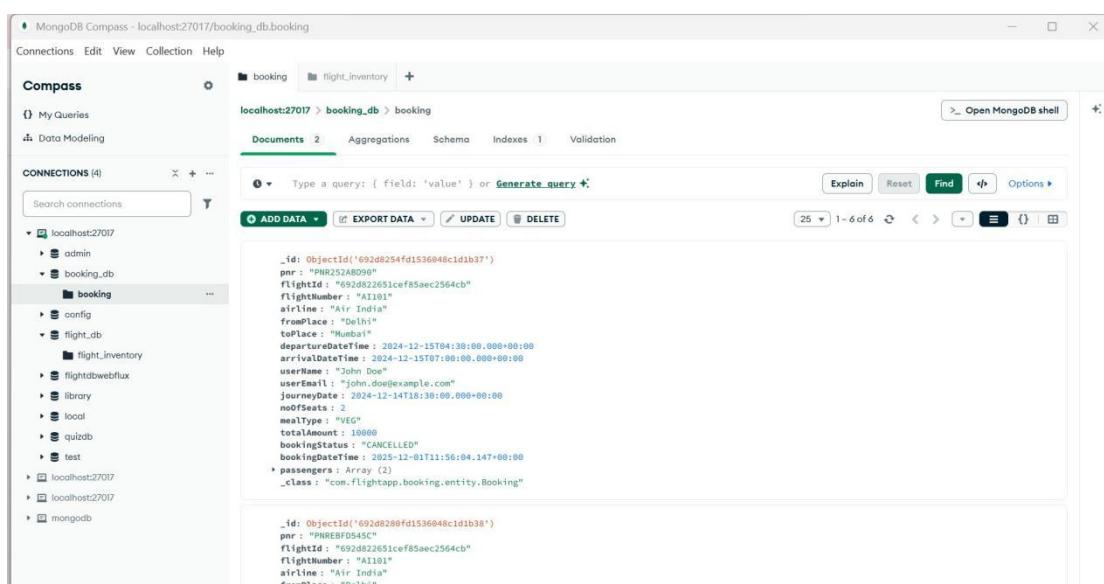
Logs Engine running RAM 7.10 GB CPU 41.79% Disk: 19.00 GB used (limit 1006.85 GB) Terminal v4.53.0

## 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
  - flighthdbwebflux
  - 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: 188
availableSeats: 174
ticketPrice: 5000
flightStatus: "ACTIVE"
oneWayPrice: 5000
roundTripPrice: 9000
mealAvailable: true
_class: "com.flightapp.flight.entity.FlightInventory"`

- Notification logs

MongoDB Compass - localhost:27017/flighthdbwebflux.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
  - flighthdbwebflux
    - booking
    - flight\_inventory
    - notification\_log
  - library
  - local
  - quizdb
  - test
- localhost:27017
- localhost:27017
- mongodb

localhost:27017 > flighthdbwebflux > 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: "692d828c54e03f4e58599e4386"
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('693880765a2c847745c44c2')
passengerId: "692d828c54e03f4e58599e4386"
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"`

## 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 body starts with a purple header 'Welcome Aboard!'. It then says 'Dear Test User,' followed by 'Welcome to our Flight Booking Service! We're excited to have you with us.' Below this is a yellow box containing 'Your Customer ID: CUST789'. The message continues with 'With our service, you can:' followed by a bulleted list: 'Search and book flights to destinations worldwide', 'Manage your bookings easily', 'Receive instant email confirmations', and 'Access 24/7 customer support'. It ends with 'Start exploring our flight options and book your next journey today!' and 'Best regards, Flight Booking Team'. At the bottom, there's a dark footer bar with the text '© 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 body starts with a green header 'Flight Booking Confirmed!'. It then says 'Dear riddhima bhanja,' followed by 'Your flight booking has been successfully confirmed. Below are your booking details:' and a 'Booking Information' section. This section lists: PNR:PNR12345678, Flight Number:AI101, Number of Seats:2, Total Amount:Rs. 10000, and Booking Date:2025-12-07. Below this is an 'Important Notes:' section with a bulleted list: 'Please arrive at the airport at least 2 hours before departure', 'Carry a valid photo ID for check-in', and 'Check-in opens 2 hours before departure'. It ends with 'Thank you for choosing our service!', 'Best regards, Flight Booking Team', and a dark footer bar.

- Flight cancel email

**Flight Booking Cancelled**

riddhimabhanja2003@gmail.com  
to john.doe

Dear John Doe,

Your flight booking has been cancelled.

Booking Details:  
PNR: PNR45010D76  
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

**Queued messages [last minute]**

Ready	Unacked	Total
0	0	0

**Message rates [last minute]**

Publish	Deliver (Per ACK)	Consumer ACK	Get (Per ACK)	Get (empty)
0.00/s	0.00/s	0.00/s	0.00/s	0.00/s

**Details**

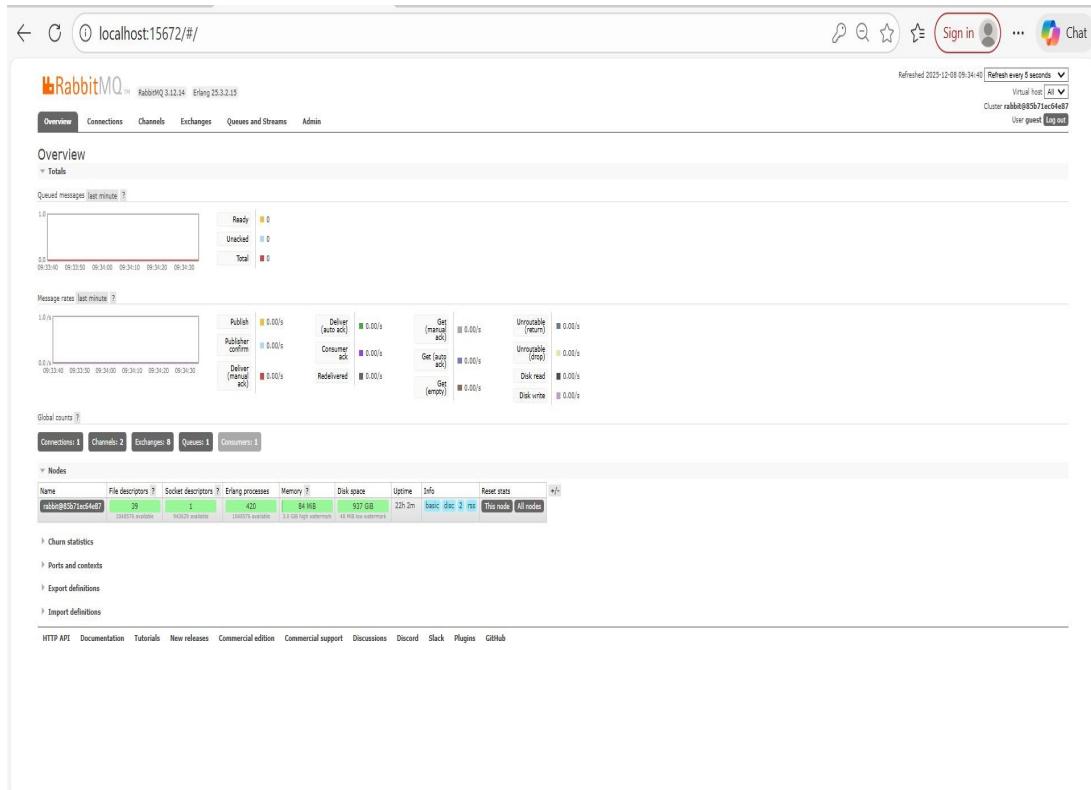
Features	durable: true	State	Consumers	Consumer capacity	Messages	Total	Ready	Unacked	In memory	Persistent	Transient	Paged Out
Policy		idle	1	100%	Message body bytes	0	0	0	0	0	0	0
Operator policy					process memory	8.8	8.8	0.8	0.8	0	0	0
Effective policy definition						10 kB						

**Consumers (1)**

- Bindings (2)
- Publish message
- Get messages
- Move messages
- Delete
- Purge
- Routine 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, 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 JSON and contains the following payload:

```
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 JSON:

```
1 {
2   "token": "eyJhbGciOiJIUzIwMjQ9.eyJzdWIiOiJhZG1pbkiIsImhdCI6MTc2NTEzNDM5MiwiZXhwIjoxNjY1MjIwNzkyfQ.8FY9CwGdLfUb0uQo0zfZvk8LuHsCdUhvJhm1k3FRVxrtor66TBWEAIM4nWso4t;01vDzhKnVTQFUfTbk1_87w",
3   "username": "admin",
4   "message": "Login successful."
5 }
```

## Unauthorized access

The screenshot shows the Postman application interface. On the left, the sidebar displays the workspace name "riddhima bhanja's Workspace" and lists collections, environments, flows, and files. The main area shows a collection named "flightService API(docker) / unauthorized request". A specific endpoint "GET /api/flights/search" is selected, which is supposed to return an unauthorized response. The request body is set to raw JSON:

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

The response tab shows a 401 Unauthorized status with a timestamp of 46 ms and a size of 137 B. The preview shows the raw response body as "1".

### **Same endpoint with auth bearer token**

The screenshot shows the Postman application interface. On the left, there's a sidebar with navigation links: Home, Workspaces (selected), API Network, Collections, Environments, History, Flows, and Files (BETA). The main workspace title is "riddhima bhanja's Workspace". A search bar at the top right says "Search Postman". There are tabs for Invite, Upgrade, and other settings.

The main area displays a collection named "riddhima bhanja's Workspace". It contains several API endpoints:

- Collections**: GET API Gateway Health, GET Flight Service Health, GET Booking Service Health.
- Environments**: FlightBookingAPI, flightbookwebfluxAPI, flightService API(docker). The "flightService API(docker)" environment has a "GET obtain JWT token" endpoint.
- Flows**: POST search flights (selected), POST add flight, GET get flight by ID, GET unauthorized request.
- notification service(docker)**: POST send email, GET get notification by CUST\_ID, GET get notification by ID, GET Get all notification, POST booking confirm, GET get customer notification.

The "POST search flights" request is currently selected. The request URL is "http://localhost:8080/api/flights/search". The method dropdown shows "POST". The "Authorization" tab is active, showing "Auth Type: Bearer Token" and a placeholder "Token" with a yellow box highlighting the token value "\*\*\*\*\*". Below it, a tooltip explains: "The authorization header will be automatically generated when you send the request. Learn more".

The "Body" tab is selected, showing a JSON response with 13 items. The first few items are:

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

On the right side, there are status indicators: 200 OK, 6.05 s, 736 B, and buttons for Save Response, Run, and Visualize.

- Circuit Breaker

## Events:

The screenshot shows the Postman interface with the following details:

- Left Sidebar:** Shows collections like "api-gateway", "circuit breaker", and "Flight Booking Microservices".
- Request URL:** `circuit breaker / events`
- Method:** GET
- Response Status:** 200 OK
- Response Body (JSON):**

```
{
  "circuitBreakerEvents": [
    {
      "circuitBreakerName": "bookingServiceCircuitBreaker",
      "type": "ERROR",
      "creationTime": "2025-12-01T21:55:37.209614500+05:30[Asia/Calcutta]",
      "errorMessage": "java.util.concurrent.TimeoutException: Did not observe any item or terminal signal within 1000ms in 'circuitBreaker' (and no fallback has been configured)",
      "durationInMs": 1000,
      "stateTransition": null
    },
    {
      "circuitBreakerName": "flightServiceCircuitBreaker",
      "type": "ERROR",
      "creationTime": "2025-12-01T21:55:45.179704400+05:30[Asia/Calcutta]",
      "errorMessage": "java.util.concurrent.TimeoutException: Did not observe any item or terminal signal within 1000ms in 'circuitBreaker' (and no fallback has been configured)"
    }
  ]
}
```

## Status:

The screenshot shows the Postman interface with the following details:

- Left Sidebar:** Shows collections like "api-gateway", "circuit breaker", and "Flight Booking Microservices".
- Request URL:** `circuit breaker / status`
- Method:** GET
- Response Status:** 200 OK
- Response Body (JSON):**

```
{
  "circuitBreakers": [
    {
      "bookingServiceCircuitBreaker": {
        "failureRate": "1.0%",
        "slowCallRate": "1.0%",
        "failureRateThreshold": "50.0%",
        "slowCallRateThreshold": "100.0%",
        "bufferedCalls": 3,
        "failedCalls": 1,
        "slowCalls": 0,
        "slowFailedCalls": 0,
        "notPermittedCalls": 0,
        "state": "CLOSED"
      },
      "flightServiceCircuitBreaker": {
        ...
      }
    }
  ]
}
```

- 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 http://localhost:8080/actuator/health
- Response Status:** 200 OK
- Response Body (JSON):**

```
1 {  
2   "status": "UP",  
3   "components": {  
4     "discoveryComposite": {  
5       "status": "UP",  
6       "components": {  
7         "discoveryClient": {  
8           "status": "UP",  
9           "details": {  
10             "services": [  
11               "api-gateway",  
12               "flight-service",  
13               "booking-service"  
14             ]  
15           ]  
16         }  
17       }  
18     }  
19   }  
20 }
```

### Booking Service Health:

The screenshot shows the Postman interface with the following details:

- Workspace:** riddhima bhanja's Workspace
- Collection:** Flight Booking Microservices
- Request:** GET http://localhost:8082/actuator/health
- Response Status:** 200 OK
- Response Body (JSON):**

```
1 {  
2   "status": "UP",  
3   "components": {  
4     "circuitBreakers": {  
5       "status": "UP",  
6       "details": {  
7         "flightService": {  
8           "status": "UP",  
9           "details": {  
10             "failureRate": "-1.0%",  
11             "failureRateThreshold": "50.0%",  
12             "slowCallRate": "-1.0%",  
13             "slowCallRateThreshold": "100.0%",  
14             "bufferedCalls": 0,  
15             "slowCalls": 0,  
16           }  
17         }  
18       }  
19     }  
20   }  
21 }
```

## Eureka server:

The screenshot shows the Postman interface with the following details:

- Collection:** riddhima bhanja's Workspace
- Request:** GET /eureka
- Response Status:** 200 OK
- Response Body (HTML):**

```
1 <!doctype html>
2 <html lang="en">
3 <title>Eureka</title>
4 <meta name="description" content="">
5 <meta name="viewport" content="width=device-width">
6 <head>
7   <base href="/">
8   <meta charset="utf-8">
9   <title>Eureka</title>
10  <meta name="description" content="">
11  <meta name="viewport" content="width=device-width">
12  <link rel="stylesheet" href="eureka/css/wro.css">
13
14 </head>
```

## Flight Service Health:

The screenshot shows the Postman interface with the following details:

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

```
1 {
2   "status": "UP",
3   "components": [
4     "discoveryComposite": {
5       "status": "UP",
6       "components": [
7         "discoveryClient": {
8           "status": "UP",
9           "details": {
10             "services": [
11               "api-gateway",
12               "flight-service",
13               "booking-service"
14             ]
15           }
16         }
17       }
18     }
19   ]
20 }
```

## ADD FLIGHT

The screenshot shows the Postman interface for the 'Flight Booking Microservices / Flight Service / Add Flight Inventory' collection. The 'POST Add Flight Inventory' endpoint is selected. The request URL is `http://localhost:8080/api/v1/flight/inventory`. The request body is a JSON object:

```
1 {
2   "airline": "Air India",
3   "flightNumber": "AI101",
4   "fromPlace": "Delhi",
5   "toPlace": "Mumbai",
6   "departureDateTime": "2024-12-15T10:00:00",
7   "arrivalDateTime": "2024-12-15T12:30:00",
8   "totalSeats": 180,
9   "availableSeats": 180,
10  "ticketPrice": 5000.0,
11  "flightStatus": "ACTIVE",
12  "oneWayPrice": 5000.0,
13  "roundTripPrice": 9000.0,
14  "mealAvailable": true
15 }
16 }
```

The response status is 201 Created, with a response time of 885 ms and a response size of 425 B.

## BOOK FLIGHT

The screenshot shows the Postman interface for the 'Flight Booking Microservices / Booking Service / Book Flight' collection. The 'POST Book Flight' endpoint is selected. The request URL is `http://localhost:8080/api/v1/booking/book/:flightId`. The request body is a JSON object:

```
1 {
2   "pnr": "PNR45018076",
3   "flightId": "692d822651cef85aec2564cb",
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 }
```

The response status is 200 OK, with a response time of 92 ms and a response size of 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
- Method:** POST
- Body (JSON):** (Empty)
- Response Status:** 200 OK
- Response Body (JSON):**

```
1 [ { 2   "id": "692d022651cef85aec2564cb", 3   "airline": "Air India", 4   "flightNumber": "AI101", 5   "fromPlace": "Delhi", 6   "toPlace": "Mumbai", 7   "departureDateTime": "2024-12-15T10:00:00", 8   "arrivalDateTime": "2024-12-15T12:30:00", 9   "totalSeats": 180, 10  "availableSeats": 180, 11  "ticketPrice": 5000.0, 12  "flightStatus": "ACTIVE", 13  "oneWayPrice": 5000.0, 14  "roundTripPrice": 9000.0, 15 } ]
```

## 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
- Method:** DEL Canc
- Headers (8):** (Hidden)
- Body (JSON):** (Empty)
- Response Status:** 200 OK
- Response Body (JSON):**

```
1 { 2   "pnr": "PNR45016D76", 3   "flightId": "692d022651cef85aec2564cb", 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 to `http://localhost:8080/api/v1/booking/:pnr`. The response body is a JSON object:

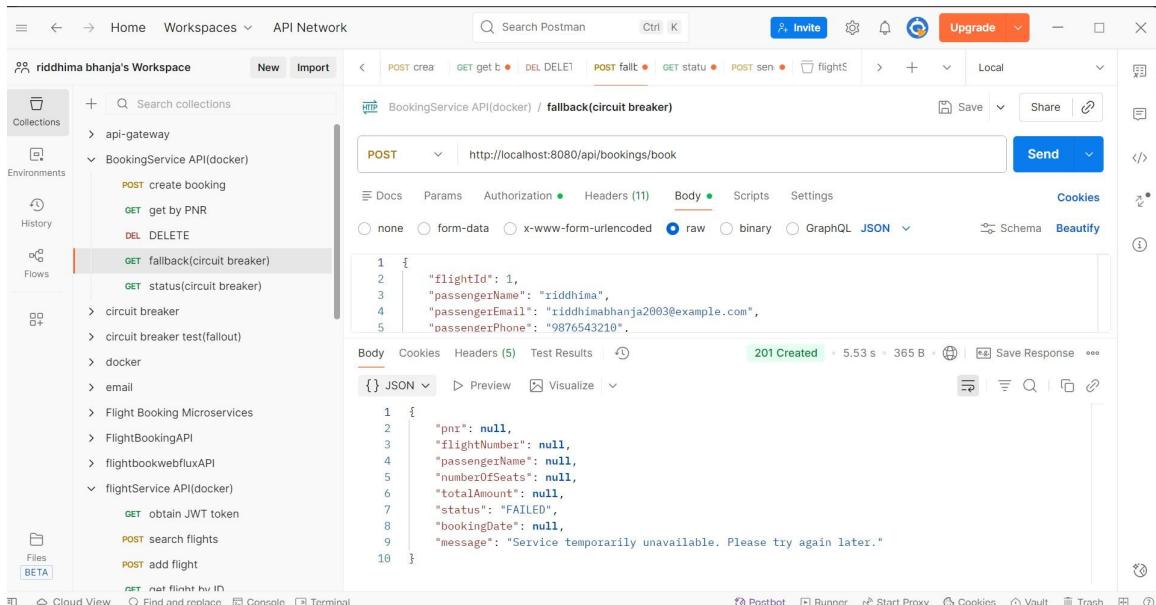
```
1 {  
2   "pnr": "PNR252ABD096",  
3   "flightId": "692d822651ce85aec2564cb",  
4   "flightNumber": "AI101",  
5   "airline": "Air India",  
6   "fromPlace": "Delhi",  
7   "toPlace": "Mumbai",  
8   "departureDateTime": "2024-12-15T10:00:00",  
9   "arrivalDateTime": "2024-12-15T12:30:00",  
10  "userName": "John Doe"
```

## GET BOOKING HISTORY

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

```
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:

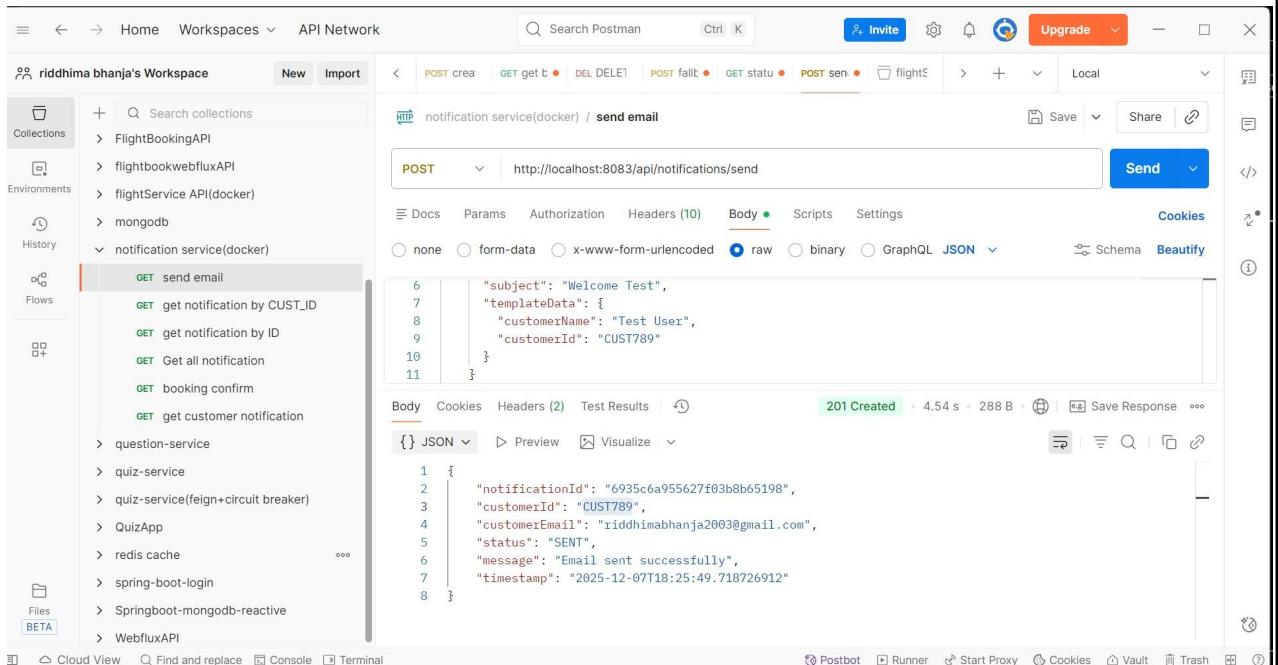
- Workspace:** riddhima bhanja's Workspace
- Collection:** BookingService API(docker)
- Request:** POST fallback(circuit breaker)
- URL:** http://localhost:8080/api/bookings/book
- Body (JSON):**

```
1 {
2   "flightId": 1,
3   "passengerName": "riddhima",
4   "passengerEmail": "riddhimabhanja2003@example.com",
5   "passengerPhone": "9876543210".
```

- Response Status:** 201 Created
- Response Body (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:** GET send email
- URL:** http://localhost:8083/api/notifications/send
- Body (JSON):**

```
6   "subject": "Welcome Test",
7   "templatedData": {
8     "customerName": "Test User",
9     "customerId": "CUST789"
10   }
11 }
```

- Response Status:** 201 Created
- Response Body (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 displays 'riddhima bhanja's Workspace' with various collections, environments, and flows. 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 the following content:

```
1  {
2     "pnr": "PNR7051E600",
3     "flightNumber": "AI101",
4     "passengerName": "riddhima",
5     "numberOfSeats": 2,
6     "totalAmount": 10000.0,
7     "status": "CONFIRMED",
8     "bookingDate": "2025-12-07T11:02:44.541433336",
9     "message": "Booking created successfully"
10 }
```

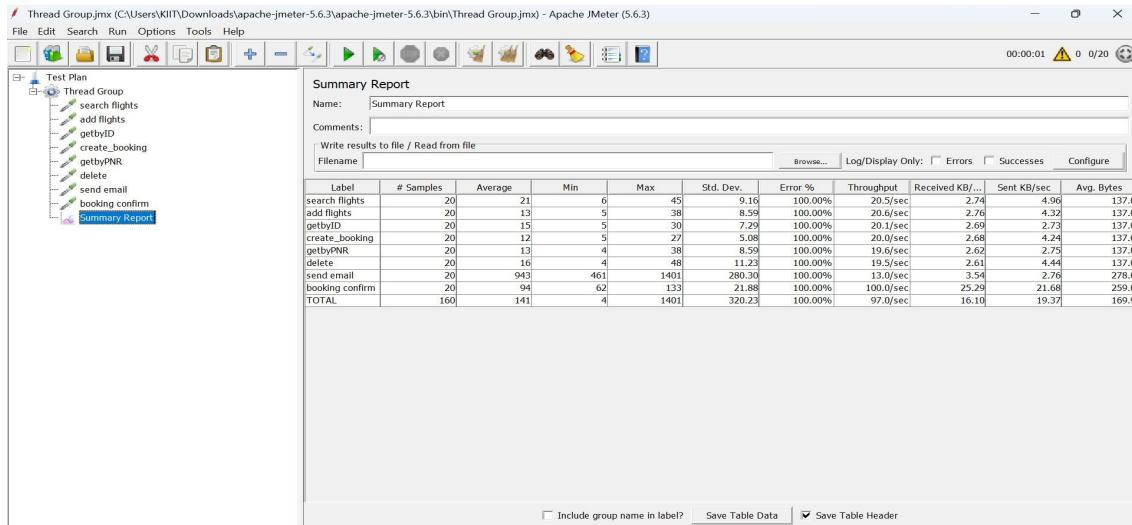
The status bar at the bottom indicates a '201 Created' response.

- Jmeter result tree

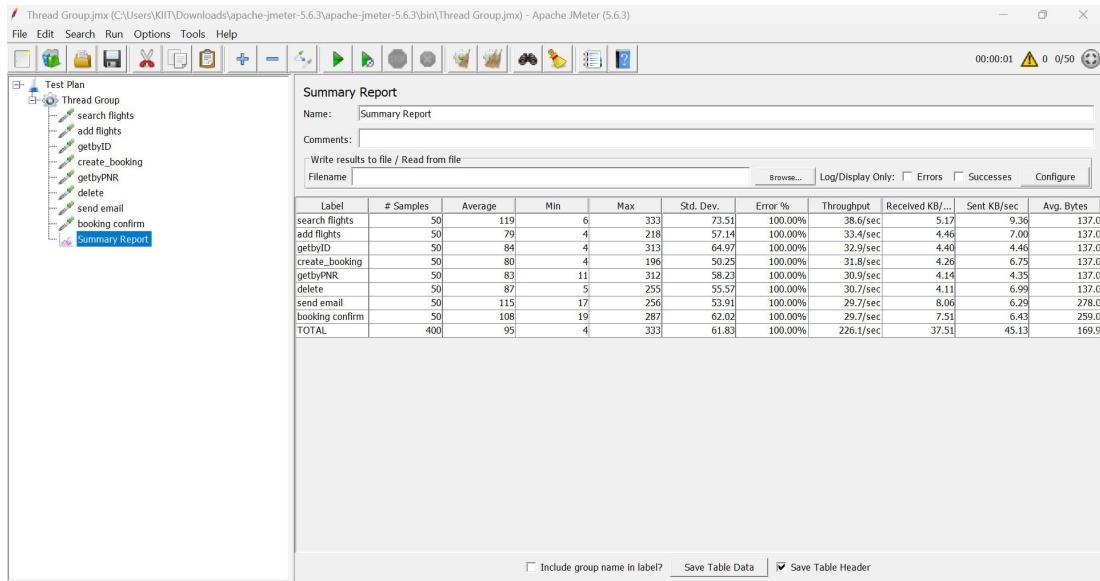
The screenshot shows the Apache JMeter interface with a test plan named 'FlightApp-LoadTest.jmx'. The 'View Results Tree' listener is selected. The results tree displays multiple 'Search Flights' sampler results, each with a green checkmark icon. The 'Text' column shows the raw response data, and the 'Sampler result' column shows the parsed JSON output. The status bar at the top right shows '00:00:09' and '0 0/50'.

- Jmeter Summary report

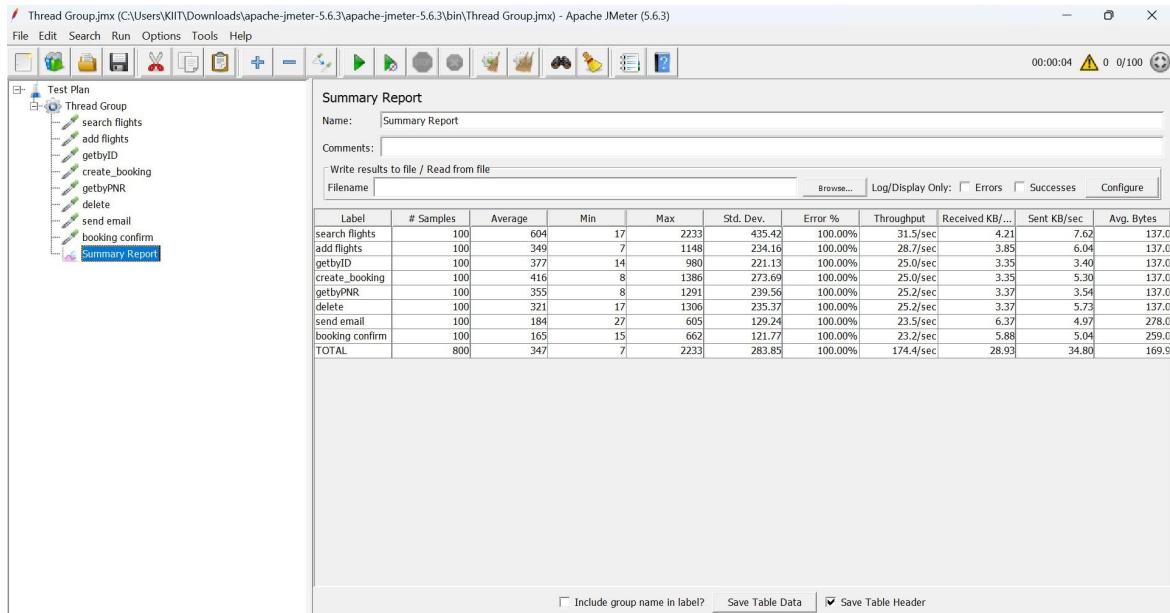
## 20 Requests



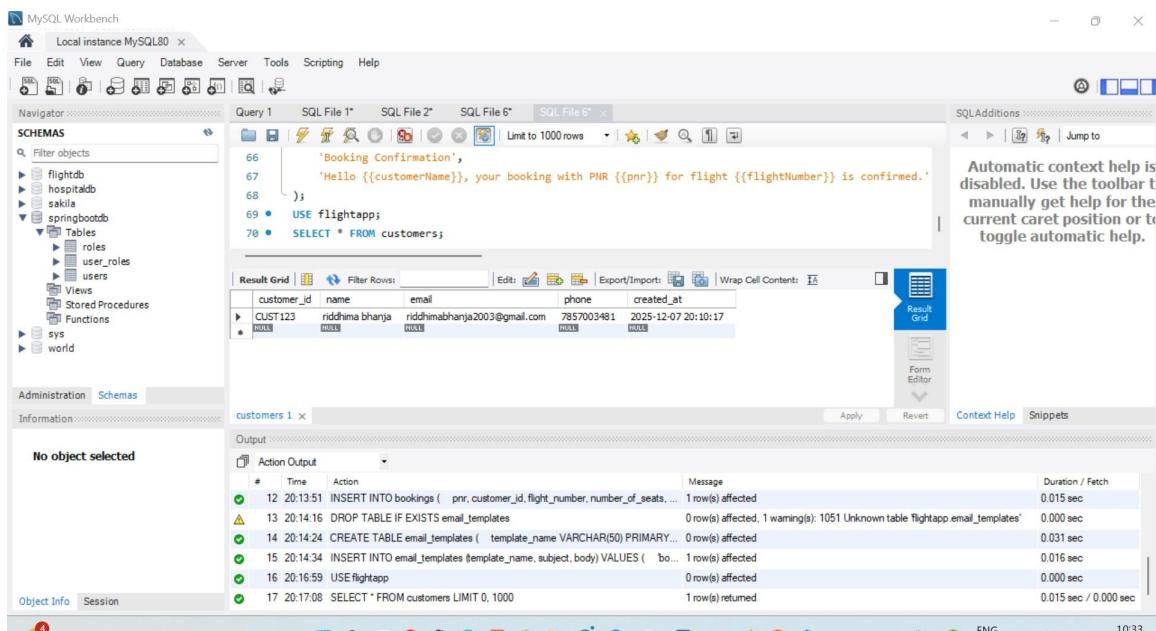
## 50 Requests



## 100 Requests



## MySQL Workbench



# EUREKA DASHBOARD

The screenshot shows the Spring Eureka dashboard interface. At the top, there's a header with a back button, a search bar, and user authentication links for 'Sign in' and 'Chat'. Below the header, the title 'spring Eureka' is displayed.

**System Status**

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

THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.

**DS Replicas**

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

**General Info**

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	

**Instance Info**