

# **Payment Service – Event Ticketing System**

## **Author – Amitosh Gautam**

### **Overview**

This documentation describes the design, development, and deployment of the Payment Service, which is one of the key microservices in the Event Ticketing and Seat Reservation System (ETS). The Payment Service manages financial transactions, including processing user charges, issuing refunds, and maintaining idempotent request handling to prevent duplicate operations. It operates as an independent Spring Boot microservice following the database-per-service pattern for full data isolation and scalability.

### **Service Responsibilities**

The Payment Service is responsible for:

- Processing new payment charges from confirmed orders.
- Recording and updating transaction statuses – PENDING, SUCCESS, FAILED, REFUNDED.
- Handling refund requests for successful payments.
- Enforcing idempotency using an Idempotency-Key to prevent duplicate charges.
- Providing RESTful APIs for order and finance modules to initiate or verify payments.
- Persisting all transactions in a dedicated PostgreSQL database.

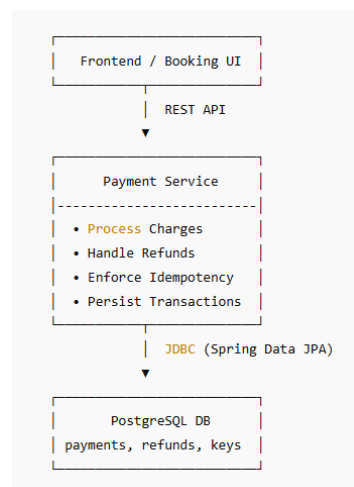
Each payment record is uniquely linked to an Order ID from the Order Service.

### **Architecture and Design**

The Payment Service follows a four-layered architecture:

- Controller Layer → Exposes REST endpoints to clients and other services.
- Service Layer → Implements business logic for charge processing, refunds, and idempotency.
- Repository Layer → Handles database persistence via Spring Data JPA.
- Model Layer → Defines entity classes for Payment, Refund, and Idempotency Key.

### **Architecture Diagram-**



### **Entity Relationship Diagram (ERD)-**

Entities:

1. Payment
  - Tracks each charge request.
  - Linked to an orderId.
  - Has status, amount, method, and reference.
2. Refund
  - Linked to a paymentId.
  - Tracks refund amount and reference number.
3. IdempotencyKey
  - Stores request fingerprints and cached responses to ensure idempotent POST operations.

Relationships:

- One Payment can have multiple Refunds.
- Each IdempotencyKey is unique per API call.

### **REST API Endpoints -**

Method	Endpoint	Description
POST	/v1/payments/charge	Process a payment charge (requires Idempotency-Key header)
POST	/v1/payments/refund	Refund a completed payment
GET	/v1/payments/{id}	Retrieve details of a specific payment
GET	/v1/payments/health	Health check endpoint
GET	/actuator/prometheus	Prometheus metrics for monitoring

### **Containerization with Docker –**

The Payment Service is fully containerized using Docker.

- Dockerfile: Builds a lightweight image from eclipse-temurin:17-jdk-alpine.
- docker-compose.yml: Deploys both the application and a dedicated PostgreSQL database.

Build and Run Commands:

```
mvn clean package -DskipTests
```

```
docker-compose up --build
```

Once running:

- Application: <http://localhost:8080/v1/payments/health>
- Database: localhost:5432 (paymentsdb)

```

C:\Windows\System32\cmd.exe
[WARNING] File encoding has not been set, using platform encoding Cp1252, i.e. build is platform dependent!
[INFO] Compiling 11 source files to C:\Users\user\Desktop\New folder (3)\target\classes
[WARNING] /C:/Users/user/Desktop/New folder (3)/src/main/java/com/ticketing/payment/service/PaymentService.java: C:\Users\
s\user\Desktop\New folder (3)\src/main/java/com/ticketing/payment/service/PaymentService.java uses unchecked or unsafe o
perations.
[WARNING] /C:/Users/user/Desktop/New folder (3)/src/main/java/com/ticketing/payment/service/PaymentService.java: Recompi
le with -Xlint:unchecked for details.
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ payment-service ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ payment-service ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding Cp1252, i.e. build is platform dependent!
[INFO] Compiling 1 source file to C:\Users\user\Desktop\New folder (3)\target\test-classes
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ payment-service ---
[INFO] Tests are skipped.
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ payment-service ---
[INFO] Building jar: C:\Users\user\Desktop\New folder (3)\target\payment-service-1.0.0.jar
[INFO] BUILD SUCCESS
[INFO] Total time: 01:20 min
[INFO] Finished at: 2025-11-02T21:32:11+05:30
[INFO]
C:\Users\user\Desktop\New folder (3)>

```

```

C:\Windows\System32\cmd.exe
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ payment-service ---
[INFO] Building jar: C:\Users\user\Desktop\New folder (3)\target\payment-service-1.0.0.jar
[INFO] BUILD SUCCESS
[INFO] Total time: 01:06 min
[INFO] Finished at: 2025-11-02T21:53:30+05:30
[INFO]

C:\Users\user\Desktop\New folder (3)>dir target
Volume in drive C has no label.
Volume Serial Number is F830-1582

Directory of C:\Users\user\Desktop\New folder (3)\target

11/02/2025  09:53 PM    <DIR>        .
11/02/2025  09:53 PM    <DIR>        ..
11/02/2025  09:53 PM    <DIR>        classes
11/02/2025  09:52 PM    <DIR>        generated-sources
11/02/2025  09:53 PM    <DIR>        generated-test-sources
11/02/2025  09:53 PM    <DIR>        maven-archiver
11/02/2025  09:52 PM    <DIR>        maven-status
11/02/2025  09:53 PM             14,686 payment-service-1.0.0.jar
11/02/2025  09:53 PM    <DIR>        test-classes
                1 File(s)          14,686 bytes
                8 Dir(s)  45,692,452,864 bytes free

C:\Users\user\Desktop\New folder (3)>jar tf target/payment-service-1.0.0.jar | find "MANIFEST.MF"
META-INF/MANIFEST.MF

C:\Users\user\Desktop\New folder (3)>docker build -t payment-service:local .
[+] Building 17.5s (8/8) FINISHED
=> [internal] load build definition from Dockerfile                                docker:desktop-linux 1.4s
=> => transferring dockerfile: 464B                                              0.6s
=> [internal] load metadata for docker.io/library/eclipse-temurin:17-jdk-alpine 4.4s
=> [internal] load .dockerignore                                                  0.3s
=> => transferring context: 28                                                    0.0s
=> [1/3] FROM docker.io/library/eclipse-temurin:17-jdk-alpine@sha256:eb42bc053cbff0d750d76fa0705b6faec2677131a1358d0bafcc844051b8872c 0.8s
=> => resolve docker.io/library/eclipse-temurin:17-jdk-alpine@sha256:eb42bc053cbff0d750d76fa0705b6faec2677131a1358d0bafcc844051b8872c 0.7s
=> [internal] load build context                                                  0.5s
=> => transferring context: 14.78kB                                              0.0s
=> CACHED [2/3] WORKDIR /app                                                    0.0s
=> [3/3] COPY target/payment-service-1.0.0.jar app.jar                          1.3s

```

```
C:\Windows\System32\cmd.exe
-> => transferring dockerfile: 464B
-> [internal] load metadata for docker.io/library/eclipse-temurin:17-jdk-alpine
-> [internal] load .dockerignore
-> => transferring context: 2B
-> [1/3] FROM docker.io/library/eclipse-temurin:17-jdk-alpine@sha256:eb42bc053cbff0d750d76fa0705b6faec2677131a1358d0bafcc844051b8872c
-> => resolve docker.io/library/eclipse-temurin:17-jdk-alpine@sha256:eb42bc053cbff0d750d76fa0705b6faec2677131a1358d0bafcc844051b8872c
-> [internal] load build context
-> => transferring context: 14.78kB
-> CACHED [2/3] WORKDIR /app
-> [3/3] COPY target/payment-service-1.0.0.jar app.jar
-> exporting to image
-> => exporting layers
-> => exporting manifest sha256:562c66da05a28ef89e55df17a64a0f90dffdff65f5e242cb47950a6b160fda09
-> => exporting config sha256:79616b874463e3ce7067a81daf76534f438dffa887908bd0cc30caf9915b965d
-> => exporting attestation manifest sha256:556c7c89dae2a407e5e86b95df7b3d53d70c1847258c3255a576f32ad99f13be
-> => exporting manifest list sha256:4e109378581c7feb1fb0316c38b41cfaf83a756dad394d78478e87a003876fe8
-> => naming to docker.io/library/payment-service:local
-> => unpacking to docker.io/library/payment-service:local

C:\Users\user\Desktop\New folder (3)>docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
payment-service     local              4e109378581c       22 seconds ago     507MB
postgres            13                 9a41ba632f72       2 weeks ago        618MB

C:\Users\user\Desktop\New folder (3)>
```

```
C:\Windows\System32\cmd.exe - docker-compose up
a 17.0.16 with PID 1 (/app/app.jar started by root in /app)
payment-service | 2025-11-02T18:23:30.650Z INFO 1 --- [
payment-service | 2025-11-02T18:23:35.247Z INFO 1 --- [
LT mode.
payment-service | 2025-11-02T18:23:35.460Z INFO 1 --- [
Found 3 JPA repository interfaces.
payment-service | 2025-11-02T18:23:38.460Z INFO 1 --- [
payment-service | 2025-11-02T18:23:38.538Z INFO 1 --- [
payment-service | 2025-11-02T18:23:38.539Z INFO 1 --- [
payment-service | 2025-11-02T18:23:38.733Z INFO 1 --- [
payment-service | 2025-11-02T18:23:38.742Z INFO 1 --- [
d in 7667 ms
payment-service | 2025-11-02T18:23:39.718Z INFO 1 --- [
payment-service | 2025-11-02T18:23:41.408Z INFO 1 --- [
.PgConnection@96abc76
payment-service | 2025-11-02T18:23:41.417Z INFO 1 --- [
payment-service | 2025-11-02T18:23:41.838Z INFO 1 --- [
Fault]
payment-service | 2025-11-02T18:23:42.049Z INFO 1 --- [
payment-service | 2025-11-02T18:23:42.205Z INFO 1 --- [
payment-service | 2025-11-02T18:23:43.122Z INFO 1 --- [
rmer
payment-service | 2025-11-02T18:23:46.450Z INFO 1 --- [
e.transaction.jta.platform' to enable JTA platform integration)
payment-service | 2025-11-02T18:23:47.763Z WARN 1 --- [
payment-service | 2025-11-02T18:23:47.764Z WARN 1 --- [
on "idempotency_keys" does not exist, skipping
payment-service | 2025-11-02T18:23:47.899Z INFO 1 --- [
e unit 'default'
payment-service | 2025-11-02T18:23:49.937Z WARN 1 --- [
efore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
payment-service | 2025-11-02T18:23:53.279Z INFO 1 --- [
payment-service | 2025-11-02T18:23:53.711Z INFO 1 --- [
h ..
payment-service | 2025-11-02T18:23:53.770Z INFO 1 --- [
(process running for 38.68)
payment-service | 2025-11-02T18:24:25.579Z INFO 1 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherSe
rvlet'
payment-service | 2025-11-02T18:24:25.583Z INFO 1 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
payment-service | 2025-11-02T18:24:25.588Z INFO 1 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 5 ms

View in Docker Desktop View Config Enable Watch
```

## <http://localhost:8080/actuator/health>

```
localhost:8080/actuator/health

Pretty-print

{"status":"UP","components":{"db":{"status":"UP","details":{"database":"PostgreSQL","validationQuery":"isValid()"},"diskSpace":{"status":"UP","details":{"total":1081101176832,"free":1023381925888,"threshold":10485760,"path":"/app/.","exists":true},"ping":{"status":"UP"}}}}
```

## <http://localhost:8080/actuator/prometheus>

```
Payment service implem x localhost:8080/actuator x localhost:8080/actuator x Google Gemini x localhost:8080/actuator x +
localhost:8080/actuator/prometheus

# HELP disk_total_bytes Total space for path
# TYPE disk_total_bytes gauge
disk_total_bytes(path="C:\\Users\\user\\Desktop\\New folder (3)\\.\\",) 2.14958075904E11
# HELP hikaricp_connections_idle Idle connections
# TYPE hikaricp_connections_idle gauge
hikaricp_connections_idle(pool="HikariPool-1",) 10.0
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count(error="none",exception="none",method="GET",outcome="SUCCESS",status="200",uri="/actuator/health",) 1.0
http_server_requests_seconds_sum(error="none",exception="none",method="GET",outcome="SUCCESS",status="200",uri="/actuator/health",) 0.3852173
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max(error="none",exception="none",method="GET",outcome="SUCCESS",status="200",uri="/actuator/health",) 0.3852173
# HELP jdbc_connections_idle Number of established but idle connections.
# TYPE jdbc_connections_idle gauge
jdbc_connections_idle(name="dataSource",) 10.0
# HELP tomcat_sessions_created_sessions_total
# TYPE tomcat_sessions_created_sessions_total counter
tomcat_sessions_created_sessions_total 0.0
# HELP hikaricp_connections_creation_seconds_max Connection creation time
# TYPE hikaricp_connections_creation_seconds_max gauge
hikaricp_connections_creation_seconds_max(pool="HikariPool-1",) 0.0
# HELP hikaricp_connections_creation_seconds Connection creation time
# TYPE hikaricp_connections_creation_seconds summary
hikaricp_connections_creation_seconds_count(pool="HikariPool-1",) 0.0
hikaricp_connections_creation_seconds_sum(pool="HikariPool-1",) 0.0
# HELP tomcat_sessions_rejected_sessions_total
# TYPE tomcat_sessions_rejected_sessions_total counter
tomcat_sessions_rejected_sessions_total 0.0
# HELP process_cpu_usage The "recent cpu usage" for the Java Virtual Machine process
# TYPE process_cpu_usage gauge
process_cpu_usage 0.3398497483480584
# HELP jvm_memory_committed_bytes The amount of memory in bytes that is committed for the Java virtual machine to use
# TYPE jvm_memory_committed_bytes gauge
jvm_memory_committed_bytes(area="nonheap",id="CodeHeap 'profiled nmethods'",) 1.1534336E7
jvm_memory_committed_bytes(area="heap",id="G1 Survivor Space",) 2097152.0
```

## [Docker](#)

Ask Gordon BETA

Containers

Images

Volumes

Kubernetes

Builds

Models

MCP Toolkit BETA

Docker Hub

Docker Scout

Extensions

Containers [Give feedback](#)

Container CPU usage ⓘ

0.25% / 400% (4 CPUs available)

Container memory usage ⓘ

275.39MB / 3.7GB

Show charts

Search

Only show running containers

	Name	Container ID	Image	Port(s)	CPU (%)	Last state	Actions
<input type="checkbox"/>	newfolder3	-	-	-	0.26%	8 m	<div><div></div><div></div><div></div></div>
<input type="checkbox"/>	postgres-db	6cbf831b303c	postgres:1	5432:5432	0%	8 m	<div><div></div><div></div><div></div></div>

Walkthroughs

Multi-container applications

8 mins

Containerize your application

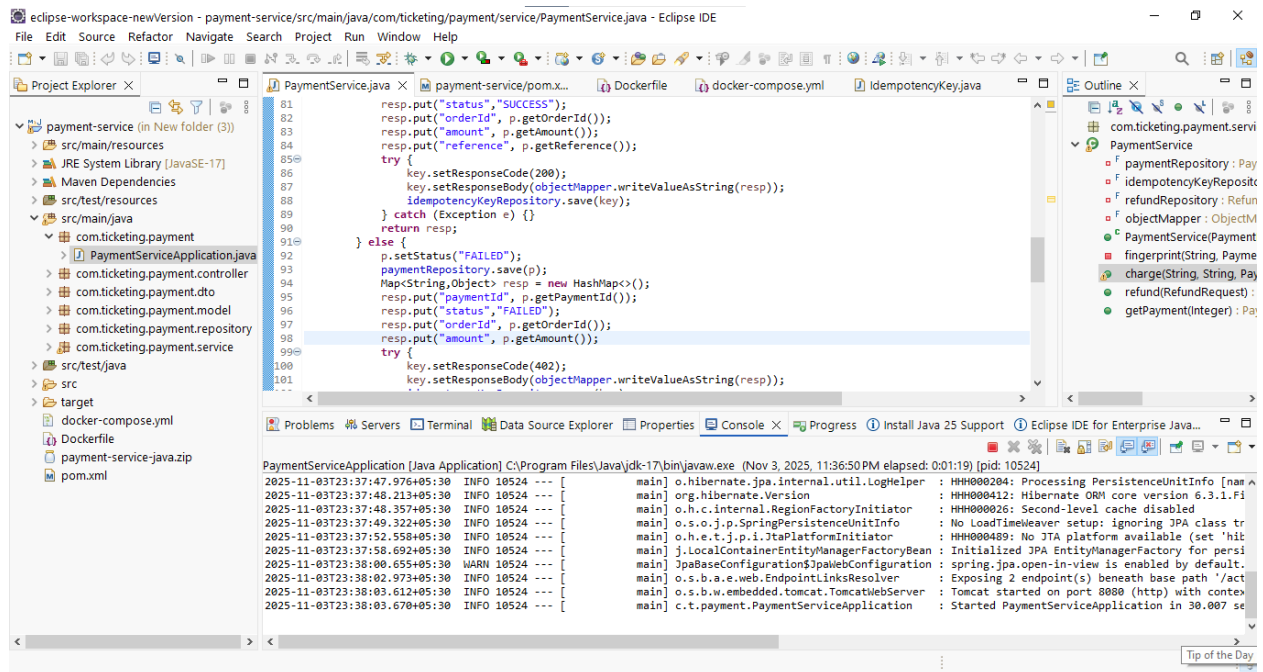
3 mins

[View more in the Learning center](#)

Engine running

RAM 1.43 GB CPU 1.51% Disk: 2.54 GB used (limit 1006.85 GB)

v4.48.0



## Sample Request and Response -

### POST – Charge Payment

Request:

```
{
  "orderId": "ORD1001",
  "amount": 500.0,
  "currency": "INR",
  "method": "CARD"
}
```

Headers:

Idempotency-Key: charge-101

Response (Success):

```
{
  "paymentId": 1,
  "status": "SUCCESS",
  "orderId": "ORD1001",
  "amount": 500.0,
  "reference": "ETP-a1b2c3d4"
}
```

Response (Duplicate Key):

```
{
  "error": "Idempotency conflict"
}
```

## POST – Refund Payment

Request:

```
{
  "paymentId": 1,
  "amount": 500.0
}
```

Response:

```
{
  "refundId": 1,
  "status": "SUCCESS",
  "paymentId": 1,
  "amount": 500.0
}
```

## GET – Retrieve Payment

Request:

GET /v1/payments/1

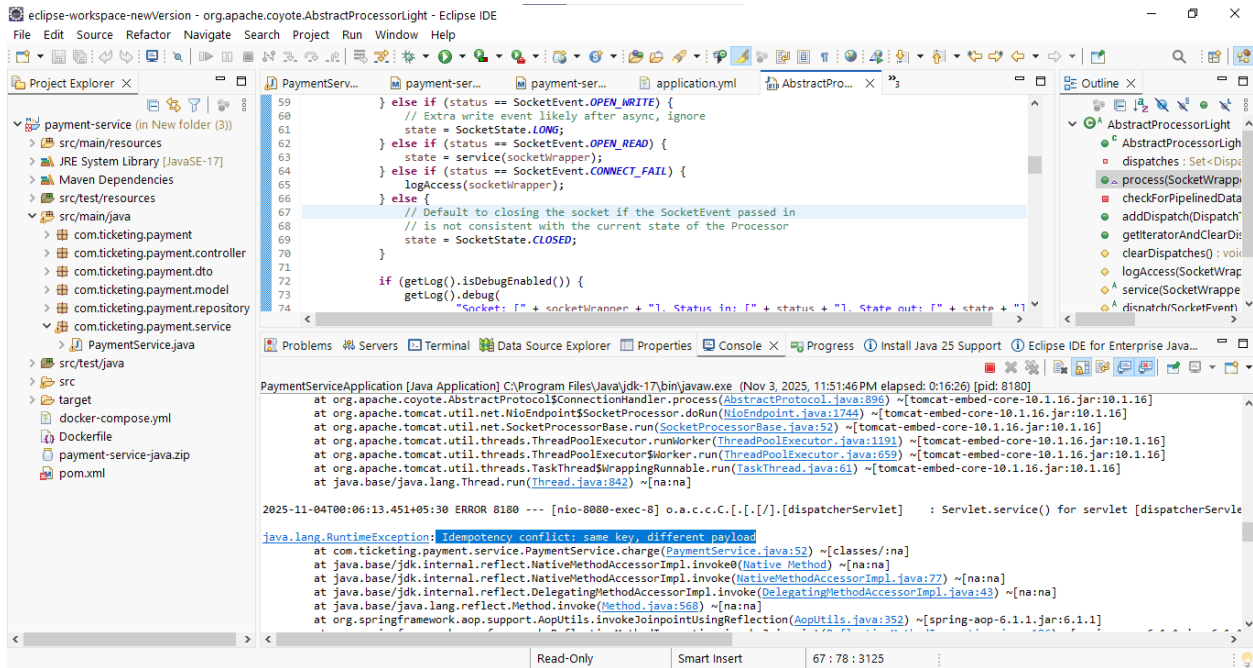
Response:

```
{
  "paymentId": 1,
  "orderId": "ORD1001",
  "amount": 500.0,
  "currency": "INR",
  "status": "SUCCESS",
  "method": "CARD",
  "reference": "ETP-a1b2c3d4"
}
```

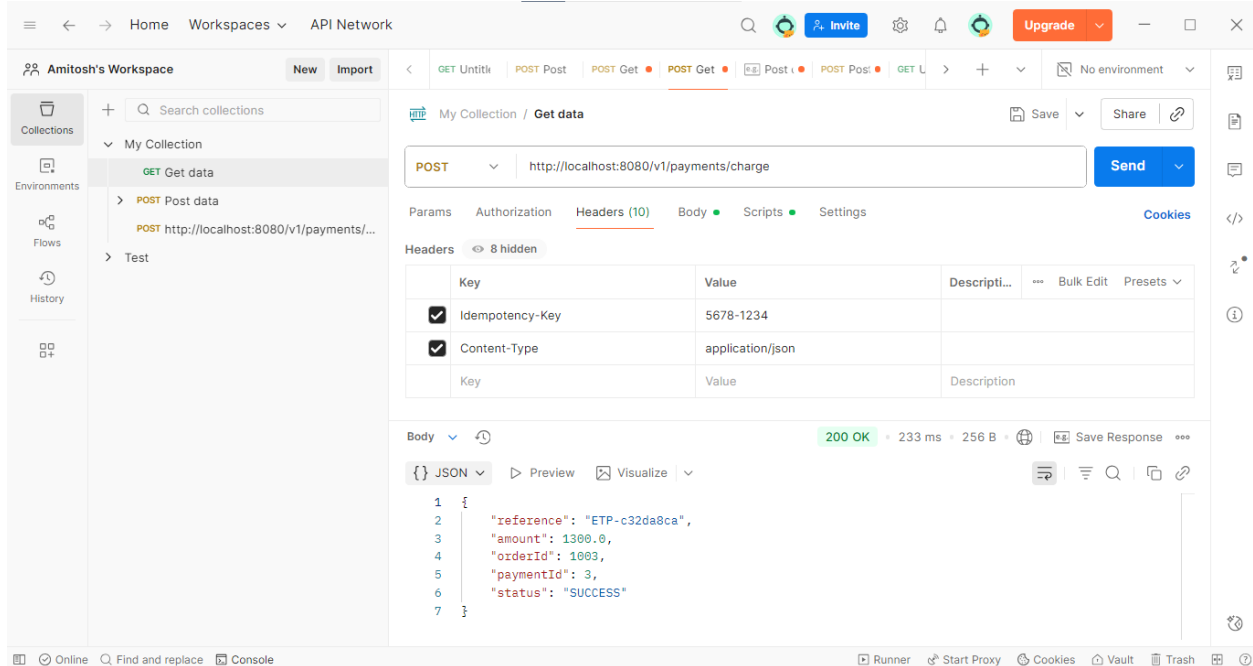
## Idempotency-

The screenshot shows the Postman API client interface. The request is a POST to `http://localhost:8080/v1/payments/charge`. The headers tab is selected, showing two headers: `Idempotency-Key` with value `5678-1234` and `Content-Type` with value `application/json`. The response is a `500 Internal Server Error` with a status of 500, 34 ms, and 273 B. The response body is JSON, showing an error object with the following fields:

```
{
  "timestamp": "2025-11-03T18:55:42.813+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "path": "/v1/payments/charge"
}
```



**<http://localhost:8080/v1/payments/charge>** -





API Network interface showing a POST request to `http://localhost:8080/v1/payments/charge` with a JSON body:

```
1 {
2   "orderId": 1003,
3   "amount": 1300,
4   "currency": "INR",
5   "method": "CARD"
6 }
```

The response is a 200 OK status with a JSON body:

```
1 {
2   "reference": "ETP-c32da8ca",
3   "amount": 1300.0,
4   "orderId": 1003,
5   "paymentId": 3,
6   "status": "SUCCESS"
7 }
```

### **Retry and same result –**

API Network interface showing the same POST request and response as above, confirming the result is consistent after a retry.

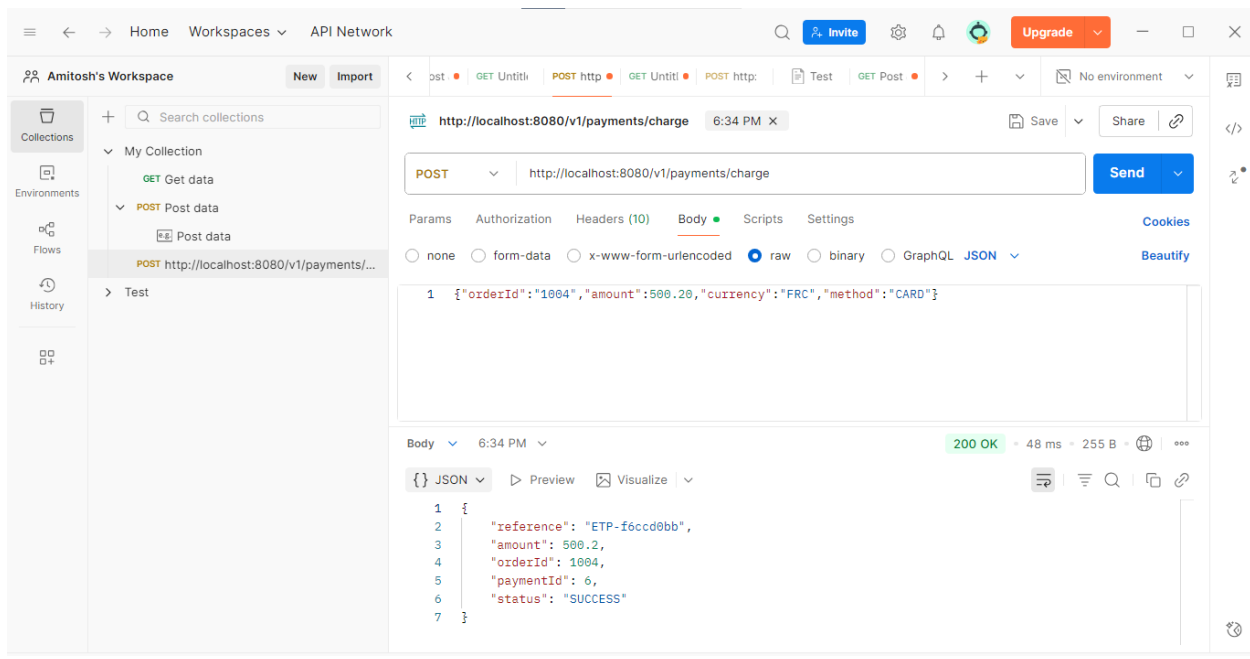
## Failed scenario when amount negative –

The screenshot shows the Postman interface with a POST request to `http://localhost:8080/v1/payments/charge`. The request body is a JSON object: `{ "orderId": "1004", "amount": -500.0, "currency": "INR", "method": "CARD" }`. The response status is `200 OK`, and the response body is a JSON object: `{ "amount": -500.0, "orderId": 1004, "paymentId": 5, "error": "Invalid payment amount. Must be greater than zero.", "status": "FAILED" }`.

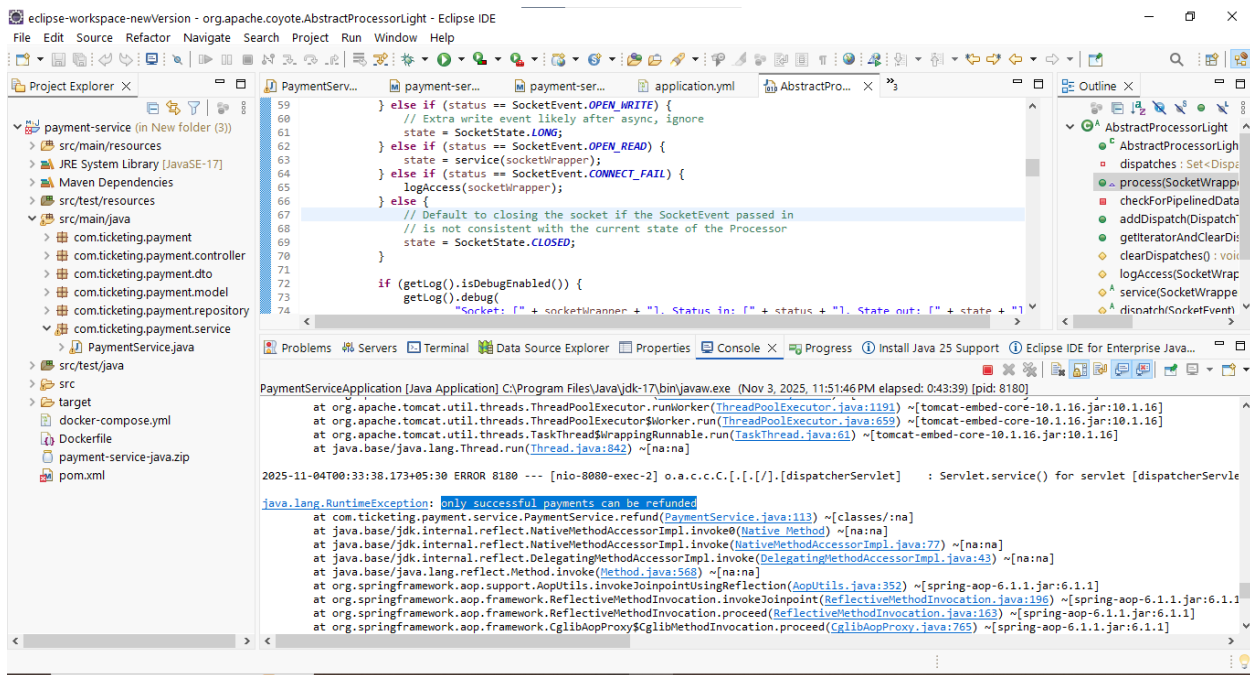
```
Pager usage is off.
paymentsdb=# SELECT * FROM payments;
 payment_id | amount | created_at | method | order_id | reference | status
-----+-----+-----+-----+-----+-----+-----
          1 |   1200 |             | CARD   |    1001 | ETP-e2ab4afb | REFUNDED
          2 |   1201 |             | CARD   |    1002 |             | FAILED
          4 |   1301 |             | CARD   |    1003 |             | FAILED
          3 |   1300 |             | CARD   |    1003 | ETP-c32da8ca | REFUNDED
          5 |   -500 |             | CARD   |    1004 |             | FAILED
(5 rows)
```

## Decimal values should accept –

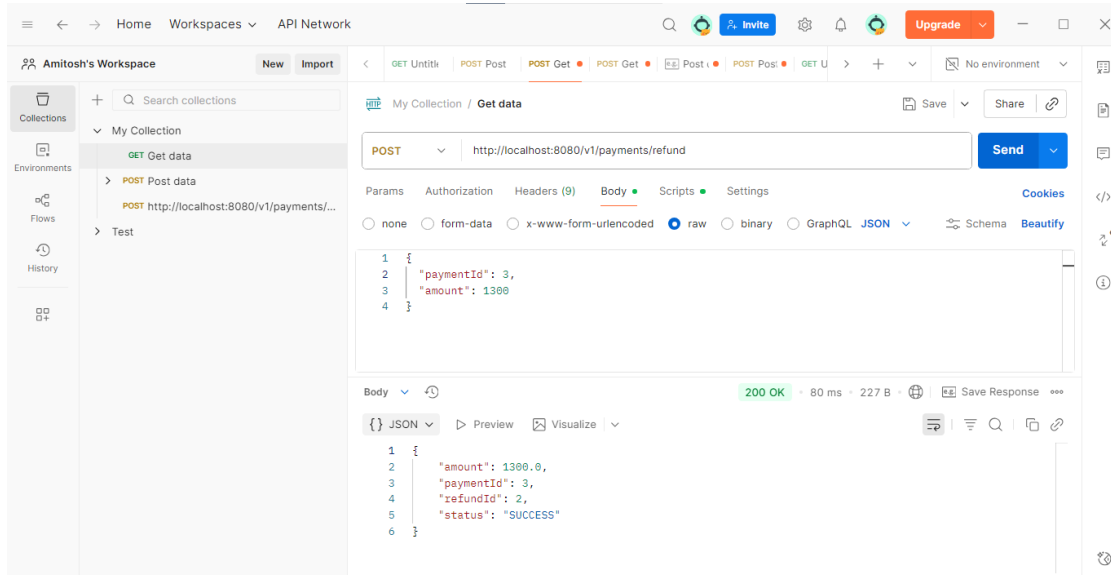
```
paymentsdb=# SELECT * FROM payments;
 payment_id | amount | created_at | method | order_id | reference | status
-----+-----+-----+-----+-----+-----+-----
          1 |   1200 |             | CARD   |    1001 | ETP-e2ab4afb | REFUNDED
          2 |   1201 |             | CARD   |    1002 |             | FAILED
          4 |   1301 |             | CARD   |    1003 |             | FAILED
          3 |   1300 |             | CARD   |    1003 | ETP-c32da8ca | REFUNDED
          5 |   -500 |             | CARD   |    1004 |             | FAILED
          6 |  500.2 |             | CARD   |    1004 | ETP-f6ccd0bb | SUCCESS
(6 rows)
```



## **Failed status should not refund –**



## **Refund - http://localhost:8080/v1/payments/refund -**



```
Command Prompt - psql -U postgres -d paymentsdb

1 | 1200 | | 1 | REF-7c4c7f4b | SUCCESS
(1 row)

paymentsdb=# SELECT * FROM payments;
 payment_id | amount | created_at | method | order_id | reference | status 
-----
1 | 1200 | | CARD | 1001 | ETP-e2ab4afb | REFUNDED
2 | 1201 | | CARD | 1002 | | FAILED
3 | 1300 | | CARD | 1003 | ETP-c32da8ca | SUCCESS
4 | 1301 | | CARD | 1003 | | FAILED
(4 rows)

paymentsdb=# SELECT * FROM refunds;
 id | amount | created_at | payment_id | provider_ref | status 
-----
1 | 1200 | | 1 | REF-7c4c7f4b | SUCCESS
(1 row)

paymentsdb=# SELECT * FROM refunds;
 id | amount | created_at | payment_id | provider_ref | status 
-----
1 | 1200 | | 1 | REF-7c4c7f4b | SUCCESS
2 | 1300 | | 3 | REF-75b28e2b | SUCCESS
(2 rows)

paymentsdb=#
```

## **Refund more than available amount -**

```
Problems Servers Terminal Data Source Explorer Properties Console X Progress Install Java 25 Support

PaymentServiceApplication [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (Nov 8, 2025, 6:22:33 PM elapsed: 0:17:08) [pid: 8988]
T18:30:52.056+05:30 INFO 8988 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet
T18:30:52.074+05:30 INFO 8988 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization
T18:30:52.323+05:30 WARN 8988 --- [nio-8080-exec-1] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.servlet.NoHandlerFoundException: No handler found for [/payments/refund] with method [POST]
T18:32:06.979+05:30 WARN 8988 --- [nio-8080-exec-4] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.servlet.NoHandlerFoundException: No handler found for [/payments/refund] with method [POST]
T18:39:28.483+05:30 ERROR 8988 --- [nio-8080-exec-6] o.a.c.c.C.[.[/].[dispatcherServlet] : Servlet.service()

RuntimeException: Refund amount cannot exceed original payment amount
: com.ticketing.payment.service.PaymentService.refund(PaymentService.java:145) ~[classes/:na]
: java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method) ~[na:na]
: java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77) ~[na:na]
```

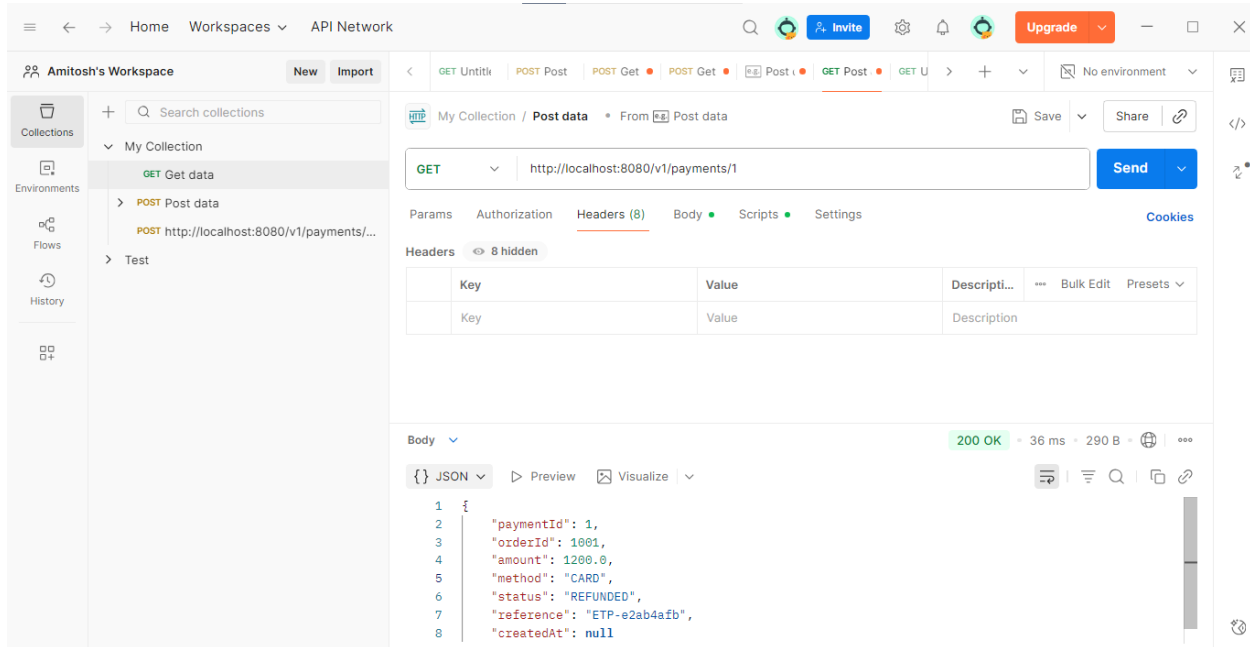
API Network interface showing a POST request to `http://localhost:8080/v1/payments/refund` with a JSON body:

```
{  "paymentId": 6,  "amount": 1300}
```

The response is a **500 Internal Server Error** (100 ms, 273 B) with the following JSON body:

```
{  "timestamp": "2025-11-08T13:09:28.543+00:00",  "status": 500,  "error": "Internal Server Error",  "path": "/v1/payments/refund"}
```

## Payment ID – 1 ---



## Testing and Validation -

Testing was performed using Postman and Swagger UI.

Verified endpoints:

- Charge request with and without Idempotency-Key.
- Refund of successful transactions.
- Retrieval of payment details.
- Health and Prometheus monitoring endpoints.

Confirmed:

- Database persistence in PostgreSQL.
- Accurate status transitions (PENDING → SUCCESS/FAILED → REFUNDED).
- Consistent JSON responses.
- Duplicate prevention via Idempotency-Key logic.

## Database – PostgreSQL -

Tables

- **payments:** Stores charge details and transaction statuses.
- **refunds:** Tracks refund operations.
- **idempotency\_keys:** Manages request fingerprints and cached responses.

SELECT \* FROM payments;

```
Command Prompt - psql -U postgres -d paymentsdb

DROP TABLE
paymentsdb=# CREATE TABLE idempotency_keys (
paymentsdb=#   id SERIAL PRIMARY KEY,
paymentsdb=#   idempotency_key VARCHAR(255) UNIQUE,
paymentsdb=#   request_fingerprint TEXT,
paymentsdb=#   response_body JSONB,
paymentsdb=#   response_code INTEGER
paymentsdb=# );
CREATE TABLE
paymentsdb=# SELECT * FROM payments;
 payment_id | amount | created_at | method | order_id | reference | status
-----
1 | 1200 | | CARD | 1001 | ETP-e2ab4afb | REFUNDED
2 | 1201 | | CARD | 1002 | | FAILED
(2 rows)

paymentsdb=# ALTER TABLE idempotency_keys
paymentsdb=# ALTER COLUMN response_body TYPE text USING response_body::text;
ALTER TABLE
paymentsdb=# SELECT * FROM payments;
 payment_id | amount | created_at | method | order_id | reference | status
-----
1 | 1200 | | CARD | 1001 | ETP-e2ab4afb | REFUNDED
2 | 1201 | | CARD | 1002 | | FAILED
3 | 1300 | | CARD | 1003 | ETP-c32da8ca | SUCCESS
(3 rows)

paymentsdb=#
```

SELECT \* FROM idempotency\_keys;

```
Command Prompt - psql -U postgres -d paymentsdb

paymentsdb=# SELECT * FROM idempotency_keys;
 id | idempotency_key | request_fingerprint | response_code |
-----
1 | 5678-1234 | c46708cdcd28d36f751645caed72a7e079aded1ba37e3d987dbd66ee89b47f94 | {"reference":"ETP-c32da8ca",
amount":1300.0,"orderId":1003,"paymentId":3,"status":"SUCCESS"} | 200
2 | 5678-5678 | c4ce304c199d2d56f0ce3d26c1d088d2498c0ceb3c3d782836a3e5d314dd8b5b | {"amount":1301.0,"orderId":10
03,"paymentId":4,"status":"FAILED"} | 402
(2 rows)

paymentsdb=#
```

SELECT \* FROM refunds;

```
Command Prompt - psql -U postgres -d paymentsdb
1 | 1200 | | 1 | REF-7c4c7f4b | SUCCESS
(1 row)

paymentsdb=# SELECT * FROM payments;
 payment_id | amount | created_at | method | order_id | reference | status
-----
1 | 1200 | | CARD | 1001 | ETP-e2ab4afb | REFUNDED
2 | 1201 | | CARD | 1002 | | FAILED
3 | 1300 | | CARD | 1003 | ETP-c32da8ca | SUCCESS
4 | 1301 | | CARD | 1003 | | FAILED
(4 rows)

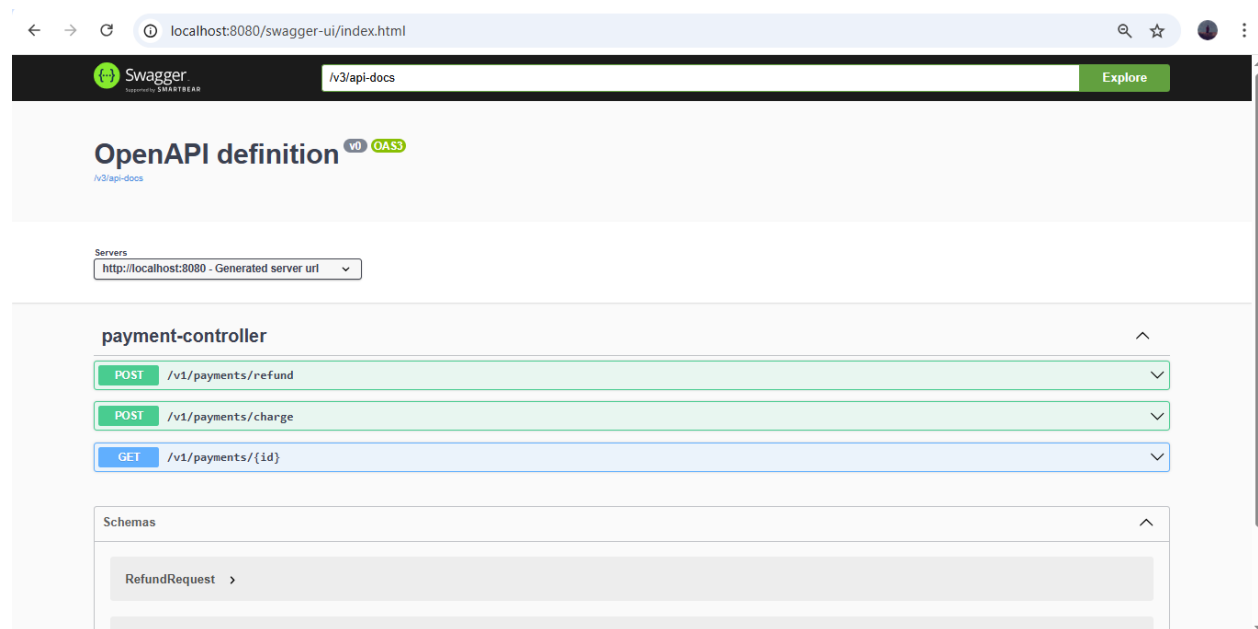
paymentsdb=# SELECT * FROM refunds;
 id | amount | created_at | payment_id | provider_ref | status
-----
1 | 1200 | | 1 | REF-7c4c7f4b | SUCCESS
(1 row)

paymentsdb=# SELECT * FROM refunds;
 id | amount | created_at | payment_id | provider_ref | status
-----
1 | 1200 | | 1 | REF-7c4c7f4b | SUCCESS
2 | 1300 | | 3 | REF-75b28e2b | SUCCESS
(2 rows)

paymentsdb=#
```

## MiniKube –

```
C:\Users\user\Desktop\New folder (3)>kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
payment-service-7fdcf58576-k2xps    1/1     Running   2 (34s ago) 6m32s
postgres-84f4c5b4fc-kgs4s          1/1     Running   0           6m30s
```





payment-controller

POST /v1/payments/refund

POST /v1/payments/charge

GET /v1/payments/{id}

Parameters

Name Description

id \* required integer(int32) (path) 1

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  "http://localhost:8080/v1/payments/1" \
  -H 'accept: */*'

```

Request URL

http://localhost:8080/v1/payments/1

Server response

Code Details

docker desktop PERSONAL

Search Ctrl+K Sign in

Containers Give feedback

Container CPU usage 995.28% / 400% (4 CPUs available)

Container memory usage 346.3MB / 3.7GB

Show charts

Search

Only show running containers

Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
minikube	ad1e1bdacbe3	k8s-miniku	49577:22 <a href="#">Show all ports (5)</a>	0%	1 hour ago	
newfolder3	-	-	-	0%	19 seconds ago	

Walkthroughs

Multi-container applications 8 mins

Containerize your application 3 mins

View more in the Learning center

Engine running

RAM 2.75 GB CPU 94.59% Disk: 6.01 GB used (limit 1006.85 GB)

v4.48.0

## Swagger UI –

localhost:8080/swagger-ui/index.html#/payment-controller/refund

### OpenAPI definition v0 OAS3

[/v3/api-docs](#)

Servers

http://localhost:8080 - Generated server url

#### payment-controller

**POST** /v1/payments/refund

Parameters Try it out

No parameters

Request body required application/json

Example Value | Schema

```
{
  "paymentId": 0,
  "amount": 0
}
```

localhost:8080/swagger-ui/index.html#/payment-controller/charge

**POST** /v1/payments/charge

Parameters Try it out

Name	Description
<b>Idempotency-Key</b> <span>required</span> string (header)	<input type="text" value="Idempotency-Key"/>
<b>X-Correlation-Id</b> string (header)	<input type="text" value="X-Correlation-Id"/>

Request body required application/json

Example Value | Schema

```
{
  "orderId": 0,
  "amount": 0,
  "currency": "string",
  "method": "string"
}
```

localhost:8080/swagger-ui/index.html#/payment-controller/getPayment

GET /v1/payments/{id}

Parameters

Name	Description
id • required	
integer(int32)	1
(path)	

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8080/v1/payments/1' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:8080/v1/payments/1
```

Server response

Code	Details
200	<p>Response body</p> <pre>{   "paymentId": 1,   "orderId": 1001,   "amount": 1200,   "method": "CARD",   "status": "REFUNDED",   "reference": "ETP-e2ab4a5b",   "createdAt": null }</pre>

Download

localhost:8080/swagger-ui/index.html#/payment-controller/getPayment

```
{
  "paymentId": 0,
  "orderId": 0,
  "amount": 0,
  "method": "string",
  "status": "string",
  "reference": "string",
  "createdAt": "2025-11-04T20:57:37.435Z"
}
```

Schemas

RefundRequest ▾

```
{
  paymentId integer(int32)
  amount number($double)
}
```

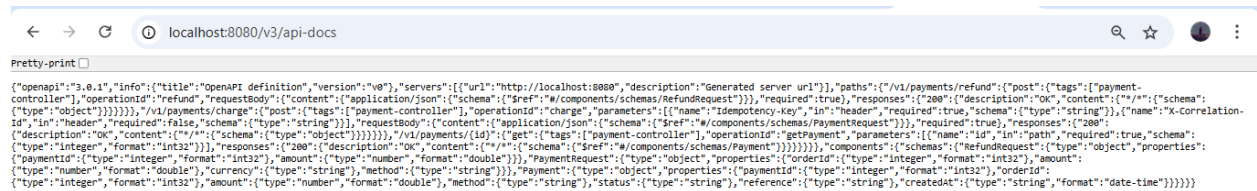
PaymentRequest ▾

```
{
  orderId integer(int32)
  amount number($double)
  currency string
  method string
}
```

Payment ▾

```
{
  paymentId integer(int32)
  orderId integer(int32)
  amount number($double)
  method string
  status string
  reference string
  createdAt string($date-time)
}
```

## API docs -



### **Key Learnings –**

- Gained hands-on experience with Spring Boot 3, PostgreSQL, and Docker.
- Implemented idempotency for safe and reliable payment operations.
- Understood microservice design with clear service boundaries.
- Achieved successful deployment on Docker and Kubernetes.
- Strengthened understanding of RESTful APIs, error handling, and monitoring.

### Conclusion –

The Payment Service successfully fulfills its role in the Event Ticketing System by ensuring secure, consistent, and idempotent transaction management.

Its modular design, containerized deployment, and clear API structure make it a reliable and scalable component in the distributed architecture.