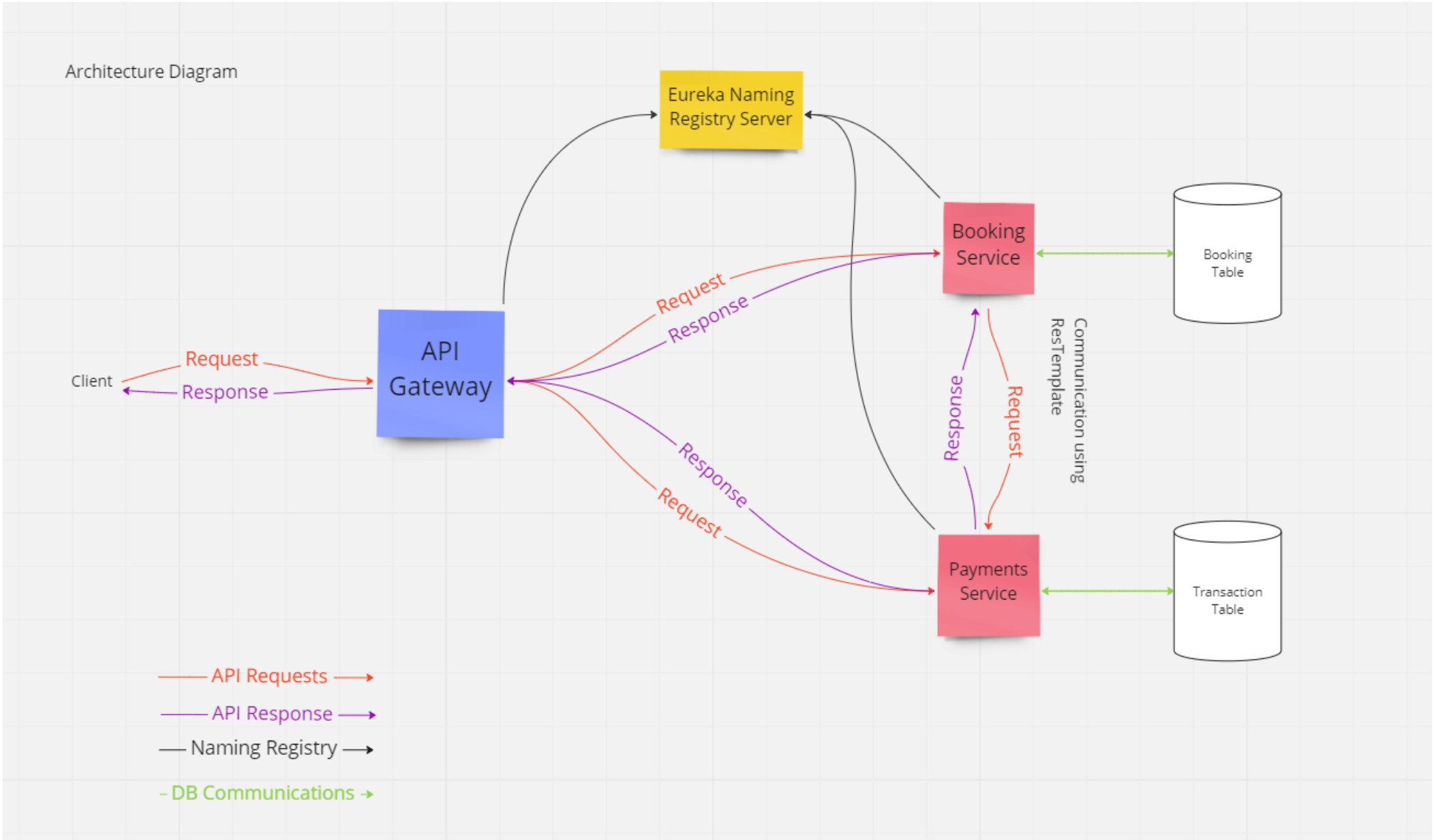
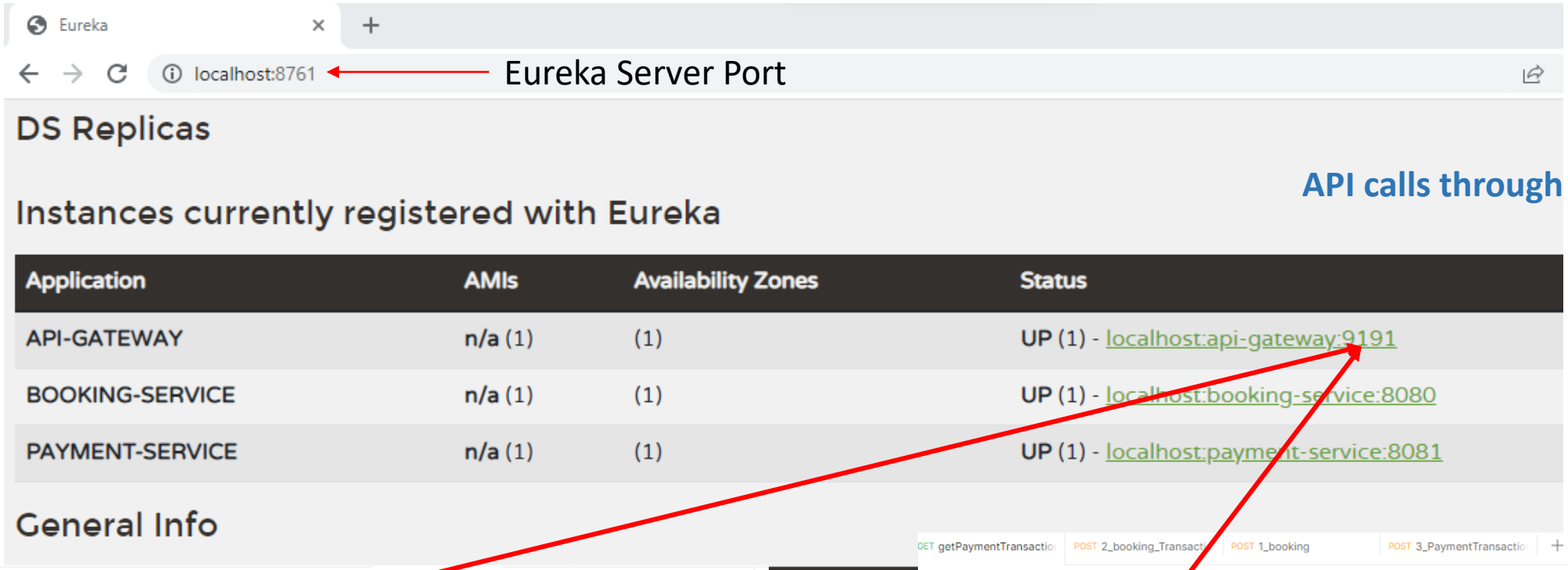
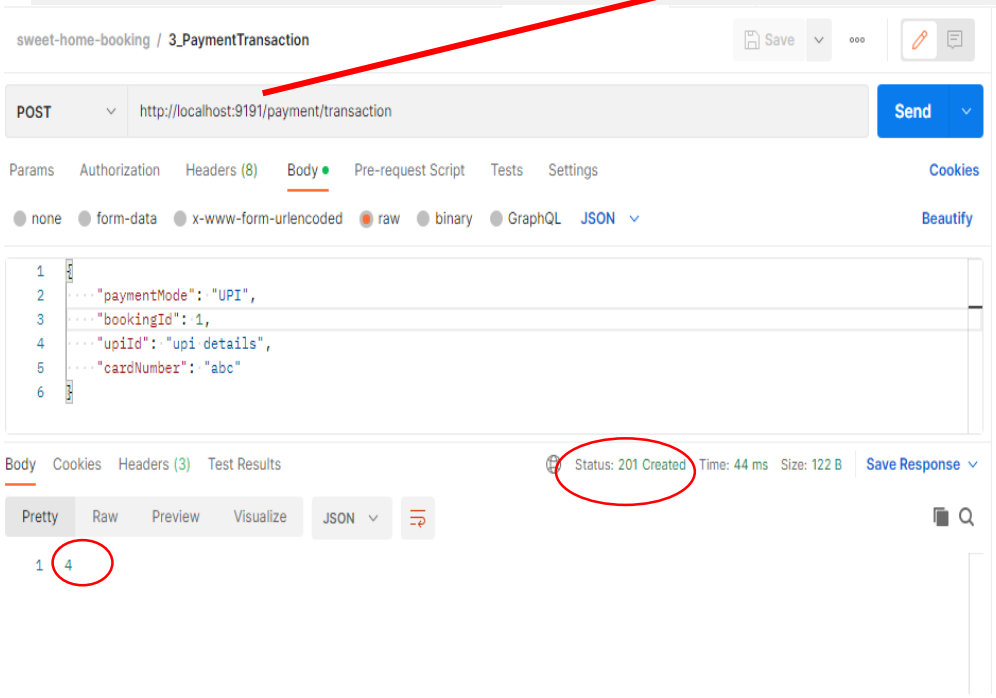
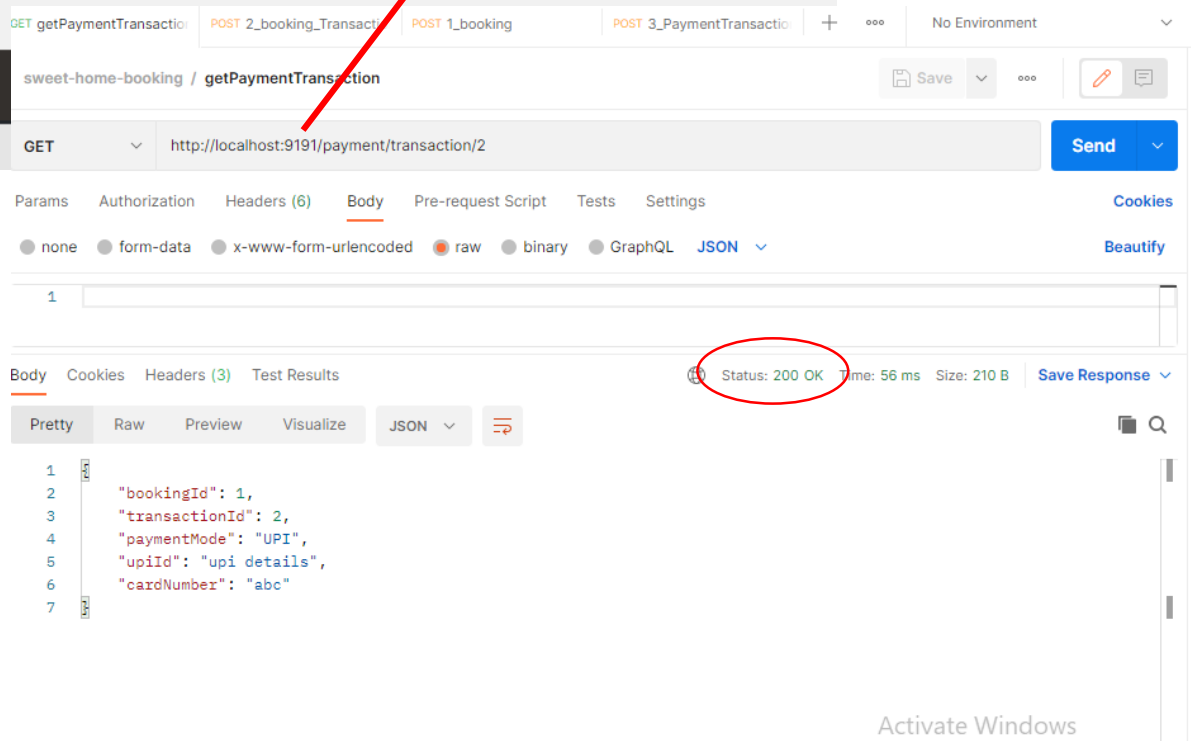

Architecture Diagram
Eureka Naming Registry Server
Booking Service
Booking Table
Client
Request
Response
API Gateway
Request
Response
Response
Request
Response
Request
Communication using ResTemplate
Payments Service
Transaction Table
API Requests
API Response
Naming Registry
DB Communications

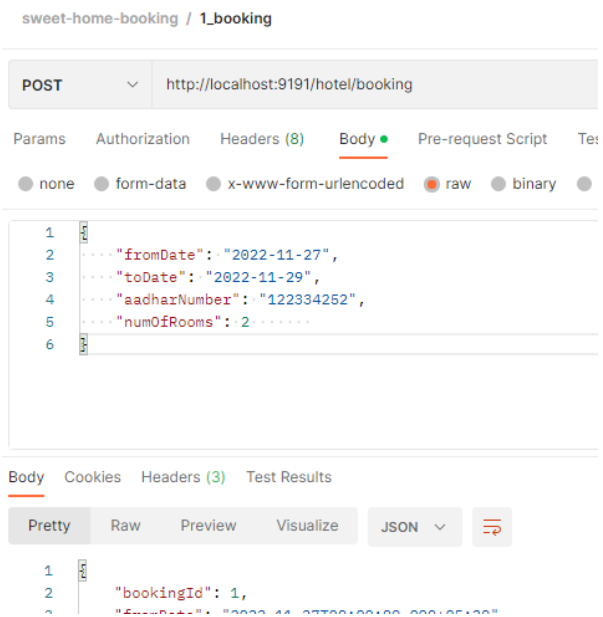


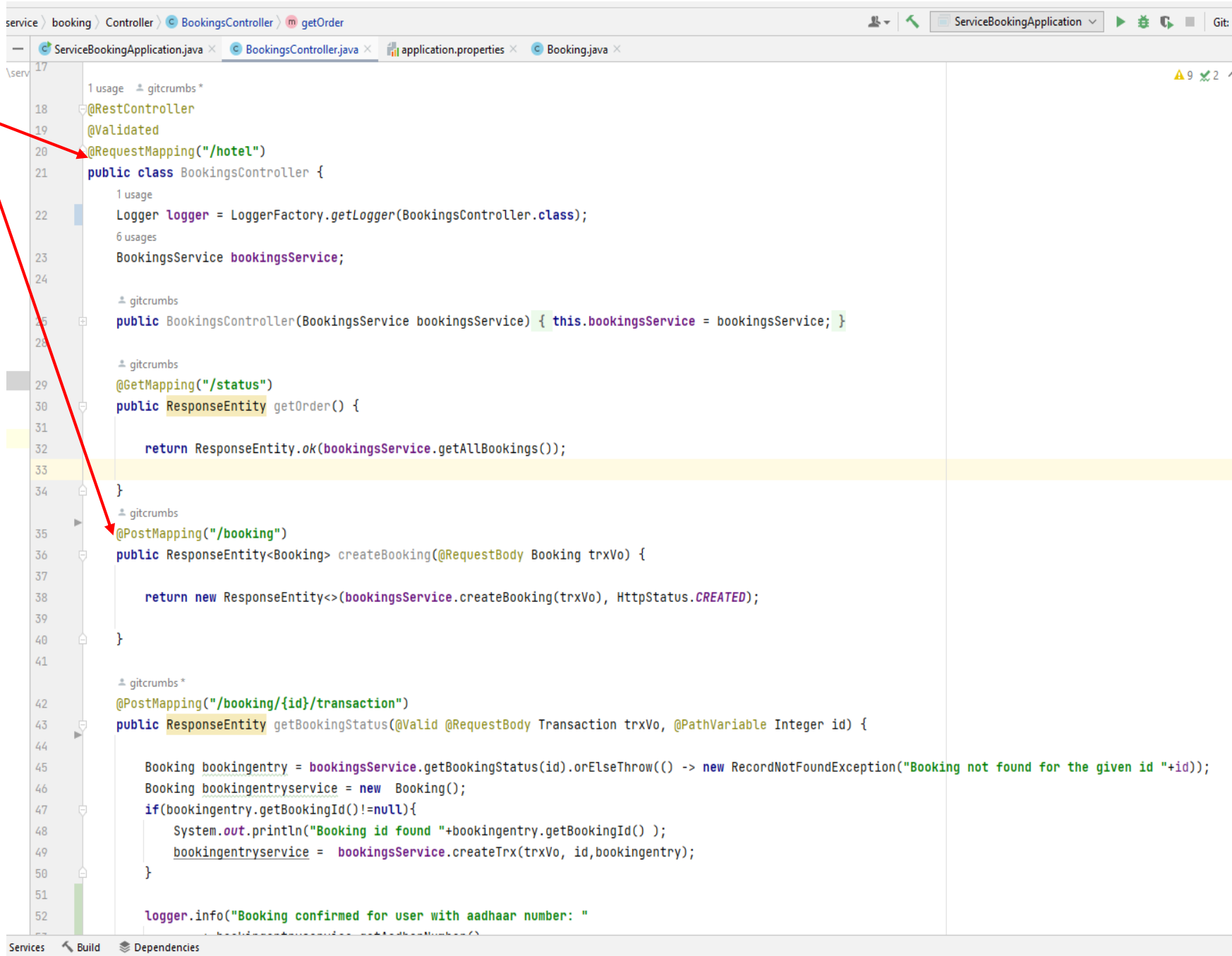
API calls through Gateway





http://localhost:9191/hotel/booking





Code Flow

Page 1

```java
}

/* The requested post call for booking brings the code flow here
* and  after formatting the data, the entry is saved in the booking repository*/
1 usage  gitcrumbs *
public Booking createBooking(Booking booking) throws Exception {
    Booking bookedItem =  bookingRepository.save(formatDataEntry(booking));
    return bookedItem;
}

1 usage  gitcrumbs
public Booking createTrx(Transaction trxVo, Integer id, Booking bookingentry){       //
    trxVo.setBookingId(id);

    Integer item= restTemplate.postForObject(transactionAppUrl, trxVo, Integer.class);
    System.out.println("Generated Transaction id is"+item);
    bookingentry.setTransactionId(item);

    bookingRepository.save(bookingentry);

    return bookingentry;
}
```

```java
/* This code block formats the data as per the given format in
 the problem statement*/
1 usage  new *
public Booking formatDataEntry(Booking booking)throws  Exception{

    DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd'T'HH:mm:ss.SSSXXX");
    int numOfRooms = booking.getNumOfRooms();
    String s= getRandomNumbers(numOfRooms).toString();
    String requiredString = s.substring(s.indexOf("[")+1 , s.indexOf("]"));
    booking.setRoomNumbers(requiredString);
    try{
        booking.setBookedOn(LocalDateTime.now());
        Calendar cl = Calendar. getInstance();
        Timestamp filterDateFromTs = null,filterDateToTs=null;
        SimpleDateFormat dateFormat = new SimpleDateFormat( pattern: "yyyy-MM-dd");
        Date firstDate = dateFormat.parse(booking.getFromDate());
        Date secondDate = dateFormat.parse(booking.getToDate());
        long diffInMillies = Math.abs(secondDate.getTime() - firstDate.getTime());
        long daysBetween = TimeUnit.DAYS.convert(diffInMillies, TimeUnit.MILLISECONDS);

        long roomPrice = 1000 * numOfRooms*(daysBetween);
        String timeStampFrom =new SimpleDateFormat( pattern: "yyyy-MM-dd'T'HH:mm:ss.SSSXXX").format((dateFormat.parse(booking.getFromDate())).getTime());
        String timeStampTo =new SimpleDateFormat( pattern: "yyyy-MM-dd'T'HH:mm:ss.SSSXXX").format((dateFormat.parse(booking.getToDate())).getTime());
        booking.setFromDate(timeStampFrom);
        booking.setToDate(timeStampTo);
        booking.setRoomPrice((int)roomPrice);
    }catch(Exception e){
        throw new Exception("Unable to create Transaction"+e.getMessage());
    }

    return booking;

}
}
```

Code Flow: Booking Service

Page 2

```java
gitcrumbs
public BookingsController(BookingsService bookingsService) { this.bookingsService = bookingsService; }

gitcrumbs
@GetMapping("/status")
public ResponseEntity getOrder() {

    return ResponseEntity.ok(bookingsService.getAllBookings());

}

/* The post mapping call with the booking endpoint passes the booking object as payload
 to the createBooking method defined in the BookingService layer and it in returns the
  generates the booking entry in the database .
 * */
gitcrumbs *
@PostMapping("/booking")
public ResponseEntity<Booking> createBooking(@RequestBody Booking trxVo) throws Exception {

    return new ResponseEntity<>(bookingsService.createBooking(trxVo), HttpStatus.CREATED);

}

/* The post mapping call with the booking id to the transaction service
```

Schema Maps

POST http://localhost:9191/hotel/booking

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
1
```

Body Cookies Headers (3) Test Results Status: 201 Created Time: 1878 ms Size: 361 B

Pretty Raw Preview Visualize JSON


```
1
2   "bookingId": 1,
3   "fromDate": "2022-11-27T00:00:00.000+05:30",
4   "toDate": "2022-11-29T00:00:00.000+05:30",
5   "aadharNumber": "122334252",
6   "numOfRooms": 2,
7   "roomNumbers": "35, 93",
8   "roomPrice": 4000,
9   "transactionId": 0,
10  "bookedOn": "2022-11-27T19:55:35.2906605"
11
```


Booking Service

```
3 usages
@NotBlank(message = "Invalid mode of payment")
private  String cardNumber;
```

```
service-booking D:\Upgrad_Projects\Sweet-home\serv
.idea
.mvn
src
main
java
com.example.service.booking
aspect
LoggingAspect
Controller
Dao
entities
exceptions
```

```
spring.application.name=booking-service
spring.jpa.hibernate.ddl-auto=create
spring.datasource.initialization-mode=always
spring.datasource.platform=postgres
spring.datasource.url=jdbc:postgresql://localhost:5432/myDB
spring.datasource.username=postgres
spring.datasource.password=postgres
spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=
spring.jpa.show-sql=true

transactionApp.url=http://localhost:8081/payment/transaction
```

```
@Value("${transactionApp.url}")
private String transactionAppUrl;
```

```
/* This method makes and internal call to the payments service,
* when the user requests the call on booking/{id}/transaction
* The call is made using the rest template and the parameters are
* injected from the constants file application.properties
* The url is injected with the @Value("${transactionApp.url}") annotation
* */
1 usage  gitcrumbs *
public Booking createTrx(Transaction trxVo, Integer id, Booking bookingentry){
    trxVo.setBookingId(id);

    Integer item= restTemplate.postForObject(transactionAppUrl, trxVo, Integer.class);

    bookingentry.setTransactionId(item);

    bookingRepository.save(bookingentry);

    return bookingentry;
```

sweet-home-booking / 2_booking_Transaction

POST http://localhost:9191/hotel/booking/1/transaction

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
{
    "paymentMode": "UPI",
    "bookingId": 1,
    "upiId": "upi details",
    "cardNumber": ""
}
```

Body Cookies Headers (3) Test Results Status: 400 Bad Request Time: 158 ms Size: 179 B

Pretty Raw Preview Visualize JSON

```
{
    "message": "Invalid mode of payment",
    "statusCode": 400
}
```

POST http://localhost:9191/hotel/booking/2/transaction

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

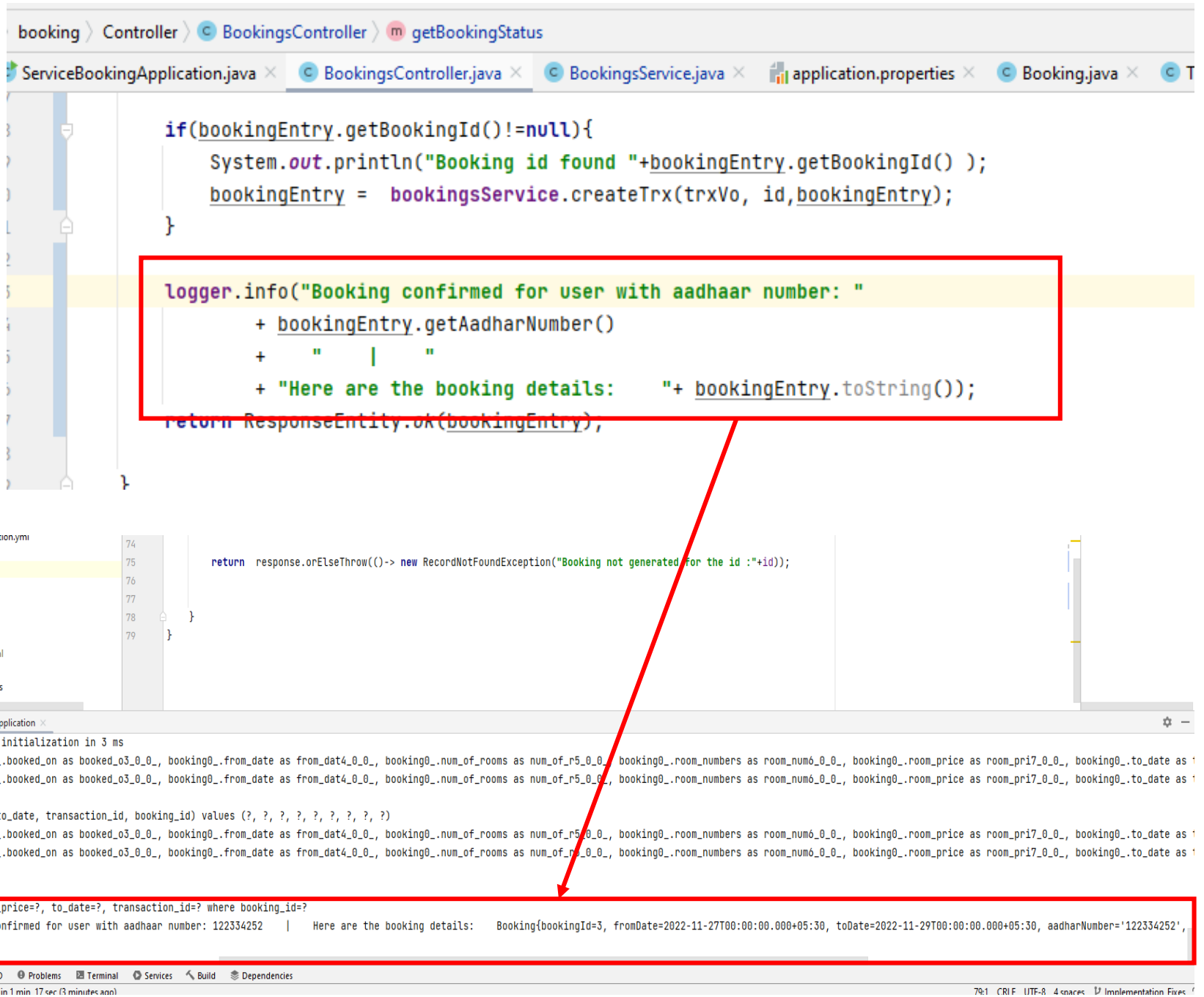
```
{
    "paymentMode": "UPI",
    "bookingId": 2,
```

Body Cookies Headers (3) Test Results Status: 200 OK Time: 590 ms Size: 355 B

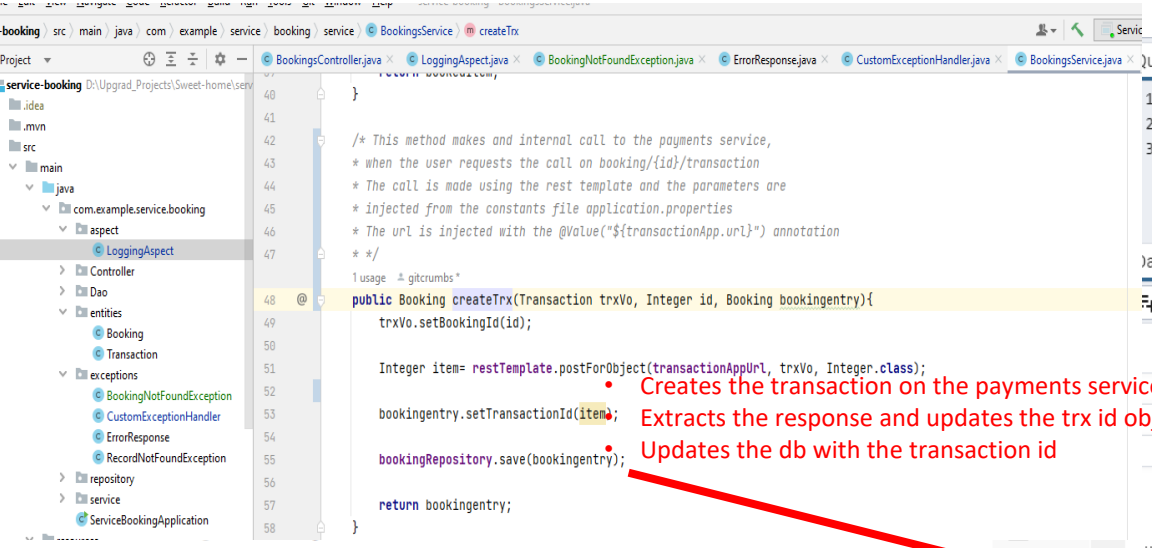
Pretty Raw Preview Visualize JSON

```
{
    "bookingId": 2,
    "fromDate": "2022-11-27T00:00:00.000+05:30",
    "toDate": "2022-11-29T00:00:00.000+05:30",
    "aadharNumber": "122334252",
    "numOfRooms": 2,
    "roomNumbers": "79, 74",
    "roomPrice": 4000,
    "transactionId": 3,
    "bookedOn": "2022-11-27T22:24:22.232142"
}
```

*Booking Service -> Payments Service

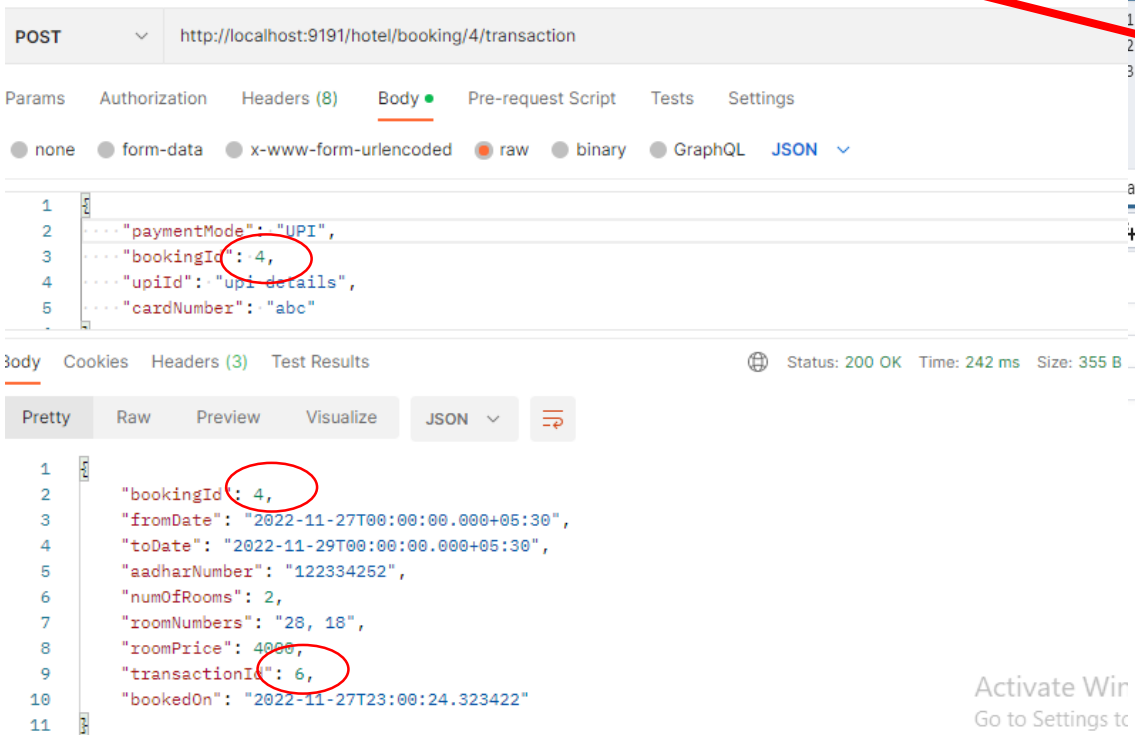


Booking Service



```
40      }
41
42      /* This method makes and internal call to the payments service,
43      * when the user requests the call on booking/{id}/transaction
44      * The call is made using the rest template and the parameters are
45      * injected from the constants file application.properties
46      * The url is injected with the @Value("${transactionApp.url}") annotation
47      * */
        1 usage  gitcrumbs *
48  @   public Booking createTrx(Transaction trxVo, Integer id, Booking bookingentry){
49          trxVo.setBookingId(id);
50
51          Integer item= restTemplate.postForObject(transactionAppUrl, trxVo, Integer.class);
52
53          bookingentry.setTransactionId(item);
54
55          bookingRepository.save(bookingentry);
56
57          return bookingentry;
58      }
```

- Creates the transaction on the payments service
- Extracts the response and updates the trx id object
- Updates the db with the transaction id



POST http://localhost:9191/hotel/booking/4/transaction

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

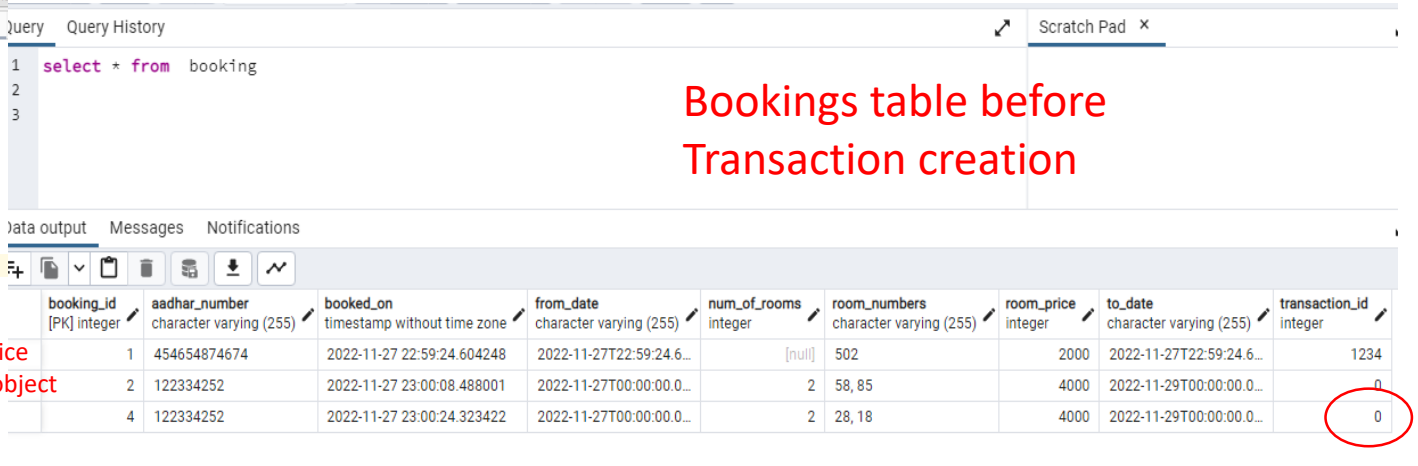
```
{
    "paymentMode": "UPI",
    "bookingId": 4,
    "upiId": "upi details",
    "cardNumber": "abc"
}
```

Body Cookies Headers (3) Test Results Status: 200 OK Time: 242 ms Size: 355 B

Pretty Raw Preview Visualize JSON

```
{
    "bookingId": 4,
    "fromDate": "2022-11-27T00:00:00.000+05:30",
    "toDate": "2022-11-29T00:00:00.000+05:30",
    "aadharNumber": "122334252",
    "numOfRooms": 2,
    "roomNumbers": "28, 18",
    "roomPrice": 4000,
    "transactionId": 6,
    "bookedOn": "2022-11-27T23:00:24.323422"
}
```


*Booking Service -> Payments Service



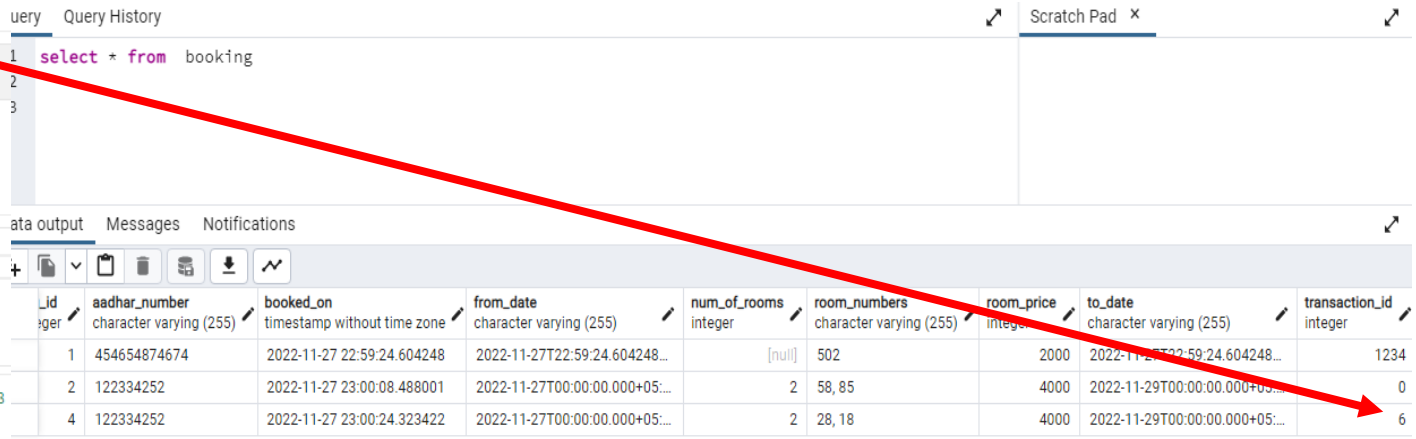
Query Query History Scratch Pad

```
select * from booking
```

Data output Messages Notifications

booking_id [PK] integer	aadhar_number character varying (255)	booked_on timestamp without time zone	from_date character varying (255)	num_of_rooms integer	room_numbers character varying (255)	room_price integer	to_date character varying (255)	transaction_id integer
1	454654874674	2022-11-27 22:59:24.604248	2022-11-27T22:59:24.6...	[null]	502	2000	2022-11-27T22:59:24.6...	1234
2	122334252	2022-11-27 23:00:08.488001	2022-11-27T00:00:00.0...	2	58, 85	4000	2022-11-29T00:00:00.0...	0
4	122334252	2022-11-27 23:00:24.323422	2022-11-27T00:00:00.0...	2	28, 18	4000	2022-11-29T00:00:00.0...	0

Bookings table before Transaction creation



Query Query History Scratch Pad

```
select * from booking
```

Data output Messages Notifications

booking_id integer	aadhar_number character varying (255)	booked_on timestamp without time zone	from_date character varying (255)	num_of_rooms integer	room_numbers character varying (255)	room_price integer	to_date character varying (255)	transaction_id integer
1	454654874674	2022-11-27 22:59:24.604248	2022-11-27T22:59:24.604248...	[null]	502	2000	2022-11-27T22:59:24.604248...	1234
2	122334252	2022-11-27 23:00:08.488001	2022-11-27T00:00:00.000+05:...	2	58, 85	4000	2022-11-29T00:00:00.000+05:...	0
4	122334252	2022-11-27 23:00:24.323422	2022-11-27T00:00:00.000+05:...	2	28, 18	4000	2022-11-29T00:00:00.000+05:...	6

Bookings table after Transaction creation, transaction id is updated

http://localhost:9191/payment/transaction

```
36
    gitcrumbs
37  @PostMapping("/transaction")
38  public ResponseEntity<Integer> createTransaction(@Valid @RequestBody Transaction trxVo){
39
40      Integer created = trxService.createTransaction(trxVo);
41      return new ResponseEntity<>(created, HttpStatus.CREATED);
42
43
44  }
```

Working locally in Scratch Pad. **Switch to a Workspace**

New | Import | GET getPaymentTransacti | POST 2_booking_Transactio | POST 1_booking | POST 3_PaymentTransactio | No Er

sweet-home-booking / 3_PaymentTransaction | Save

POST | http://localhost:9191/payment/transaction ← Payments Service

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON

```
{
    "paymentMode": "UPI",
    "bookingId": 4,
    "upiId": "upi details for bookingId 4",
    "cardNumber": "abc"
}
```

Body | Cookies | Headers (3) | Test Results | Status: 201 Created | Time: 429 ms | Size:

Pretty | Raw | Preview | Visualize | JSON

```
1  5
```

pgAdmin 4

File | Object | Tools | Help

Dashboard | Properties | SQL | Statistics | myDB/postgres@PostgreSQL 14*

myDB/postgres@PostgreSQL 14 | No limit

Query | Query History

```
1  select * from  transaction
2
3
```

Data output | Messages | Notifications

	transaction_id [PK] integer	booking_id integer	card_number character varying (255)	payment_mode character varying (255)	upi_id character varying (255)
1	1	1234	1234567	Card	12345@upi

pgAdmin 4

File | Object | Tools | Help

Dashboard | Properties | SQL | Statistics | myDB/postgres@PostgreSQL 14*

myDB/postgres@PostgreSQL 14 | No limit

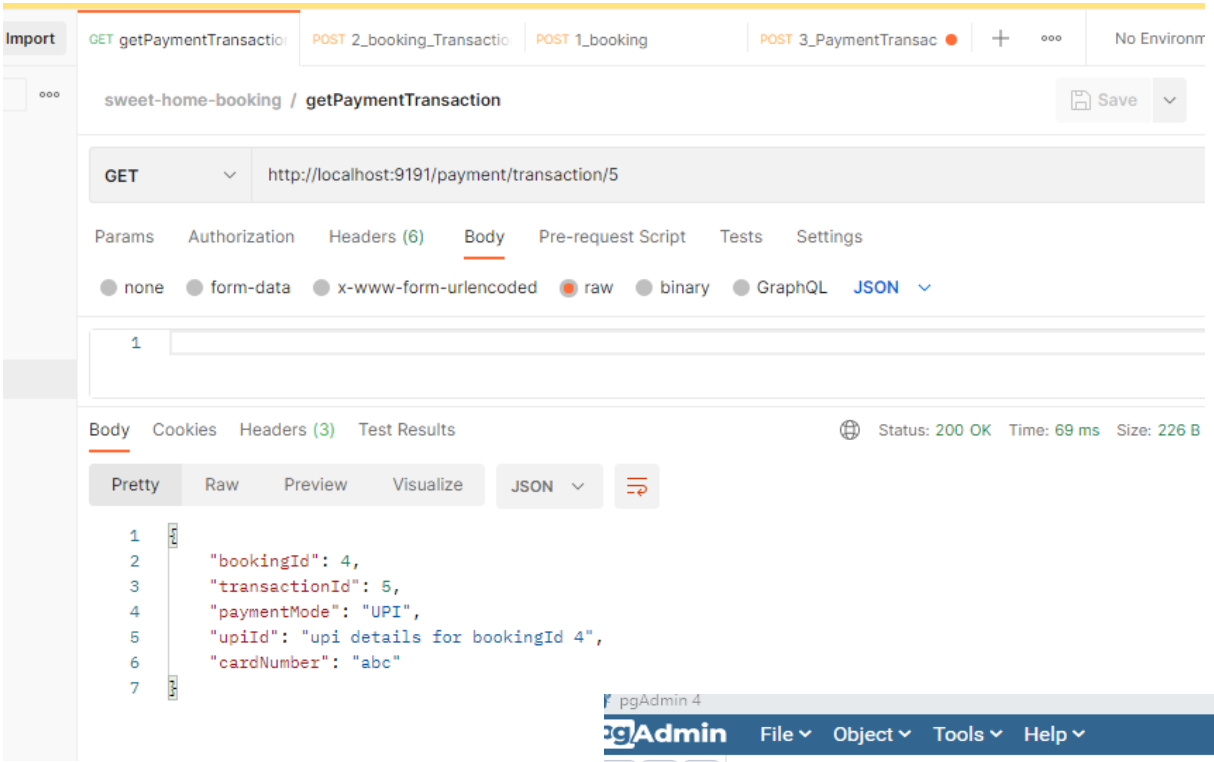
Query | Query History

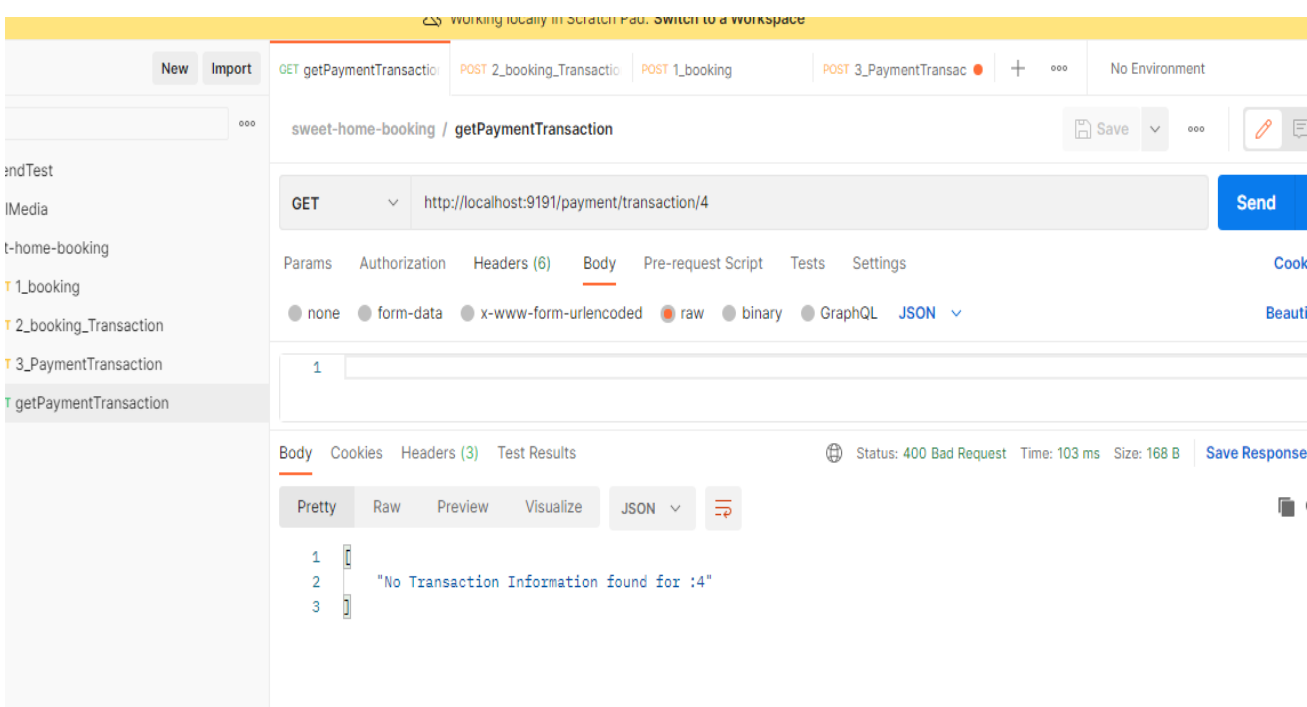
```
1  select * from  transaction
2
3
```

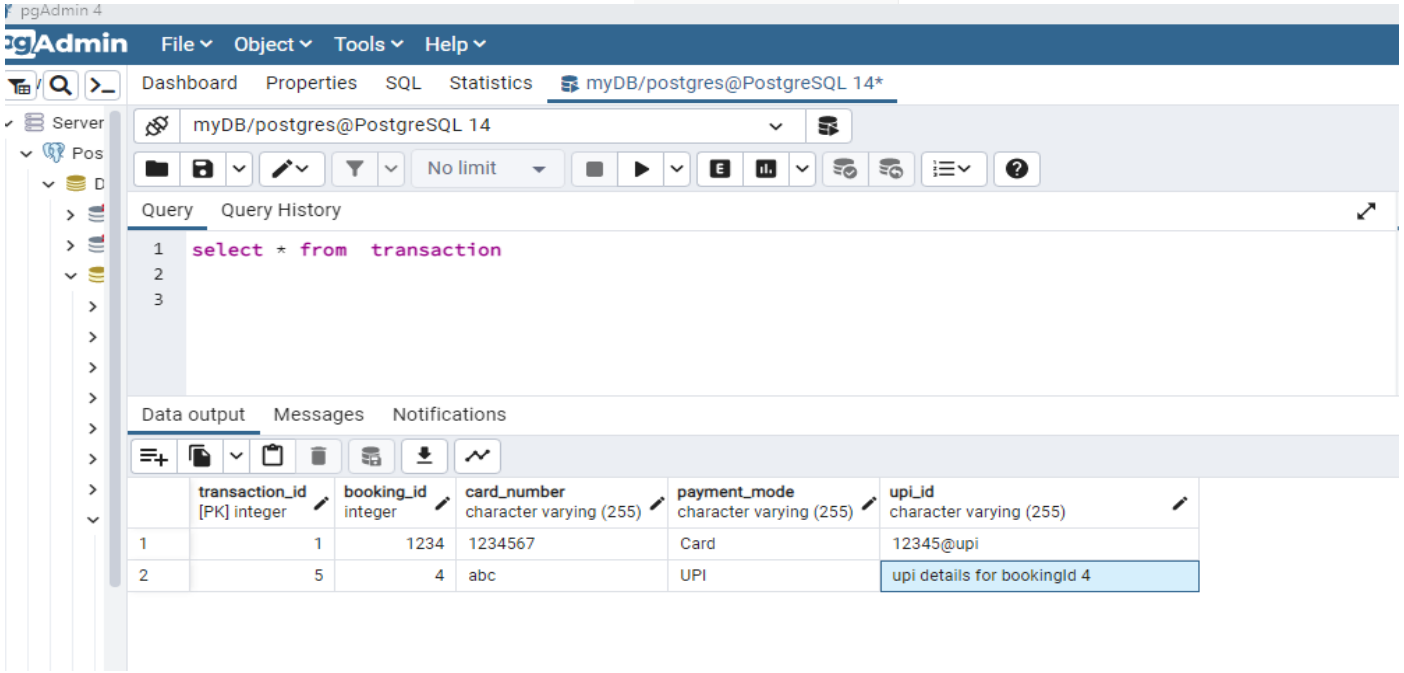
Data output | Messages | Notifications

	transaction_id [PK] integer	booking_id integer	card_number character varying (255)	payment_mode character varying (255)	upi_id character varying (255)
1	1	1234	1234567	Card	12345@upi
2	5	4	abc	UPI	upi details for bookingId 4

Payment Service

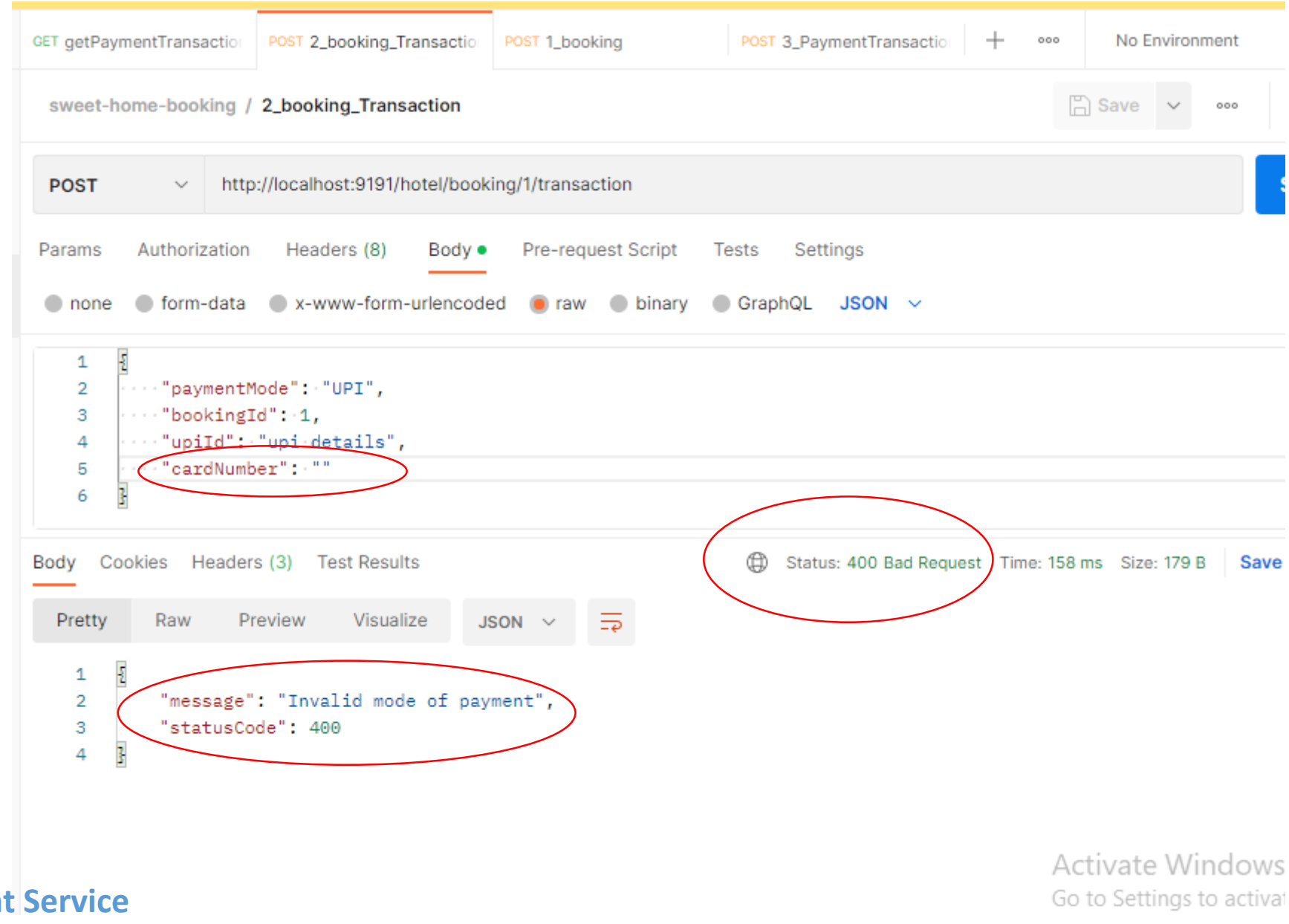






Payment Service

Page 2



Payment Service

```
.137  INFO 23264 --- [nio-8081-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
.141  INFO 23264 --- [nio-8081-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 4 ms
.303  INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect  : In TransactionController entering createTransaction
.339  INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect  : In TransactionService entering createTransaction
.368  INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect  : In CrudRepository entering save
iextval ('hibernate_sequence')
.nto transaction (booking_id, card_number, payment_mode, upi_id, transaction_id) values (?, ?, ?, ?, ?)
.472  INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect  : In CrudRepository exiting save
.472  INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect  : In TransactionService exiting createTransaction
.474  INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect  : In TransactionController exiting createTransaction
.374  INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect  : In TransactionController entering createTransaction
.375  INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect  : In TransactionService entering createTransaction
.376  INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect  : In CrudRepository entering save
iextval ('hibernate_sequence')
.nto transaction (booking_id, card_number, payment_mode, upi_id, transaction_id) values (?, ?, ?, ?, ?)
.383  INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect  : In CrudRepository exiting save
.383  INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect  : In TransactionService exiting createTransaction
.383  INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect  : In TransactionController exiting createTransaction
```

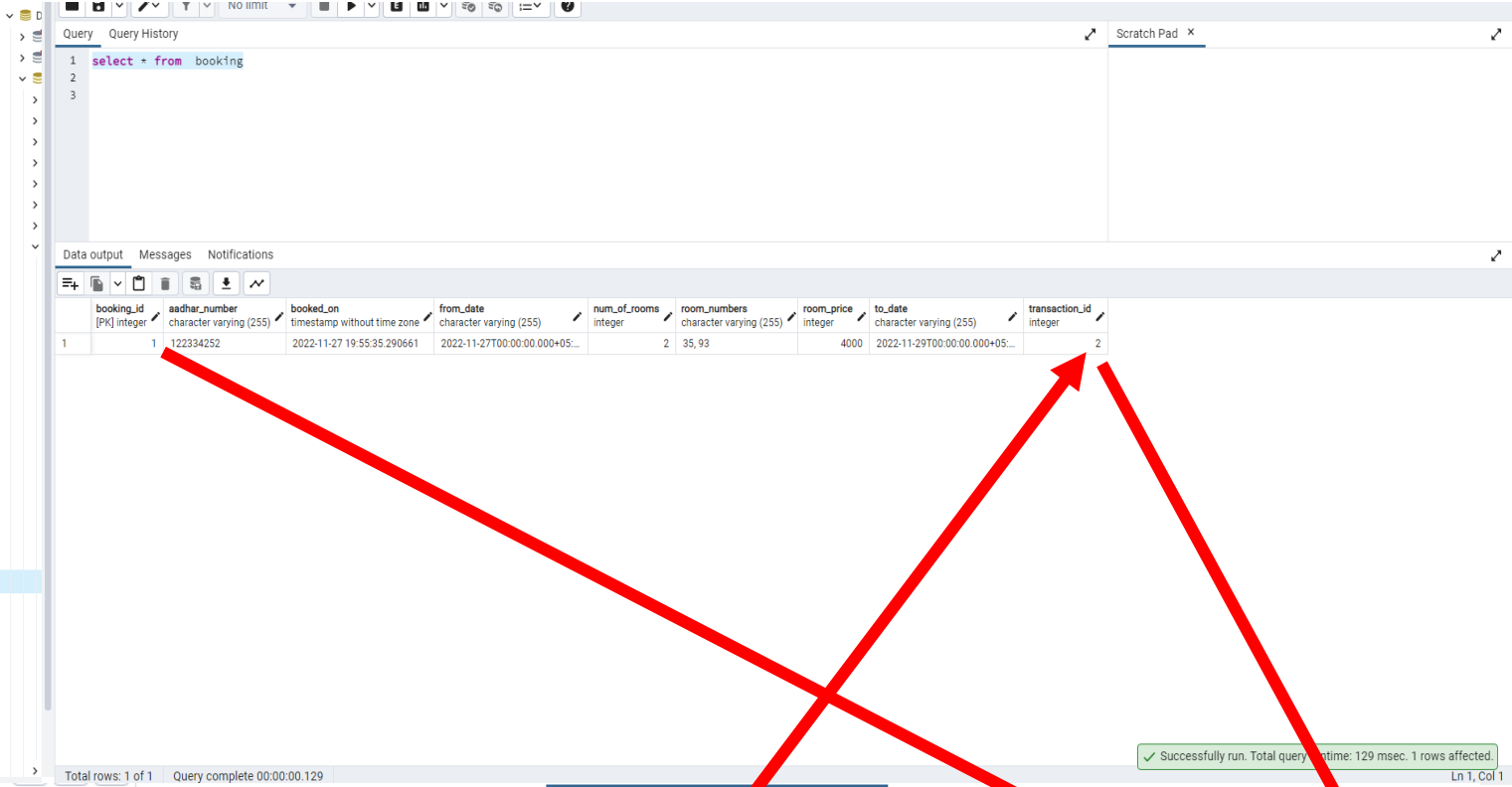
```
)-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
)-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
)-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 3 ms
)-8080-exec-1] c.e.s.booking.aspect.LoggingAspect       : In BookingsController entering createBooking
)-8080-exec-1] c.e.s.booking.aspect.LoggingAspect       : In BookingsService entering createBooking
)-8080-exec-1] c.e.s.booking.aspect.LoggingAspect       : In CrudRepository entering save
nce')
er, booked_on, from_date, num_of_rooms, room_numbers, room_price, to_date, transaction_id, booking_id) values (?, ?, ?, ?, ?, ?, ?, ?, ?)
-8080-exec-1] c.e.s.booking.aspect.LoggingAspect       : In CrudRepository existing save
-8080-exec-1] c.e.s.booking.aspect.LoggingAspect       : In BookingsService existing createBooking
-8080-exec-1] c.e.s.booking.aspect.LoggingAspect       : In BookingsController existing createBooking
-8080-exec-2] c.e.s.booking.aspect.LoggingAspect       : In BookingsController entering createBooking
-8080-exec-2] c.e.s.booking.aspect.LoggingAspect       : In BookingsService entering createBooking
-8080-exec-2] c.e.s.booking.aspect.LoggingAspect       : In CrudRepository entering save
nce')
er, booked_on, from_date, num_of_rooms, room_numbers, room_price, to_date, transaction_id, booking_id) values (?, ?, ?, ?, ?, ?, ?, ?, ?)
-8080-exec-2] c.e.s.booking.aspect.LoggingAspect       : In CrudRepository existing save
-8080-exec-2] c.e.s.booking.aspect.LoggingAspect       : In BookingsService existing createBooking
-8080-exec-2] c.e.s.booking.aspect.LoggingAspect       : In BookingsController existing createBooking
```

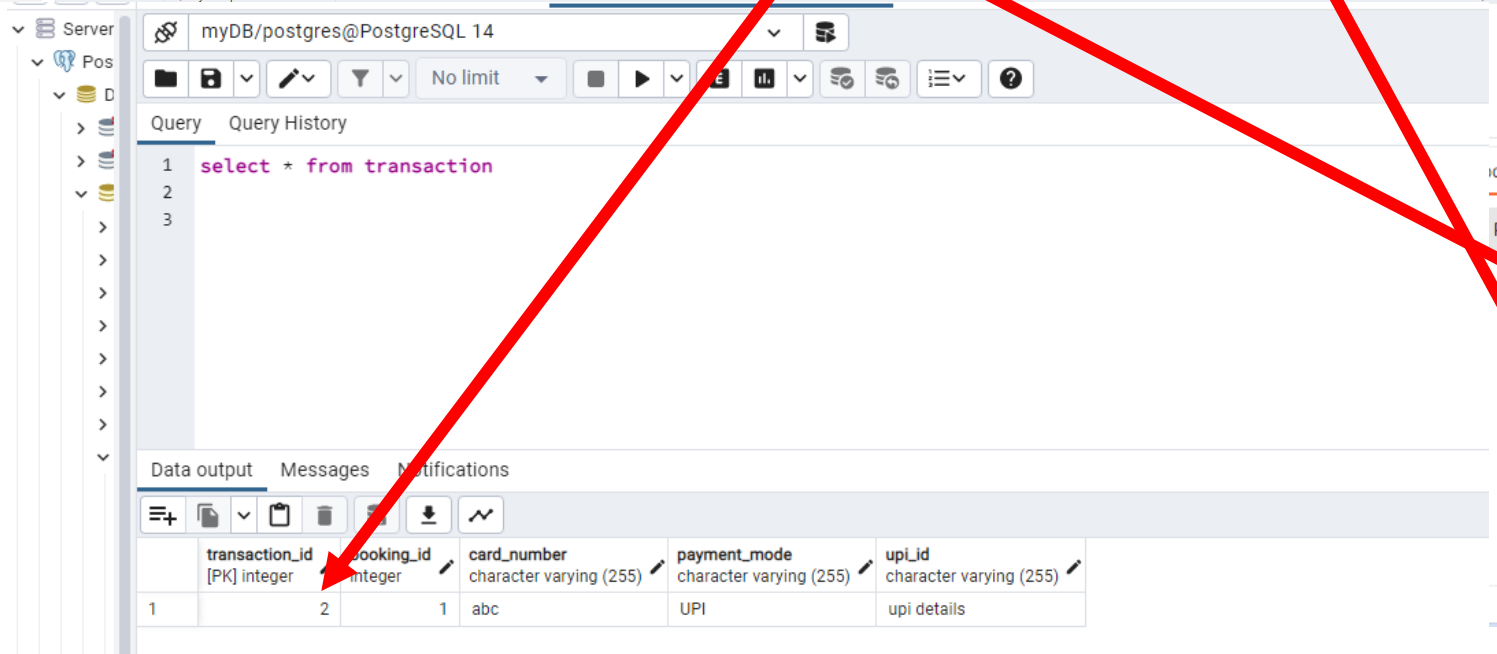
```
LoggingAspect > applyLogging
TransactionController.java | LoggingAspect.java | application.properties | TransactionService.java | Transaction.java | ErrorResponse.java

import org.springframework.stereotype.Component;

1 usage  gitcrumbs*
@Aspect
@Component
public class LoggingAspect {
    2 usages
    Logger logger = LoggerFactory.getLogger(LoggingAspect.class);
    @Around("execution(* com.example.payments..*(..))")
    public Object applyLogging(ProceedingJoinPoint joinPoint) throws Throwable{
        MethodSignature signature = (MethodSignature) joinPoint.getSignature();
        String className = signature.getDeclaringType().getSimpleName();
        String methodName = signature.getName();

        logger.info("In "+className+" entering "+methodName);
        Object result= joinPoint.proceed();
```

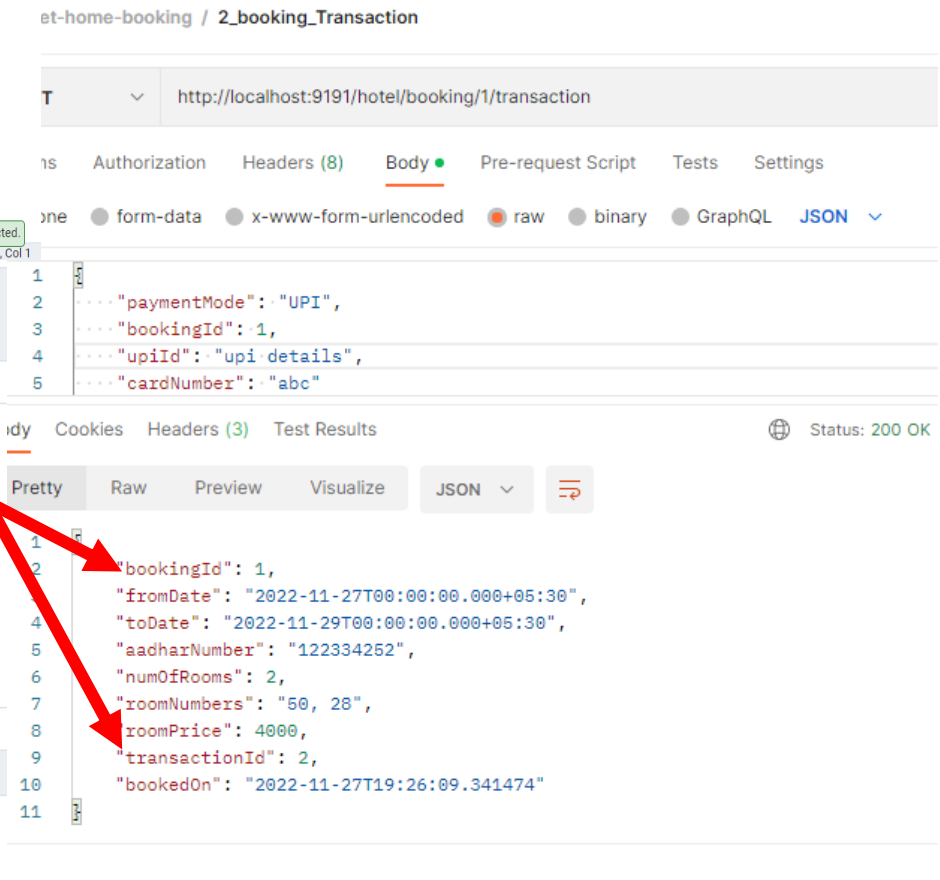
```
booking D:\Upgrad_Projects\Sweet-home\serv

main
java
com.example.service.booking
aspect
LoggingAspect
Controller
Dao
entities
Booking
Transaction
exceptions
BookingNotFoundException
CustomExceptionHandler
ErrorResponse
RecordNotFoundException
repository

import org.aspectj.lang.reflect.MethodSignature;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Component;

1 usage  gitcrumbs*
@Aspect
@Component
public class LoggingAspect {
    2 usages
    Logger logger = LoggerFactory.getLogger(LoggingAspect.class);
    @Around("execution(* com.example.service.booking..*(..))")
    public Object applyLogging(ProceedingJoinPoint joinPoint) throws Throwable{
        MethodSignature signature = (MethodSignature) joinPoint.getSignature();
        String className = signature.getDeclaringType().getSimpleName();
        String methodName = signature.getName();
```

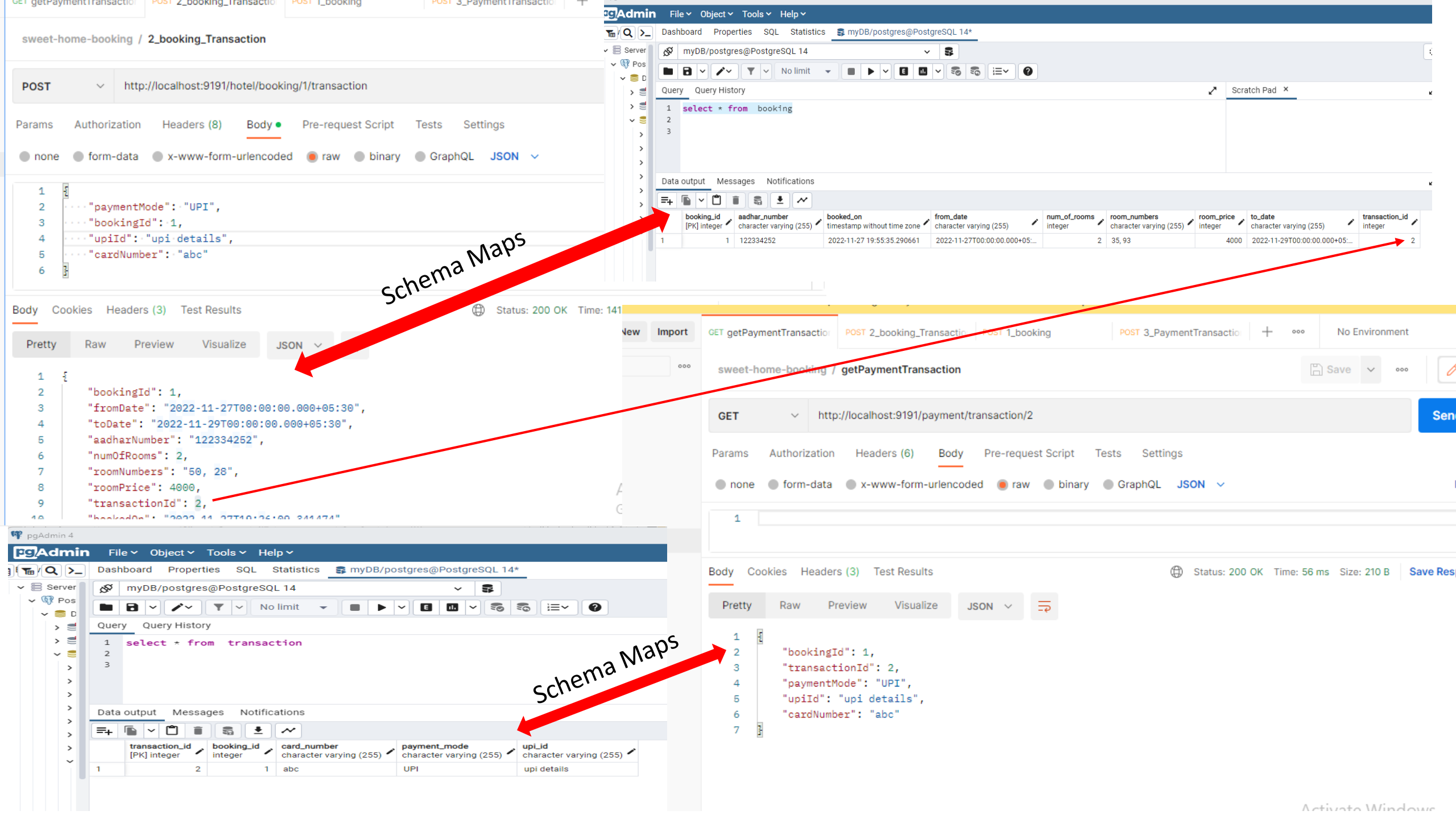
AOP Logging Aspects

Database Entries for Booking and Transactions








sweet-home-booking / 2_booking_Transaction
POST
http://localhost:9191/hotel/booking/1/transaction
Params Authorization Headers (8) Body Pre-request Script Tests Settings
none form-data x-www-form-urlencoded raw binary GraphQL JSON
{
"paymentMode": "UPI",
"bookingId": 1,
"upiId": "upi details",
"cardNumber": "abc"
}
Schema Maps
select * from booking
Data output Messages Notifications
booking_id [PK] integer
aadhar_number character varying (255)
booked_on timestamp without time zone
from_date character varying (255)
num_of_rooms integer
room_numbers character varying (255)
room_price integer
to_date character varying (255)
transaction_id integer
1 1 122334252 2022-11-27 19:55:35.290661 2022-11-27T00:00:00.000+05:... 2 35, 93 4000 2022-11-29T00:00:00.000+05:... 2
Body Cookies Headers (3) Test Results Status: 200 OK Time: 141
Pretty Raw Preview Visualize JSON
{
"bookingId": 1,
"fromDate": "2022-11-27T00:00:00.000+05:30",
"toDate": "2022-11-29T00:00:00.000+05:30",
"aadharNumber": "122334252",
"numOfRooms": 2,
"roomNumbers": "50, 28",
"roomPrice": 4000,
"transactionId": 2,
GET getPaymentTransactio
POST 2_booking_Transactio
POST 1_booking
POST 3_PaymentTransactio
No Environment
sweet-home-booking / getPaymentTransaction
Save
GET
http://localhost:9191/payment/transaction/2
Params Authorization Headers (6) Body Pre-request Script Tests Settings
none form-data x-www-form-urlencoded raw binary GraphQL JSON
Body Cookies Headers (3) Test Results Status: 200 OK Time: 56 ms Size: 210 B
Pretty Raw Preview Visualize JSON
{
"bookingId": 1,
"transactionId": 2,
"paymentMode": "UPI",
"upiId": "upi details",
"cardNumber": "abc"
}
Schema Maps
select * from transaction
Data output Messages Notifications
transaction_id [PK] integer
booking_id integer
card_number character varying (255)
payment_mode character varying (255)
upi_id character varying (255)
1 2 1 abc UPI upi details