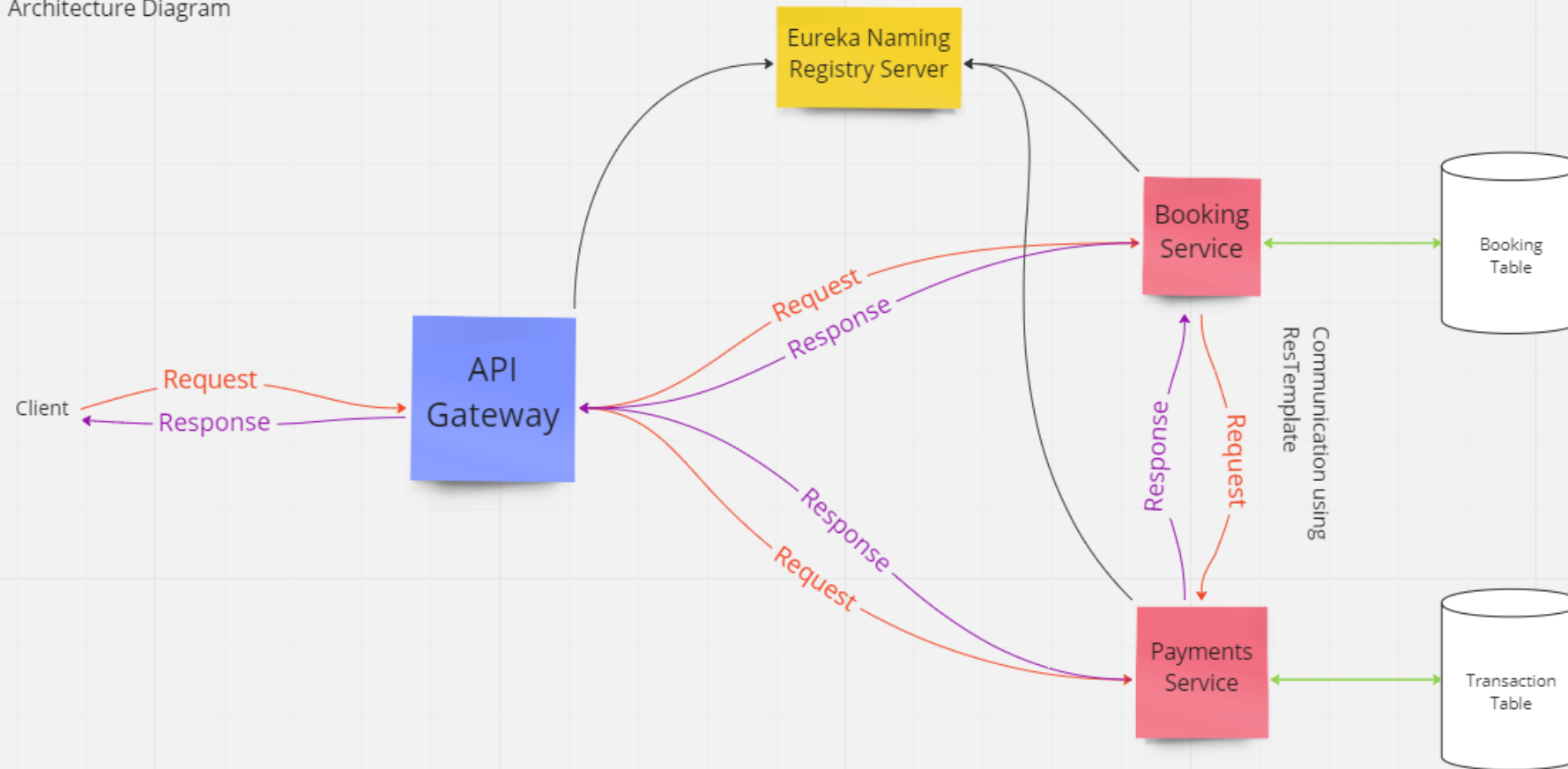


Architecture Diagram



— API Requests —>

— API Response —>

— Naming Registry —>

- DB Communications ->

# DS Replicas

## Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
API-GATEWAY	n/a (1)	(1)	UP (1) - <a href="#">localhost:api-gateway:9191</a>
BOOKING-SERVICE	n/a (1)	(1)	UP (1) - <a href="#">localhost:booking-service:8080</a>
PAYMENT-SERVICE	n/a (1)	(1)	UP (1) - <a href="#">localhost:payment-service:8081</a>

API calls through Gateway

## General Info

sweet-home-booking / 3\_PaymentTransaction

POST http://localhost:9191/payment/transaction

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

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

```
1 {
2   "paymentMode": "UPI",
3   "bookingId": 1,
4   "upiId": "upi_details",
5   "cardNumber": "abc"
6 }
```

Body Cookies Headers (3) Test Results

Status: 201 Created Time: 44 ms Size: 122 B Save Response

Pretty Raw Preview Visualize JSON

1 4

sweet-home-booking / getPaymentTransaction

GET http://localhost:9191/payment/transaction/2

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Beautify

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

1

Body Cookies Headers (3) Test Results

Status: 200 OK Time: 56 ms Size: 210 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "bookingId": 1,
3   "transactionId": 2,
4   "paymentMode": "UPI",
5   "upiId": "upi_details",
6   "cardNumber": "abc"
7 }
```

http://localhost:9191/hotel/booking

The screenshot shows the REST Client interface with a POST request to `http://localhost:9191/hotel/booking`. The request body is a JSON object:

```
{
  "fromDate": "2022-11-27",
  "toDate": "2022-11-29",
  "aadhaarNumber": "122334252",
  "numOfRooms": 2
}
```

The response body is a JSON object:

```
{
  "bookingId": 1,
  "fromDate": "2022-11-27 03:30:00",
  "toDate": "2022-11-29 03:30:00"
}
```

## Code Flow

Page 1

Service > booking > Controller > BookingsController > getOrder

ServiceBookingApplication.java BookingsController.java application.properties Booking.java

17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52

```
1 usage gitcrumbs *
18 @RestController
19 @Validated
20 @RequestMapping("/hotel")
21 public class BookingsController {
22     1 usage
23     Logger logger = LoggerFactory.getLogger(BookingsController.class);
24     6 usages
25     BookingsService bookingsService;
26
27     gitcrumbs
28     public BookingsController(BookingsService bookingsService) { this.bookingsService = bookingsService; }
29
30     gitcrumbs
31     @GetMapping("/status")
32     public ResponseEntity getOrder() {
33
34         return ResponseEntity.ok(bookingsService.getAllBookings());
35     }
36
37     gitcrumbs
38     @PostMapping("/booking")
39     public ResponseEntity<Booking> createBooking(@RequestBody Booking trxVo) {
40
41         return new ResponseEntity<>(bookingsService.createBooking(trxVo), HttpStatus.CREATED);
42     }
43
44     gitcrumbs *
45     @PostMapping("/booking/{id}/transaction")
46     public ResponseEntity getBookingStatus(@Valid @RequestBody Transaction trxVo, @PathVariable Integer id) {
47
48         Booking bookingentry = bookingsService.getBookingStatus(id).orElseThrow(() -> new RecordNotFoundException("Booking not found for the given id "+id));
49         Booking bookingentryservice = new Booking();
50         if(bookingentry.getBookingId()!=null){
51             System.out.println("Booking id found "+bookingentry.getBookingId() );
52             bookingentryservice = bookingsService.createTrx(trxVo, id,bookingentry);
53         }
54
55         logger.info("Booking confirmed for user with aadhaar number: "
56             + bookingentryservice.getAadhaarNumber());
57     }
58 }
```

Services Build Dependencies

```

33 }
34
35 /* The requested post call for booking brings the code flow here
36 * and after formatting the data, the entry is saved in the booking repository*/
37 1 usage gitcrumbs*
38 public Booking createBooking(Booking booking) throws Exception {
39     Booking bookedItem = bookingRepository.save(formatDataEntry(booking));
40     return bookedItem;
41 }
42
43 1 usage gitcrumbs
44 @ public Booking createTrx(Transaction trxVo, Integer id, Booking bookingentry){ //
45     trxVo.setBookingId(id);
46
47     Integer item= restTemplate.postForObject(transactionAppUrl, trxVo, Integer.class);
48     System.out.println("Generated Transaction id is"+item);
49     bookingentry.setTransactionId(item);
50
51     bookingRepository.save(bookingentry);
52
53     return bookingentry;
54 }

```

```

ServiceBookingApplication
BookingsService
formatDataEntry
ServiceBookingApplication.java
BookingsController.java
BookingsService.java
application.properties
Booking.java
73
74 /* This code block formats the data as per the given format in
75 the problem statement*/
76 1 usage new*
77 @ public Booking formatDataEntry(Booking booking)throws Exception{
78     DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd'T'HH:mm:ss.SSSXXX");
79     int numofRooms = booking.getNumOfRooms();
80     String s= getRandomNumbers(numOfRooms).toString();
81     String requiredString = s.substring(s.indexOf("[")+1 , s.indexOf("]"));
82     booking.setRoomNumbers(requiredString);
83     try{
84         booking.setBookedOn(LocalDate.now());
85         Calendar cl = Calendar.getInstance();
86         Timestamp filterDateFromTs = null,filterDateToTs=null;
87         SimpleDateFormat dateFormat = new SimpleDateFormat( pattern: "yyyy-MM-dd");
88         Date firstDate = dateFormat.parse(booking.getFromDate());
89         Date secondDate = dateFormat.parse(booking.getToDate());
90         long diffInMillies = Math.abs(secondDate.getTime() - firstDate.getTime());
91         long daysBetween = TimeUnit.DAYS.convert(diffInMillies, TimeUnit.MILLISECONDS);
92
93         long roomPrice = 1000 * numofRooms*(daysBetween);
94         String timeStampFrom =new SimpleDateFormat( pattern: "yyyy-MM-dd'T'HH:mm:ss.SSSXXX").format((dateFormat.parse(booking.getFromDate())).getTime());
95         String timeStampTo =new SimpleDateFormat( pattern: "yyyy-MM-dd'T'HH:mm:ss.SSSXXX").format((dateFormat.parse(booking.getToDate())).getTime());
96         booking.setFromDate(timeStampFrom);
97         booking.setToDate(timeStampTo);
98         booking.setRoomPrice((int)roomPrice);
99     }catch(Exception e){
100         throw new Exception("Unable to create Transaction"+e.getMessage());
101     }
102
103     return booking;
104 }
105
106
107

```

## Code Flow: Booking Service

```

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

```

Activate Window

3 usages

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

```
1 spring.application.name=booking-service
2 spring.jpa.hibernate.ddl-auto=create
3 spring.datasource.initialization-mode=always
4 spring.datasource.platform=postgres
5 spring.datasource.url=jdbc:postgresql://localhost:5432/myDB
6 spring.datasource.username=postgres
7 spring.datasource.password=postgres
8 spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true
9
10
11 transactionApp.url=http://localhost:8081/payment/transaction
```

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

application.properties

```
/* This method makes an 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

```
1
2 ... "paymentMode": "UPI",
3 ... "bookingId": 1,
4 ... "upiId": "upi details",
5 ... "cardNumber": ""
6
```

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize JSON

```
1
2 "message": "Invalid mode of payment",
3 "statusCode": 400
4
```

Status: 400 Bad Request Time: 158 ms Size: 179 B Save Response

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

```
1
2 ... "paymentMode": "UPI",
3 ... "bookingId": 2,
4
5 "bookingId": 2,
6 "fromDate": "2022-11-27T00:00:00.000+05:30",
7 "toDate": "2022-11-29T00:00:00.000+05:30",
8 "aadharNumber": "122334252",
9 "numOfRooms": 2,
10 "roomNumbers": "79, 74",
11 "roomPrice": 4000,
12 "transactionId": 3,
13 "bookedOn": "2022-11-27T22:24:22.232142"
```

Status: 200 OK Time: 590 ms Size: 355 B

Pretty Raw Preview Visualize JSON

\*Booking Service -> Payments Service

```
booking > Controller > BookingsController > getBookingStatus

ServiceBookingApplication.java x BookingsController.java x BookingsService.java x application.properties x Booking.java x T

if(bookingEntry.getBookingId()!=null){
    System.out.println("Booking id found "+bookingEntry.getBookingId() );
    bookingEntry = bookingsService.createTrx(trxVo, id,bookingEntry);
}

logger.info("Booking confirmed for user with aadhaar number: "
    + bookingEntry.getAadhaarNumber()
    + " | "
    + "Here are the booking details:  "+ bookingEntry.toString());
return ResponseEntity.ok(bookingEntry);
}
```

```
application.yml
> test
> target
.gitignore
mvnw
mvnw.cmd
pom.xml
service-booking.iml
External Libraries
Scratches and Consoles

74
75
76
77
78
79

return response.orElseThrow(()-> new RecordNotFoundException("Booking not generated for the id :"+id));
}

Run: ServiceBookingApplication x
: Completed initialization in 3 ms
., booking0_.booked_on as booked_o3_0_0_, booking0_.from_date as from_dat4_0_0_, booking0_.num_of_rooms as num_of_r5_0_0_, booking0_.room_numbers as room_num6_0_0_, booking0_.room_price as room_pri7_0_0_, booking0_.to_date as :
., booking0_.booked_on as booked_o3_0_0_, booking0_.from_date as from_dat4_0_0_, booking0_.num_of_rooms as num_of_r5_0_0_, booking0_.room_numbers as room_num6_0_0_, booking0_.room_price as room_pri7_0_0_, booking0_.to_date as :
om_price, to_date, transaction_id, booking_id) values (?, ?, ?, ?, ?, ?, ?, ?)
., booking0_.booked_on as booked_o3_0_0_, booking0_.from_date as from_dat4_0_0_, booking0_.num_of_rooms as num_of_r5_0_0_, booking0_.room_numbers as room_num6_0_0_, booking0_.room_price as room_pri7_0_0_, booking0_.to_date as :
., booking0_.booked_on as booked_o3_0_0_, booking0_.from_date as from_dat4_0_0_, booking0_.num_of_rooms as num_of_r5_0_0_, booking0_.room_numbers as room_num6_0_0_, booking0_.room_price as room_pri7_0_0_, booking0_.to_date as :
rs=?, room_price=?, to_date=?, transaction_id=? where booking_id=?
: Booking confirmed for user with aadhaar number: 122334252 | Here are the booking details: Booking{bookingId=3, fromDate=2022-11-27T00:00:00.000+05:30, toDate=2022-11-29T00:00:00.000+05:30, aadhaarNumber='122334252',

Git Run TODO Problems Terminal Services Build Dependencies
Build completed successfully in 1 min 17 sec (3 minutes ago)
79:1 CRIF UTF-8 4 spaces Implementation Fixes
```







http://localhost:9191/payment/transaction

36  
37  
38  
39  
40  
41  
42  
43  
44

gitcrumbs

@PostMapping("/transaction")

public ResponseEntity<Integer> createTransaction(@Valid @RequestBody Transaction trxVo){

Integer created = trxService.createTransaction(trxVo);

return new ResponseEntity<>(created, HttpStatus.CREATED);

}

Working locally in Scratch Pad. Switch to a Workspace

NewImport

GET getPaymentTransactionPOST 2\_booking\_TransactionPOST 1\_bookingPOST 3\_PaymentTransaction

...

sweet-home-booking / 3\_PaymentTransaction

Save

POST

http://localhost:9191/payment/transaction

Payments Service

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettings

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSON

1  
2  
3  
4  
5  
6

"paymentMode": "UPI",

"bookingId": 4,

"upiId": "upi details for bookingId 4",

"cardNumber": "abc"

BodyCookiesHeaders (3)Test Results

Status: 201 CreatedTime: 429 msSize:

PrettyRawPreviewVisualizeJSON

15

Payment Service

pgAdmin 4

pgAdminFileObjectToolsHelp

DashboardPropertiesSQLStatisticsmyDB/postgres@PostgreSQL 14\*

ServermyDB/postgres@PostgreSQL 14

Pos

QueryQuery History

1  
2  
3

select \* from transaction

Data outputMessagesNotifications

transaction\_id [PK] integerbooking\_id integercard\_number character varying (255)payment\_mode character varying (255)upi\_id character varying (255)

1112341234567Card12345@upi

pgAdmin 4

pgAdminFileObjectToolsHelp

DashboardPropertiesSQLStatisticsmyDB/postgres@PostgreSQL 14\*

ServermyDB/postgres@PostgreSQL 14

Pos

QueryQuery History

1  
2  
3

select \* from transaction

Data outputMessagesNotifications

transaction\_id [PK] integerbooking\_id integercard\_number character varying (255)payment\_mode character varying (255)upi\_id character varying (255)

1112341234567Card12345@upi

254abcUPIupi details for bookingId 4

Import GET getPaymentTransaction POST 2\_booking\_Transaction POST 1\_booking POST 3\_PaymentTransac + ... No Environr

sweet-home-booking / getPaymentTransaction Save

GET http://localhost:9191/payment/transaction/5

Params Authorization Headers (6) 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: 200 OK Time: 69 ms Size: 226 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "bookingId": 4,
3   "transactionId": 5,
4   "paymentMode": "UPI",
5   "upiId": "upi details for bookingId 4",
6   "cardNumber": "abc"
7 }
```

New Import GET getPaymentTransaction POST 2\_booking\_Transaction POST 1\_booking POST 3\_PaymentTransac + ... No Environment

sweet-home-booking / getPaymentTransaction Save

GET http://localhost:9191/payment/transaction/4 Send

Params Authorization Headers (6) 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: 400 Bad Request Time: 103 ms Size: 168 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "No Transaction Information found for :4"
3 }
```

pgAdmin 4

pgAdmin File Object Tools Help

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

myDB/postgres@PostgreSQL 14

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

GET getPaymentTransaction POST 2\_booking\_Transaction POST 1\_booking POST 3\_PaymentTransaction + ... No Environment

sweet-home-booking / 2\_booking\_Transaction Save

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

```
1 {
2   ... "paymentMode": "UPI",
3   ... "bookingId": 1,
4   ... "upiId": "upi-details",
5   "cardNumber": ""
6 }
```

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Invalid mode of payment",
3   "statusCode": 400
4 }
```

Status: 400 Bad Request Time: 158 ms Size: 179 B Save

Activate Windows  
Go to Settings to activate Windows.

```

1.137 INFO 23264 --- [nio-8081-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
1.141 INFO 23264 --- [nio-8081-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 4 ms
1.303 INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect : In TransactionController entering createTransaction
1.339 INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect : In TransactionService entering createTransaction
1.368 INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect : In CrudRepository entering save
nextval ('hibernate_sequence')
into transaction (booking_id, card_number, payment_mode, upi_id, transaction_id) values (?, ?, ?, ?, ?)
1.472 INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect : In CrudRepository exiting save
1.472 INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect : In TransactionService exiting createTransaction
1.474 INFO 23264 --- [nio-8081-exec-1] c.example.payments.aspect.LoggingAspect : In TransactionController exiting createTransaction
1.374 INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect : In TransactionController entering createTransaction
1.375 INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect : In TransactionService entering createTransaction
1.376 INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect : In CrudRepository entering save
nextval ('hibernate_sequence')
into transaction (booking_id, card_number, payment_mode, upi_id, transaction_id) values (?, ?, ?, ?, ?)
1.383 INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect : In CrudRepository exiting save
1.383 INFO 23264 --- [nio-8081-exec-3] c.example.payments.aspect.LoggingAspect : In TransactionService exiting createTransaction
1.383 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

```

The screenshot shows the LoggingAspect.java file in an IDE. The class is annotated with `@Aspect` and `@Component`. It has a method `applyLogging` that uses `@Around` to log the execution of methods in the `com.example.payments` package. The IDE also shows the BookingService class with a similar logging aspect applied to its methods.

```

import org.springframework.stereotype.Component;

10
11 @Aspect
12 @Component
13 public class LoggingAspect {
14     2 usages
15     Logger logger = LoggerFactory.getLogger(LoggingAspect.class);
16
17     @Around("execution(* com.example.payments..*(..))")
18     public Object applyLogging(ProceedingJoinPoint joinPoint) throws Throwable{
19         MethodSignature signature = (MethodSignature) joinPoint.getSignature();
20         String className = signature.getDeclaringType().getSimpleName();
21         String methodName = signature.getName();
22
23         logger.info("In "+className+" entering "+methodName);
24         Object result= joinPoint.proceed();
25     }
26 }

```

The IDE also shows the BookingService class with a similar logging aspect applied to its methods.

```

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

10
11 @Aspect
12 @Component
13 public class LoggingAspect {
14     2 usages
15     Logger logger = LoggerFactory.getLogger(LoggingAspect.class);
16
17     @Around("execution(* com.example.service.booking..*(..))")
18     public Object applyLogging(ProceedingJoinPoint joinPoint) throws Throwable{
19         MethodSignature signature = (MethodSignature) joinPoint.getSignature();
20         String className = signature.getDeclaringType().getSimpleName();
21         String methodName = signature.getName();
22
23         logger.info("In "+className+" entering "+methodName);
24         Object result= joinPoint.proceed();
25     }
26 }

```

## AOP Logging Aspects

Database Entries for Booking and Transactions

Query

Query History

1  
2  
3

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	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

Total rows: 1 of 1

Query complete 00:00:00.129

Successfully run. Total query runtime: 129 msec. 1 rows affected.

Server

myDB/postgres@PostgreSQL 14

Pos

No limit

Query

Query History

1  
2  
3

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

et-home-booking / 2\_booking\_Transaction

T

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

ns

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

one

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

id

Cookies

Headers (3)

Test Results

Status: 200 OK

Pretty

Raw

Preview

Visualize

JSON

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11

```
"paymentMode": "UPI",
"bookingId": 1,
"upiId": "upi details",
"cardNumber": "abc"
"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,
"bookedOn": "2022-11-27T19:26:09.341474"
```

getPaymentTransaction POST 2\_booking\_transaction POST 1\_booking POST 3\_PaymentTransaction

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

```
1 {
2   "paymentMode": "UPI",
3   "bookingId": 1,
4   "upiId": "upi details",
5   "cardNumber": "abc"
6 }
```

Body Cookies Headers (3) Test Results

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": "50, 28",
8   "roomPrice": 4000,
9   "transactionId": 2,
10  "bookedOn": "2022-11-27T10:36:00.244474"
```

pgAdmin 4

Schema Maps

pgAdmin 4 File Object Tools Help

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

myDB/postgres@PostgreSQL 14

Query Query History

```
1 select * from booking
2
3
```

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:...

New Import GET getPaymentTransaction POST 2\_booking\_Transaction POST 1\_booking POST 3\_PaymentTransaction

sweet-home-booking / getPaymentTransaction

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

```
1 {
2   "bookingId": 1,
3   "transactionId": 2,
4   "paymentMode": "UPI",
5   "upiId": "upi details",
6   "cardNumber": "abc"
7 }
```

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "bookingId": 1,
3   "transactionId": 2,
4   "paymentMode": "UPI",
5   "upiId": "upi details",
6   "cardNumber": "abc"
7 }
```

pgAdmin 4

myDB/postgres@PostgreSQL 14

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	2	1	abc	UPI

Schema Maps