**­­­­­­­­Mini Project -2 (MicroService Based Application)**

**Hotel Room Booking Application**

***Avtar Singh (20CSU241)***

**1. Context**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO.** | **Section** | **Description** | **Page No.** |
| 1 | Introduction | Provides an overview of the documentation, including its purpose, intended audience, and scope. | **1** |
| 2 | Getting Started | Covers essential information for users to quickly set up and begin using the software, including installation, requirements, and initial configuration. | **3** |
| 3 | High Level Diagram | Explains the working by using High Level Diagram. | **5** |
| 4 | System Configuration | Describes various configuration options, settings, and environment variables that users can customize. | **5** |
| 5 | Installation | Step Wise Guide to Install the Application | **6** |
| 6 | Usage | Tools for Proper Functioning of Application | **15** |
| 7 | Endpoint | Explains how to use the software, including basic and advanced usage scenarios, examples, and troubleshooting tips. | **16** |
| 8 | Documentation Reference | Explains from where the problem statement is taken and though process. | **30** |
| 9 | Changelog | Presents the version history and release notes, documenting changes, improvements, and new features in each release. | **31** |
| 10 | License | Offers avenues for support, such as community forums, contact information, and instructions for reporting bugs. | **32** |
| 11 | References and Links | All the Files and Links Mentioned in the Project. | **32** |
| 12 | Turnitin Plagiarism Report | Screen Short of Plagiarism Report | **33** |

**2. Introduction**

Welcome to our user-friendly hotel booking application! Crafted with care using Java and Spring Boot, this microservice-based project ensures a seamless experience. Behind the scenes, we've integrated MySQL for secure data storage. This project serves as a basic template for a hotel booking application. It includes two main entities: `Hotel` and `Booking`. The application provides RESTful APIs for managing hotels and bookings.

We are using MicroServices Architecture in which we are dividing our application into Three Different Spring Projects :

***API-Gateway: Single Entry Point***

* External Face: Serves as the public interface of our system, handling incoming requests.
* Router: Directs requests to the appropriate microservices within the application.
* Roles : Like a security, ensures smooth communication between external requests and internal services.

***Booking Service: Handles Hotel and Booking***

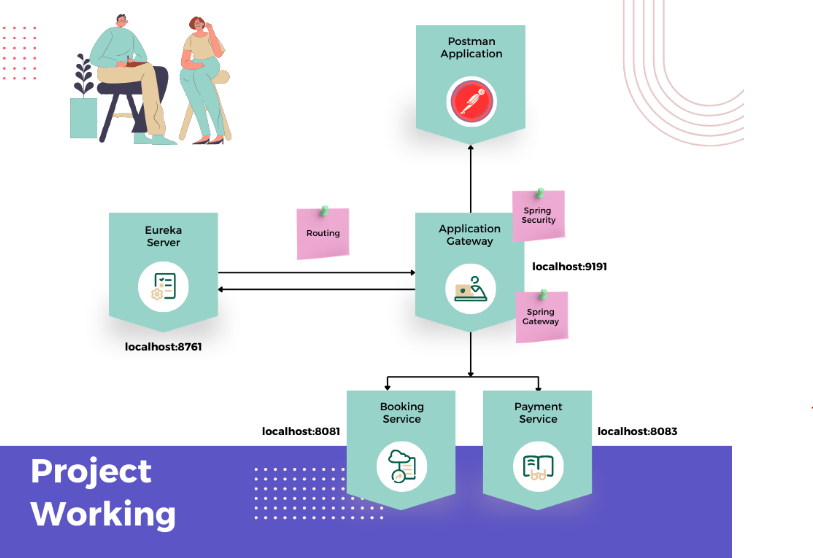
* Information Hub: Gathers and processes user details and booking preferences.
* Confirmation Provider: Manages the confirmation process once a booking is selected.
* Communication Hub: Sends out confirmation messages to users, akin to a hotel receptionist confirming reservations.

***Booking Service: Handles Payments and Transactions***

* Virtual Cashier: Although a "dummy" service, it plays a crucial role in the system.
* Triggered by Booking Service: Initiates the payment process after a booking is confirmed.
* Ensures Transaction: Similar to a cashier, it ensures the seamless execution of financial transactions within the booking system.

**2. Getting Started**

Lets Understand the working using the workflow diagram :



Figure

As Explained by this diagram :

* **Work via PostMan:**

API Gateway, Booking service, and Payment service will work and can we used by API Platform - PostMan , Curl etc.

* **Service Registration:**

API Gateway, Booking service, and Payment service register themselves on the Eureka server during the system initialization.

* **User Interaction Flow:**

All user interactions start with the API Gateway; users don't directly interact with the Booking or Payment services.

The 'Booking' service is triggered when a user initiates room booking, providing details like toDate, fromDate, aadharNumber, and numOfRooms.

* **Booking Service Response:**

'Booking' service responds with a list of available room numbers and their associated prices.

User is prompted to enter payment details if they choose to proceed, and their provided information is stored in the 'Booking' service database.

* **Payment Transaction Process:**

If the user decides to proceed with the booking, they provide payment details (bookingMode, upiId/cardNumber) to the 'Booking' service.

'Booking' service forwards payment details to the 'Payment' service for a dummy transaction.

'Payment' service performs the transaction, returning the associated transaction Id to the 'Booking' service.

All transaction details are stored in the 'Payment' service database.

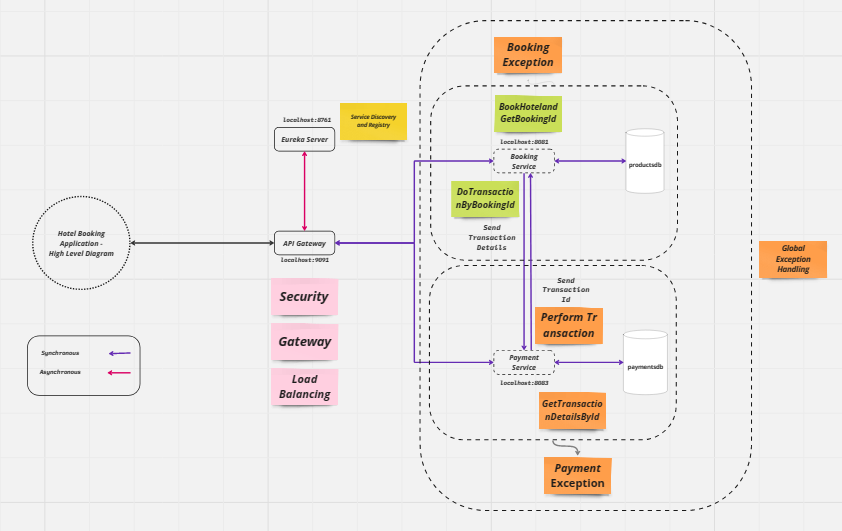
* **Confirmation and Completion:**

Once the transaction is completed, the 'Booking' service acknowledges the confirmed booking by sending a confirmation message to the console.

The entire process ensures a secure and streamlined room booking experience for the user, with clear communication between services and effective storage of relevant data in their respective databases.

**3. High Level Diagram (Hotel Booking Application)**

**Board Link** -> https://miro.com/app/board/uXjVNCqSVsQ=/?share\_link\_id=764323176558



**4. System Configuration**

Make sure you have the following installed on your system:

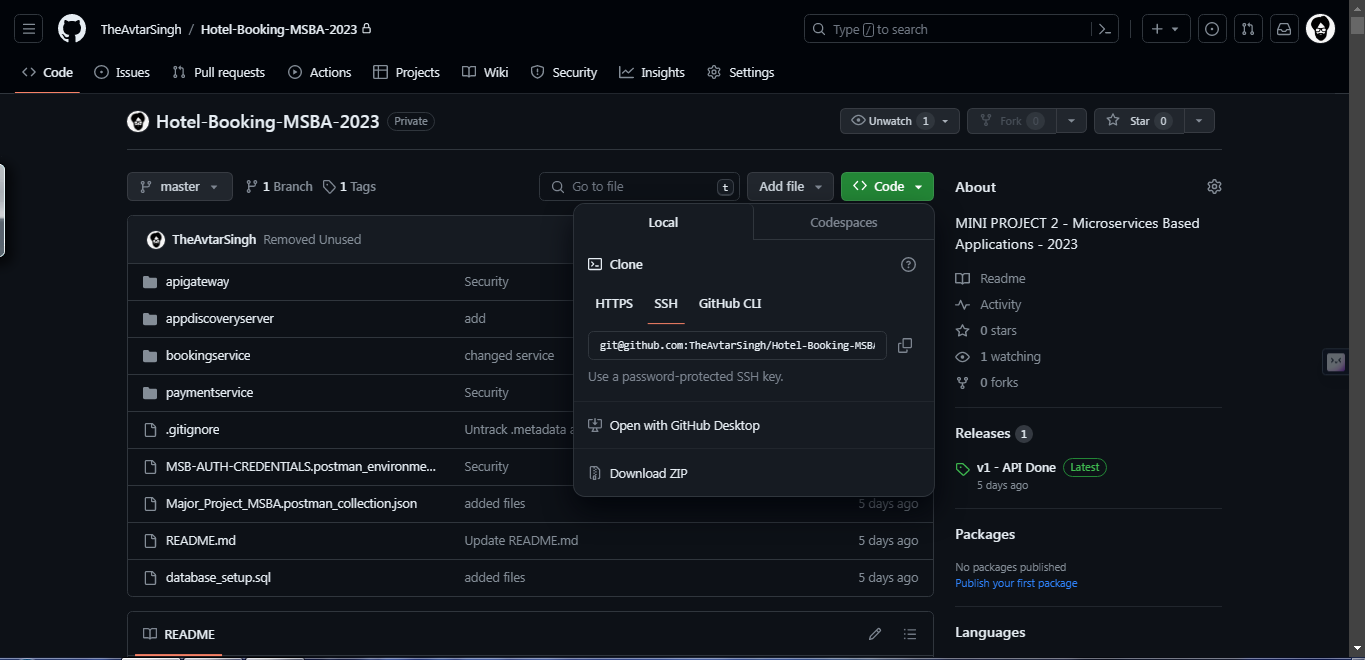
* Java 8 or higher
* Maven
* Spring Boot
* MySQL or another compatible database (for data storage)

**5. Installation**

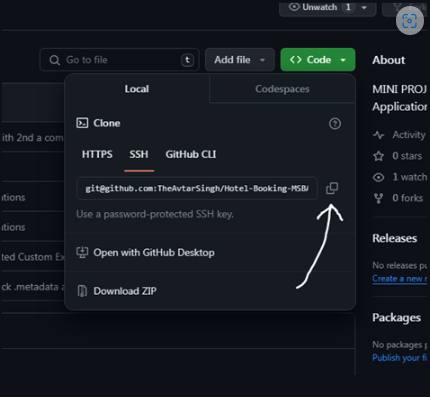
***Follow Proper Procedure to run Smoothly***

1. **Clone the repository**

*a) Find the Link and Clone using Git Bash /Github Desktop ( You Can Use Https/SSH)*

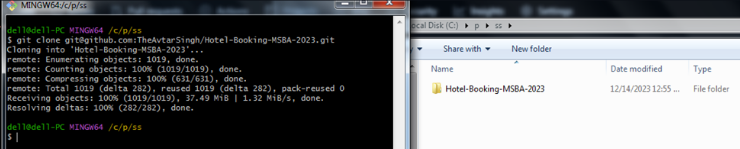
******

*b) Find the Link ( You Can Use Https/SSH)*

******

1. **Clone the repository using**

*git clone https://github.com/TheAvtarSingh/Hotel-Booking-MSBA-2023.git*

**

1. **Setup Database**

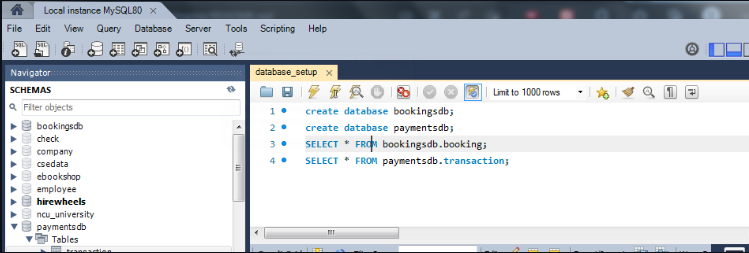
**Use Commands :**

*create database bookingsdb;*

*create database paymentsdb;*

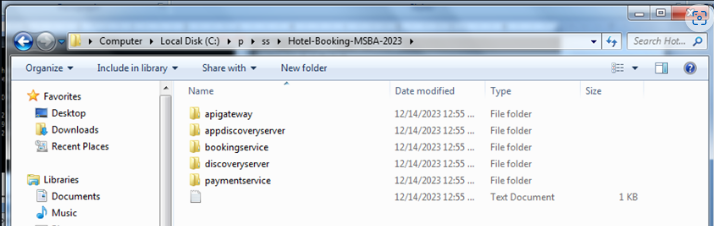
*SELECT \* FROM bookingsdb.booking;*

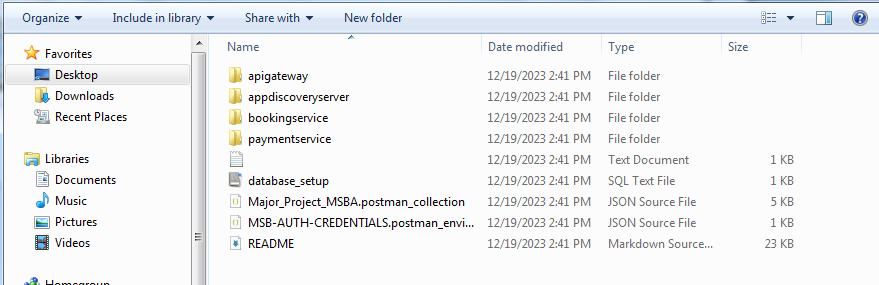
*SELECT \* FROM paymentsdb.transaction;*

******

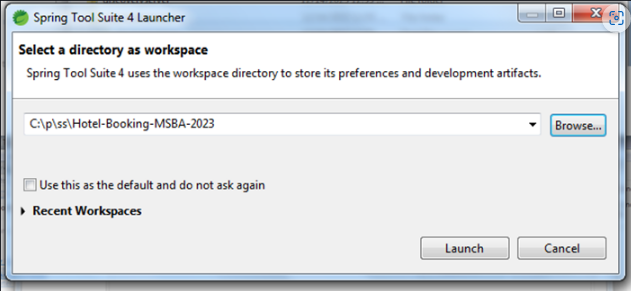
1. **Setup Your IDE (I am Using Spring Tool Suite) :**

*4 a). Local Structure*

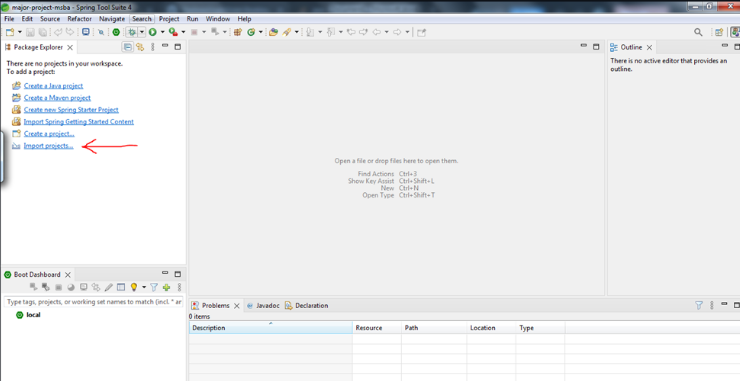
****

****

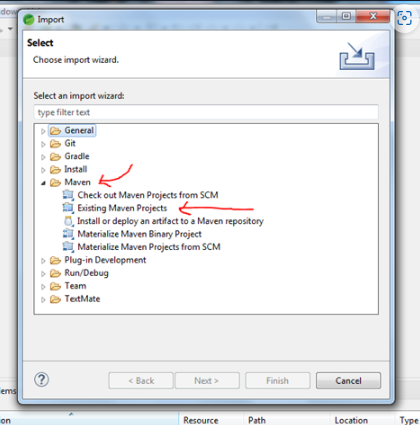
*4 b). Open IDE*

****

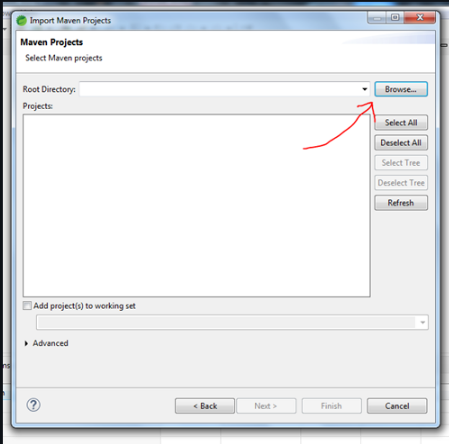
*4 c). Import Projects*



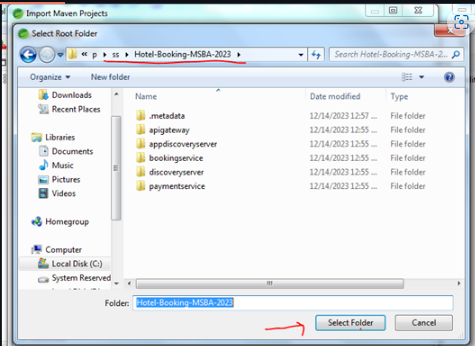
*4 d). Import Maven Projects*



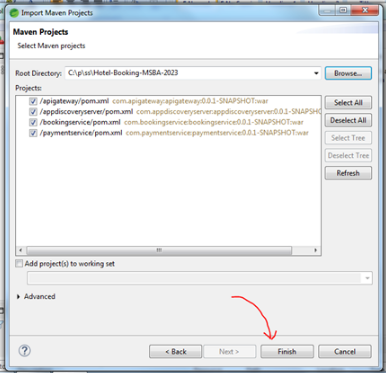
*4 e). Choose the directory of Project*

**

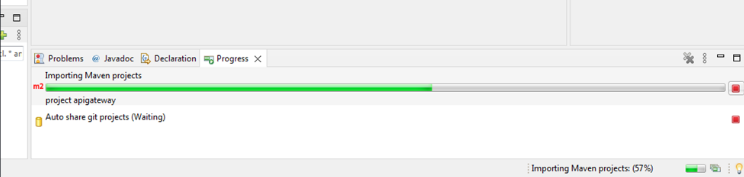
*4 f). Choose the root directory of Project*



*4 g). Import All the Projects*

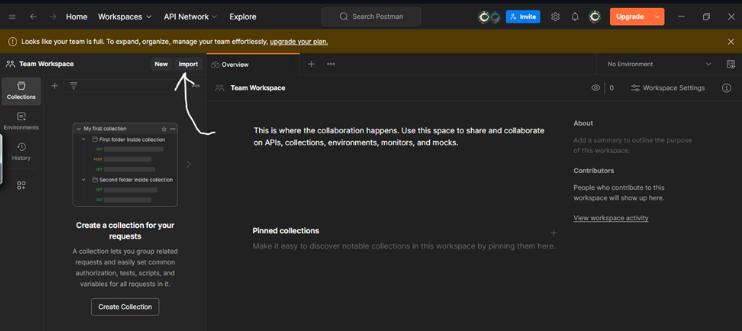


*4 h). Importing and Building will be done by IDE*

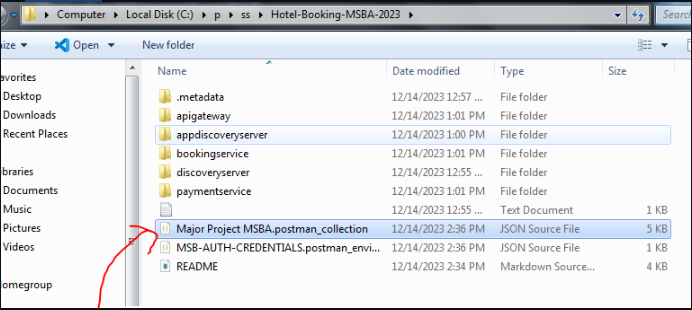


1. **Setup PostMan :**

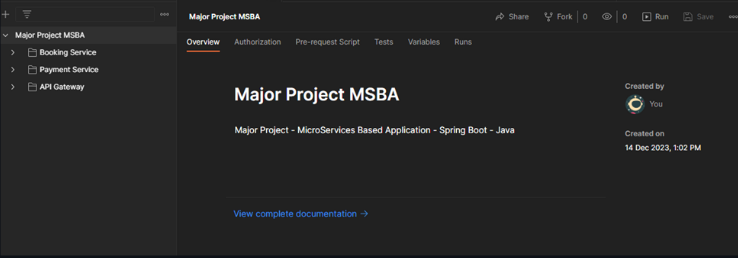
*5 a). Import the Collection File - Present in ReadMe Files Section*

**

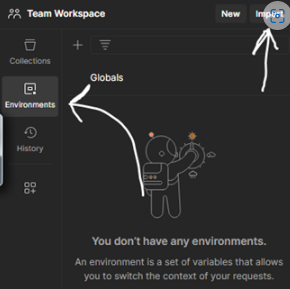
*5 b). Choose File*

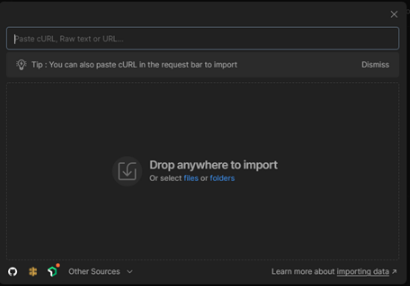
**

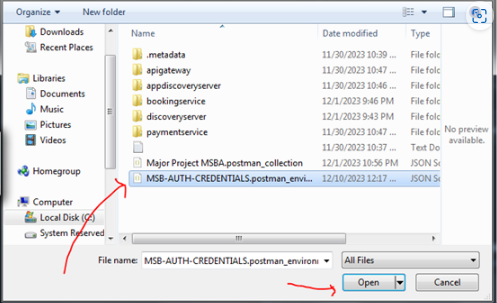
*5 c). Result*

**

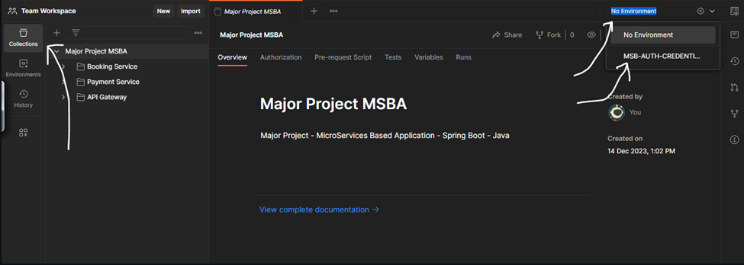
*5 d). Import Security Credentails*

**

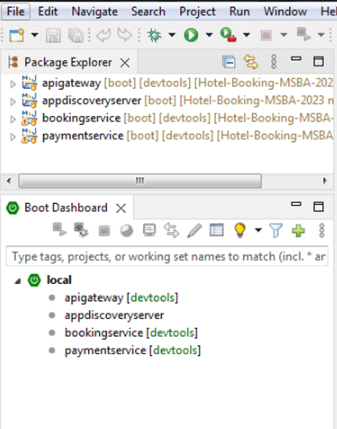
**

**

*5 e). Choose Proper Environment*

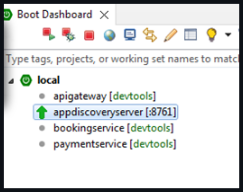
**

1. **Run the Application :**

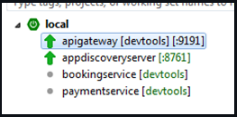
****

*6 a). Order to Run the Application ( Preventing Unknown Server Error)*

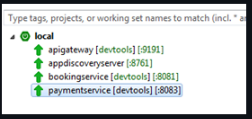
*6 a). Start Server*

**

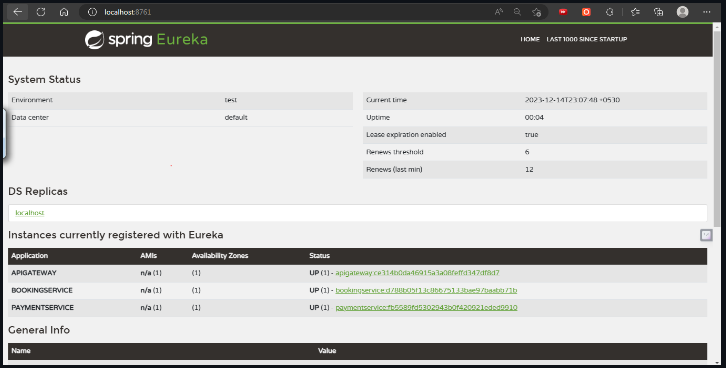
*6 b). Start Gateway*

**

*6 a). Start Both Services*

**

1. **Verify Succesful Start at *https://localhost:8761* :**

******

**6. Usage**

*The application provides RESTful APIs for managing hotels and bookings. You can use tools like curl or Postman to interact with the APIs.*

**7. Endpoint**

***7. Booking Service : [Booking Service](bookingservice)***

***7.1. Logic Class (****To Calculate Price and Room Numbers****) -***

***7.2. Model Classes*** *[BookingInfoEntity]*

***7.3. Controller -***

***Endpoint 1: POST /booking***

*•* ***URI:*** */booking*

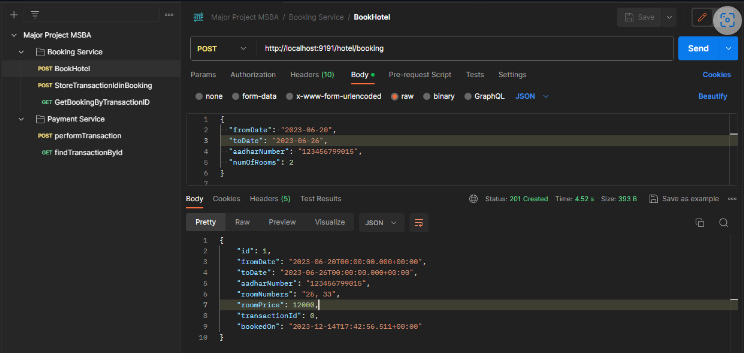
*•* ***HTTP METHOD:*** *POST*

*•* ***RequestBody****: fromDate, toDate,aadharNumber,numOfRooms*

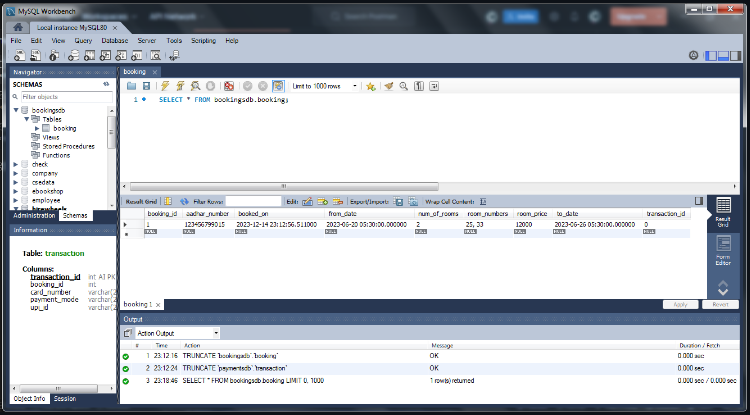
*•* ***Response Status****: Created*

*•* ***Response****: ResponseEntity<BookingInfoEntity>*

***Working***

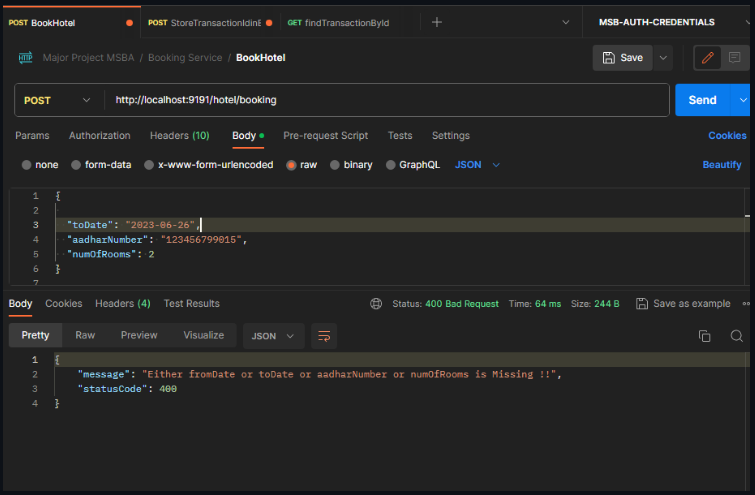
**

***Database***

**

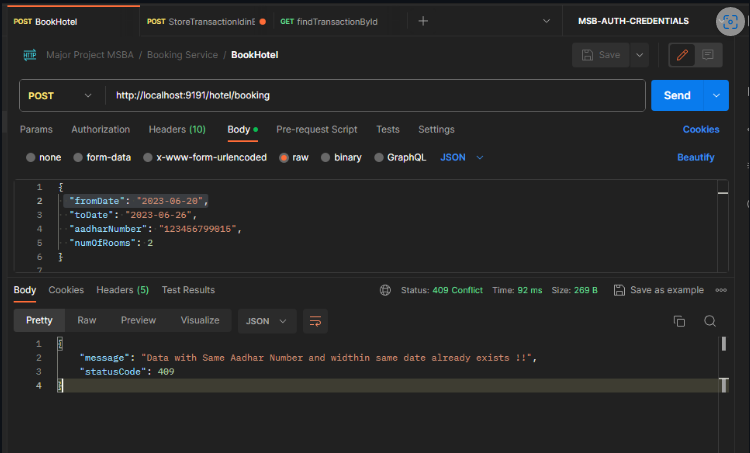
*------------------------------------------------------------------------------------------------------------------------------*

***Exception 1 - If Any Of One Field is Missing***

**

*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 2 - If Data within Same Date and AAdhar Number is Present***

**

*------------------------------------------------------------------------------------------------------------------------------------*

***Endpoint 2: POST booking/{bookingId}/transaction***

*•* ***URI:*** */booking/{bookingId}/transaction*

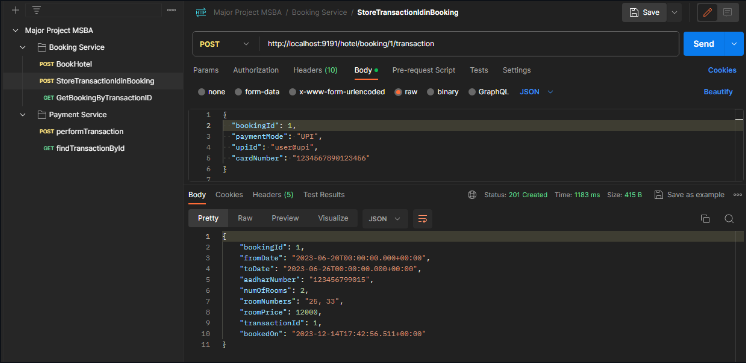
*•* ***HTTP METHOD****: POST*

*•* ***PathVariable*** *: int*

*•* ***RequestBody*** *: paymentMode, bookingId, upiId,cardNumber*

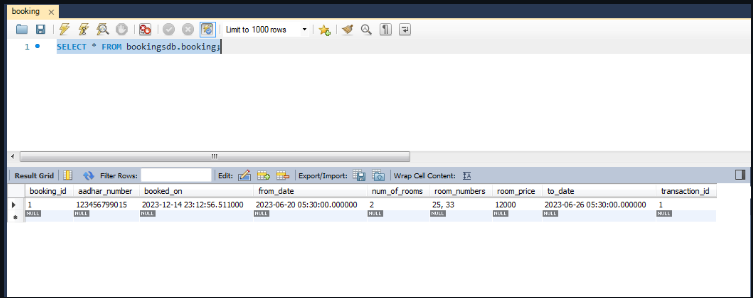
***• Response****: ResponseEntity<BookingInfoEntity>*

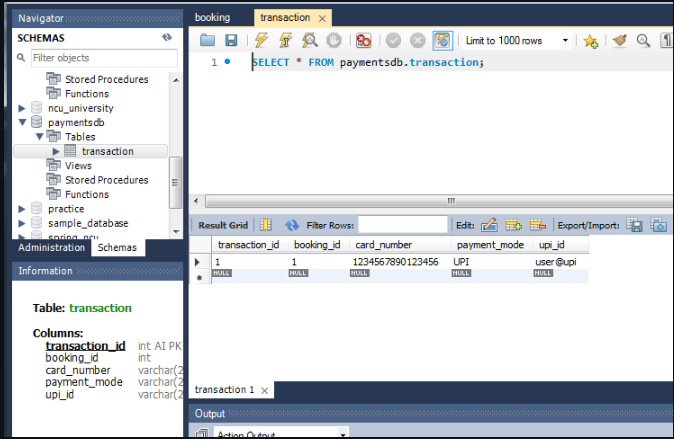
***Working***

**

*------------------------------------------------------------------------------------------------------------------------------------*

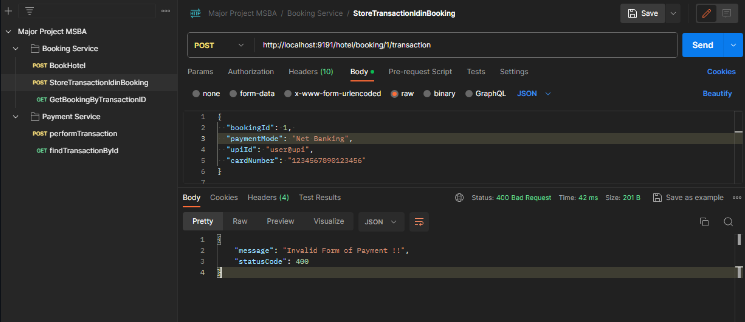
***Database***

**

**

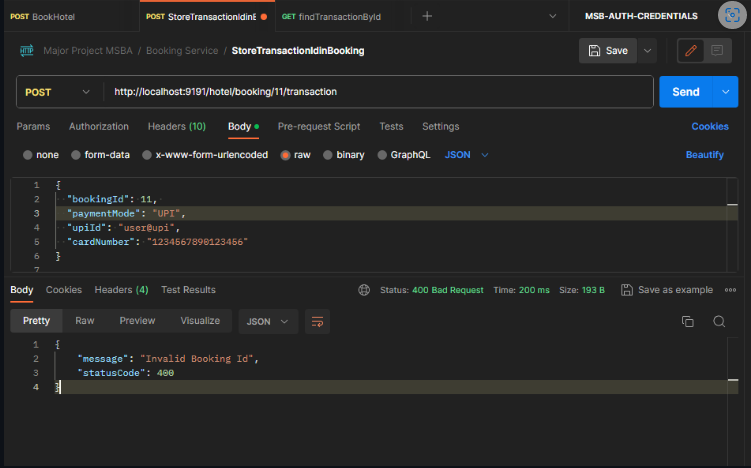
*----------------------------------------------------------------------------------------------------------------------------------*

***Exception 1 - If the user gives any other input apart from “UPI” or “CARD”***

**

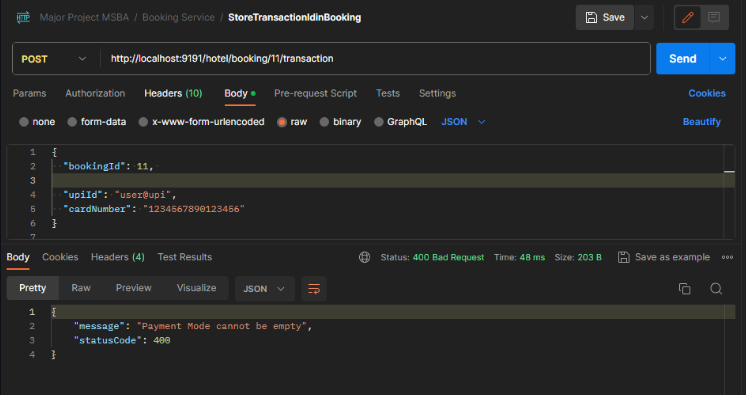
*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 2 - If no transactions exist for the Booking Id passed to this endpoint***

**

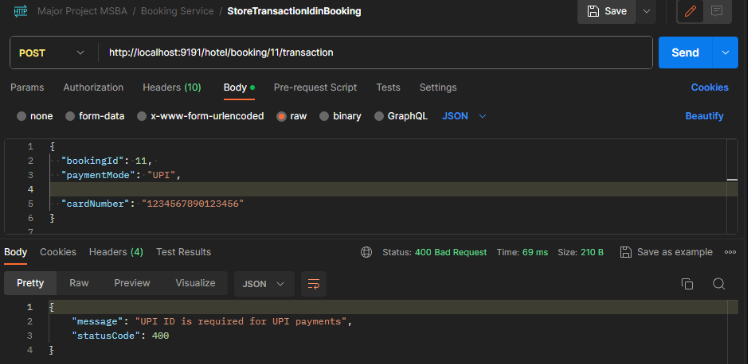
*------------------------------------------------------------------------------------------------------------------------------------*

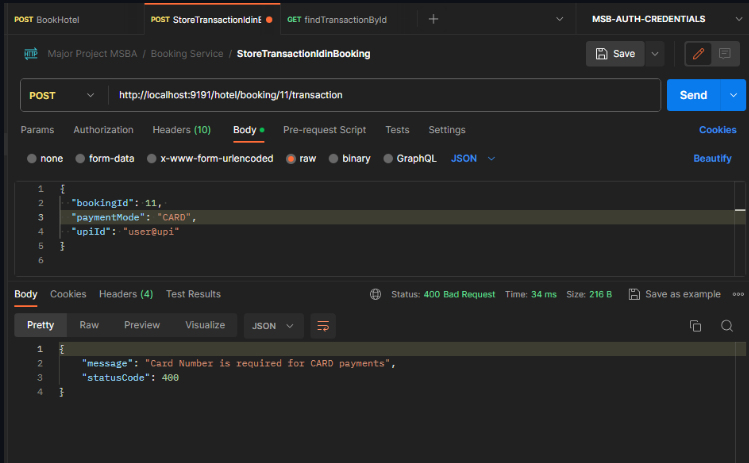
***Exception 3 - If no Payment form is given***

**

*------------------------------------------------------------------------------------------------------------------------------------*

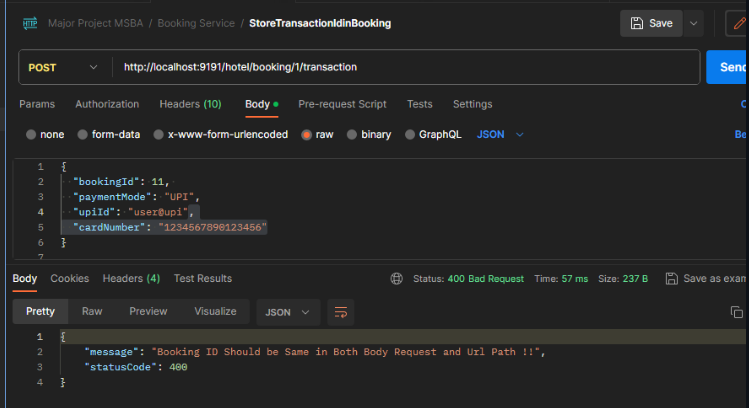
***Exception 4 - If no Payment details is given for payment form***

**

**

*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 5 - If Booking Id is different in Path Variable and Request Body***

**

*------------------------------------------------------------------------------------------------------------------------------------*

***Endpoint 3: GET hotel/getBookingByTransId/{transactionId}***

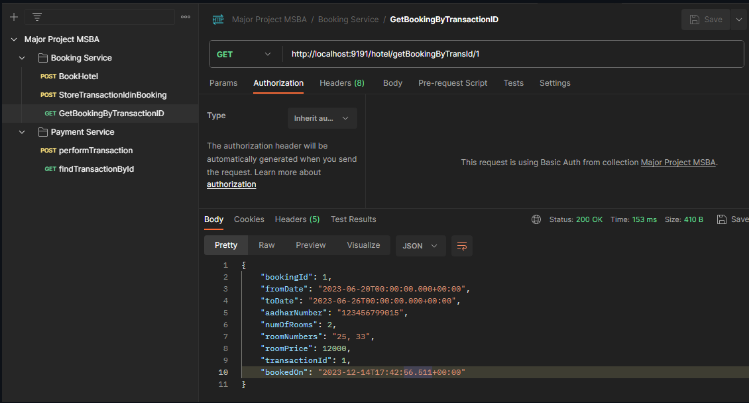
*•* ***URI:*** *hotel/getBookingByTransId/{transactionId}*

*•* ***HTTP METHOD:*** *GET*

*•* ***PathVariable*** *: int*

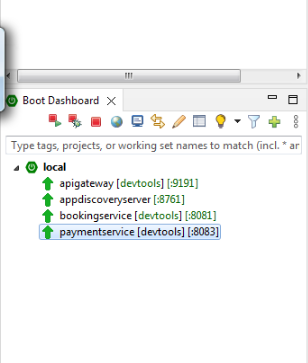
*•* ***Response****: ResponseEntity<BookingInfoEntity>*

***Working***

**

*-----------------------------------------------------------------------------------------------------------------------------*

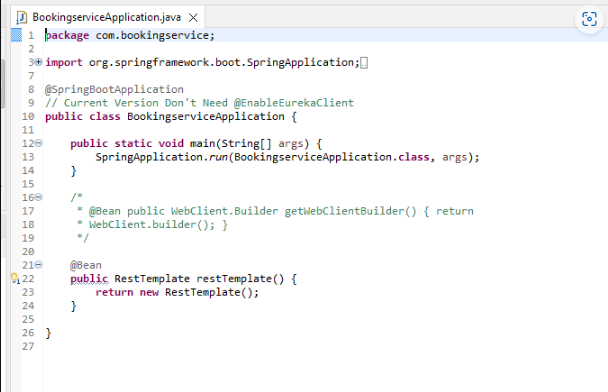
***1.3 Configure this service to run on port number 8081.***

**

*---------------------------------------------------------------------------------------------------------------------------------*

***1.4 Configure the hotel booking service as Eureka Client***

***[MainFile](bookingservice/src/main/java/com/bookingservice/BookingserviceApplication.java)***

**

*------------------------------------------------------------------------------------------------------------------------------------*

***2. Payment Service : [Payment Service](paymentservice)***

***2.1. Model Classes -******[TransactionDetailsEntity](paymentservice/src/main/java/com/paymentservice/entity/TransactionDetailsEntity.java)***

***2.2. Controller -***

***Note: This endpoint will be called by the ‘endpoint 2’ of the Booking service***

***Endpoint 1: POST /transaction***

*•* ***URI****: /transaction*

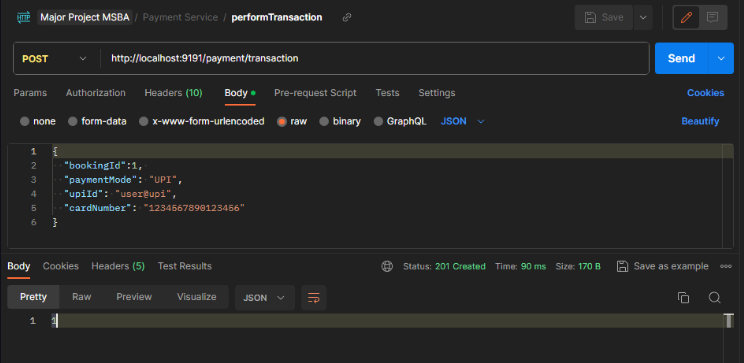
*•* ***HTTP METHOD****: POST*

*•* ***RequestBody****: paymentMode, bookingId, upiId, cardNumber*

*•* ***Response Status****: Created*

***• Response:*** *ResponseEntity<transactionId>*

***Working***

**

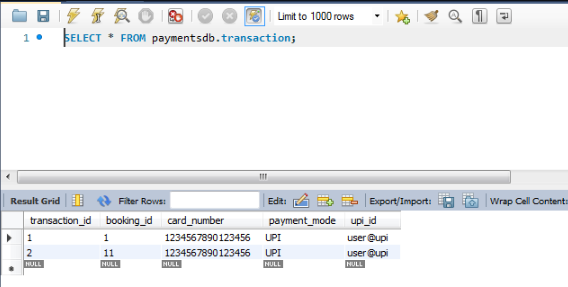
*------------------------------------------------------------------------------------------------------------------------------------*

***Console Message***

**

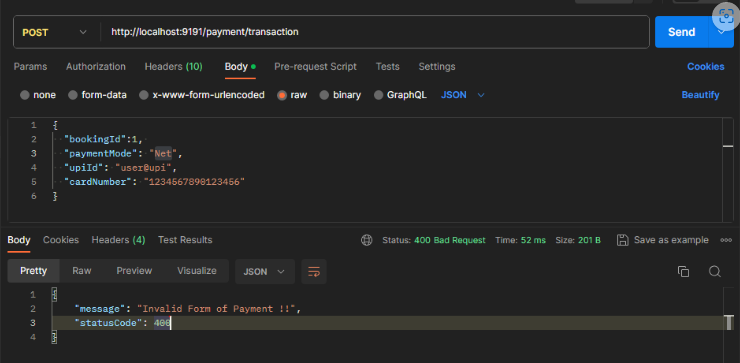
*------------------------------------------------------------------------------------------------------------------------------------*

***Database***

**

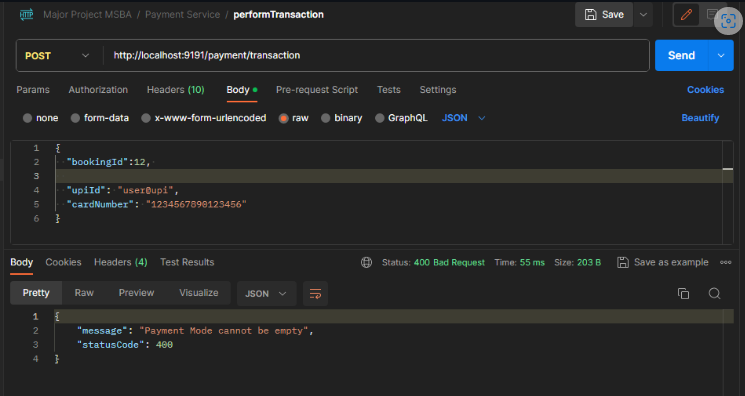
*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 1 - If the user gives any other input apart from “UPI” or “CARD”***

**

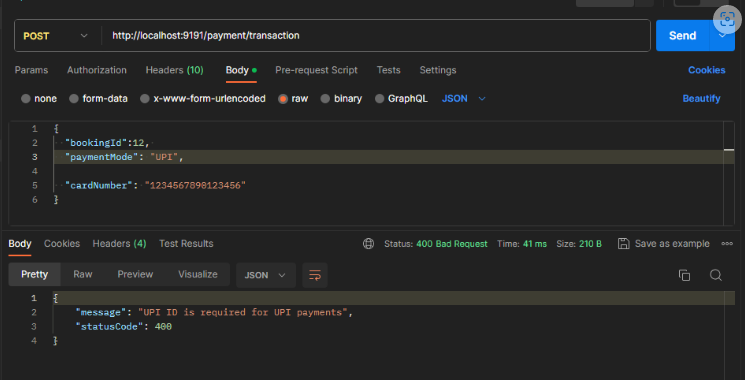
*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 2 - If no Payment form is given***

**

*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 3 - If no Payment details is given for payment form***

**

*------------------------------------------------------------------------------------------------------------------------------------*

***Endpoint 2: GET /transaction/{transactionId}***

*•* ***URI:*** */transaction/{transactionId}*

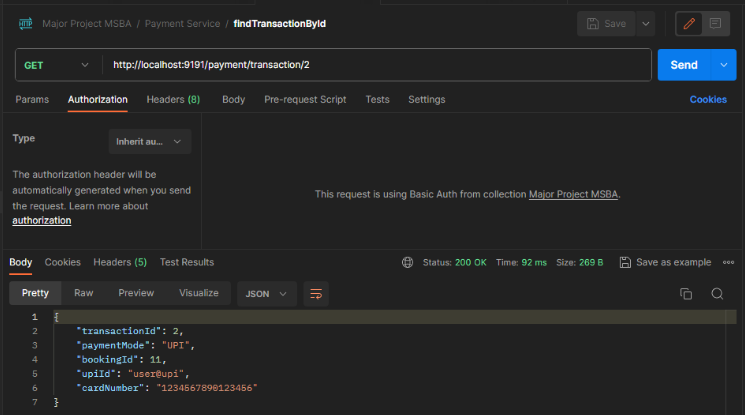
*•* ***HTTP METHOD:*** *GET*

*•* ***Path Variable****: int*

*•* ***Response Status****: OK*

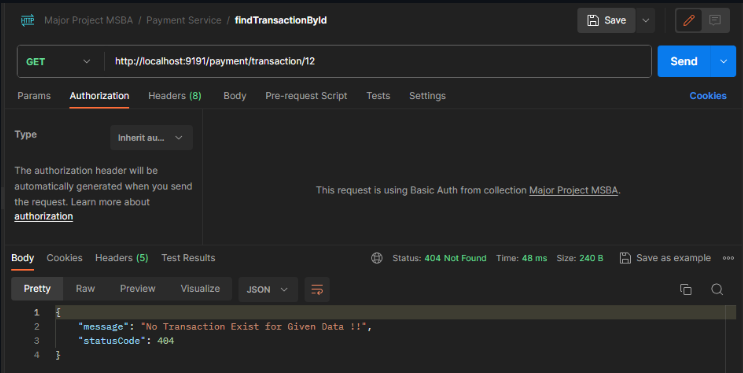
*•* ***Response:*** *ResponseEntity<TransactionDetailsEntity>*

***Working***

**

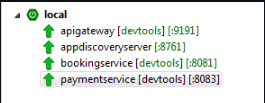
*------------------------------------------------------------------------------------------------------------------------------------*

***Exception 1 - If transaction Not Exists***

**

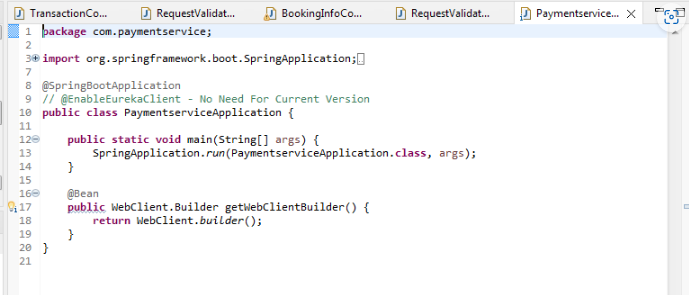
*------------------------------------------------------------------------------------------------------------------------------------*

***2.3 Configure this service to run on port number 8083.***

**

*------------------------------------------------------------------------------------------------------------------------------------*

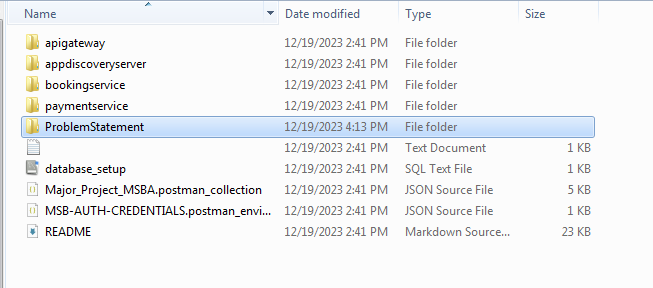
***2.4 Configure the service as a Eureka client***

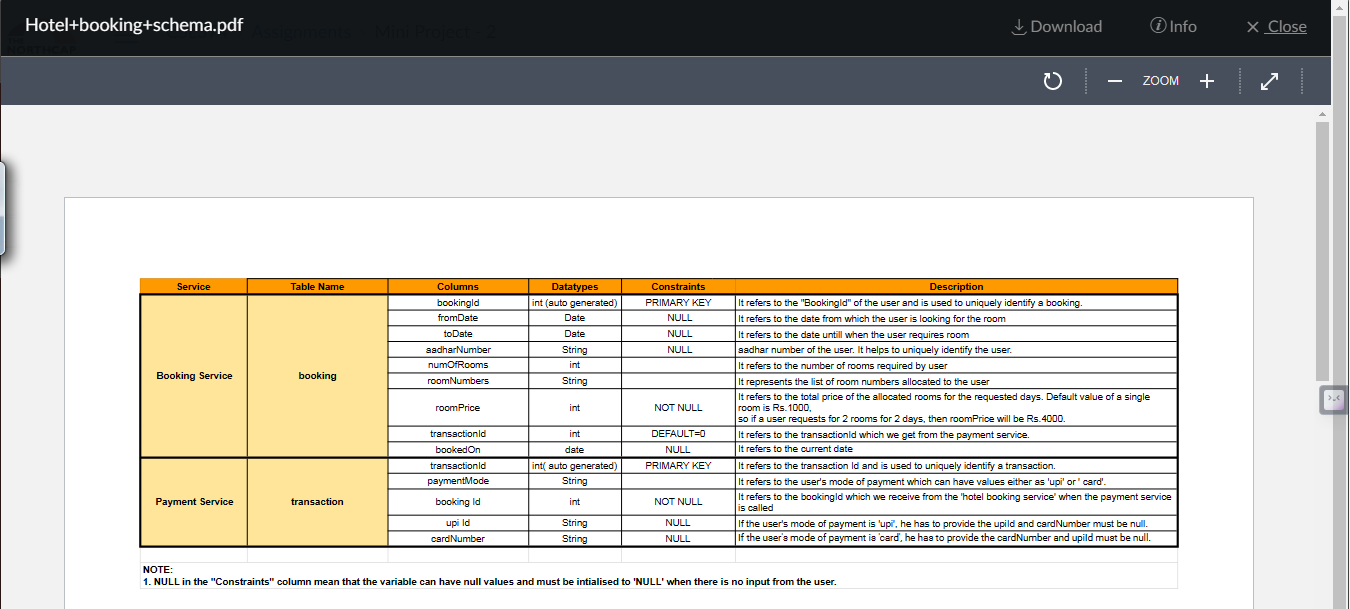
**

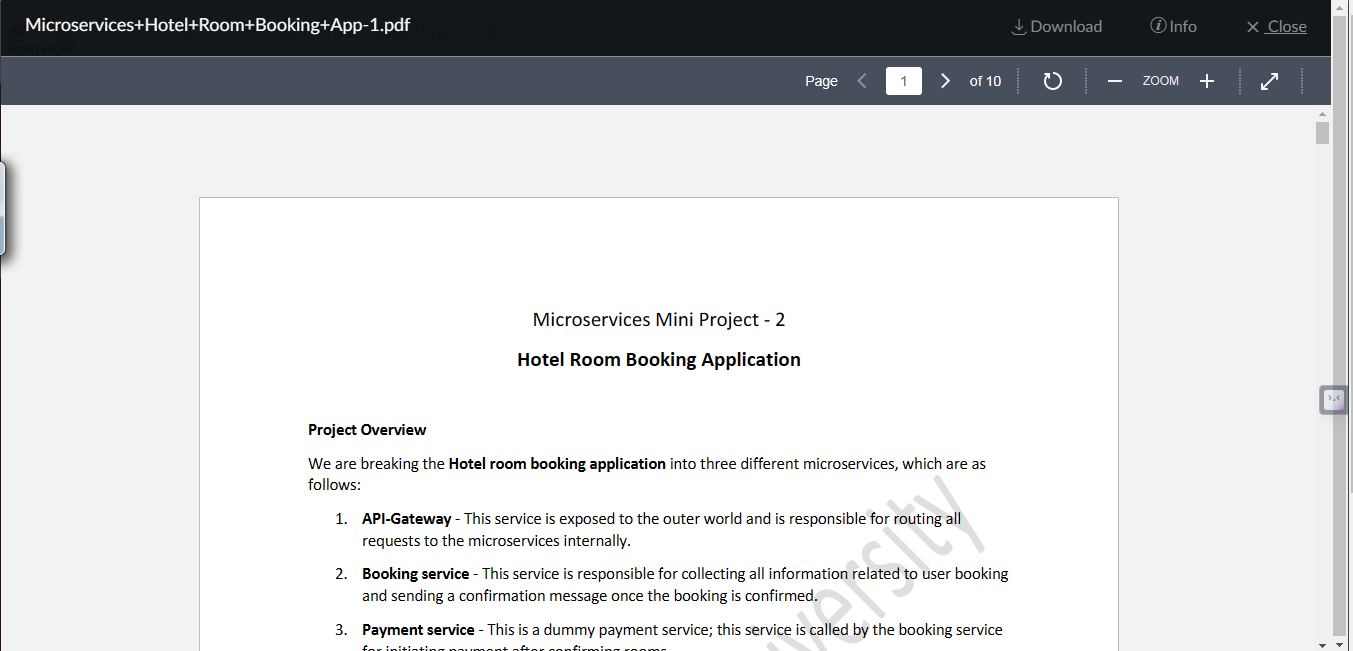
*------------------------------------------------------------------------------------------------------------------------------------*

**8. Documentation Reference**

**8.1 -> All Documents are attached in ProblemStatement Folder**

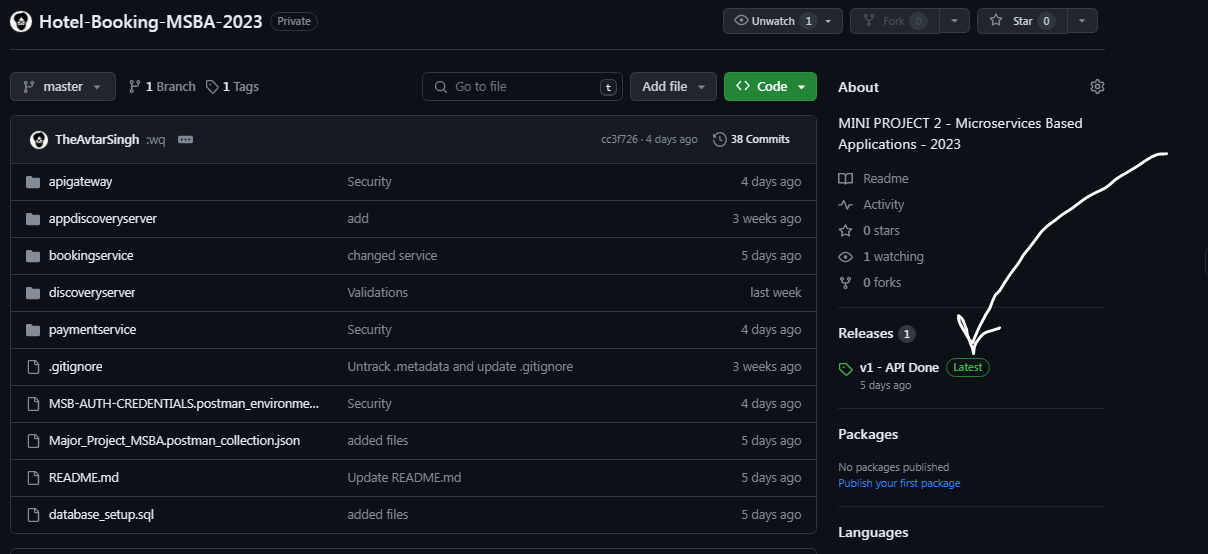
****

**

**

**9. ChangeLog**

**9.1 Current Version is in production only - v1**

****

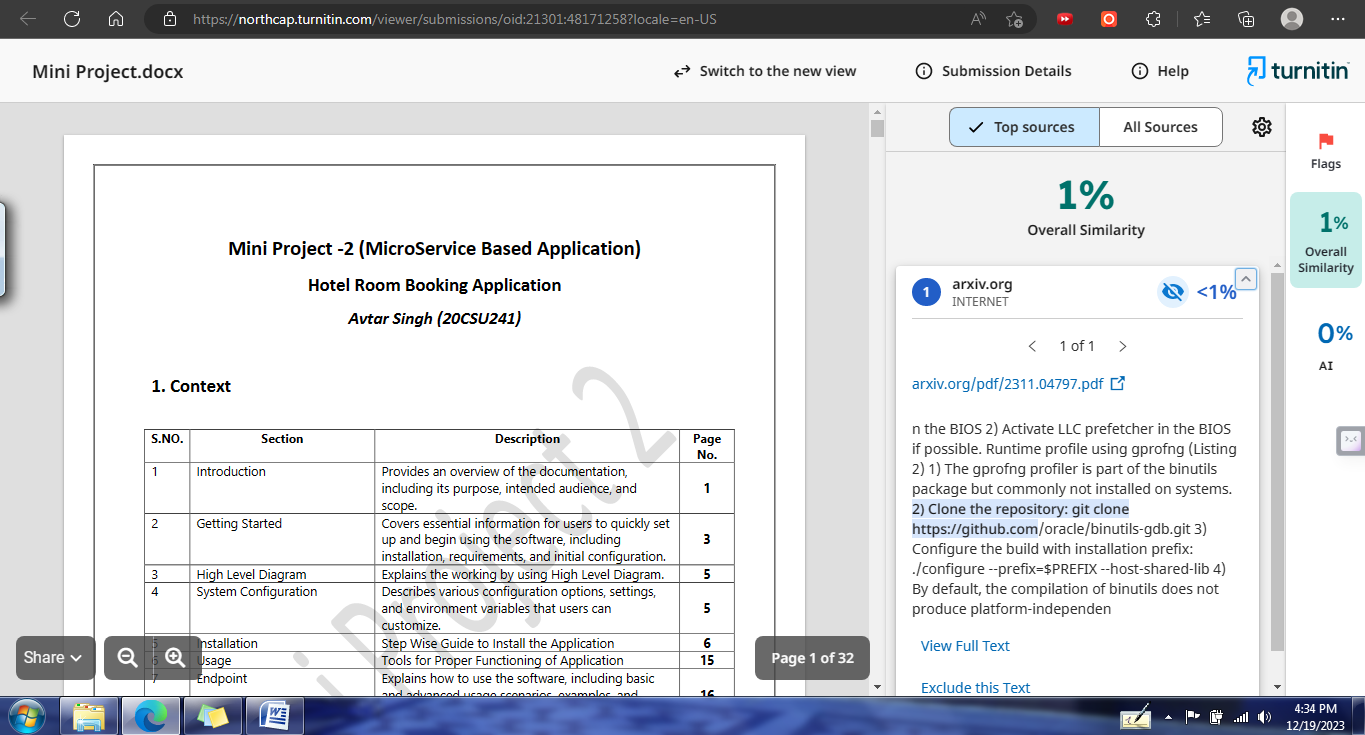
**10. License**

*This project is Copyrighted to* ***@TheAvtarSingh* [https://theavtarsingh.github.io]**.

**11. References and Links**

* **GITHUB - Hotel-Booking-MSBA-2023 -** https://github.com/TheAvtarSingh/Hotel-Booking-MSBA-2023
* **POSTMAN ENVIROMENT FILE -** https://github.com/TheAvtarSingh/Hotel-Booking-MSBA-2023/blob/master/Major\_Project\_MSBA.postman\_collection.json
* **POSTMAN CREDENTAILS FILE -** https://github.com/TheAvtarSingh/Hotel-Booking-MSBA-2023/blob/master/MSB-AUTH-CREDENTIALS.postman\_environment.json
* **MYSQL DATA FILE -** https://github.com/TheAvtarSingh/Hotel-Booking-MSBA-2023/blob/master/database\_setup.sql
* **PROJECT README FILE -** https://github.com/TheAvtarSingh/Hotel-Booking-MSBA-2023/blob/master/README.md

**12. Turnitin Plagiarism Report**

****

*- • -*

*End of Documentation*

*- • -*