**Microservices assignment 2023**

**Booking System**

**Table of contents**

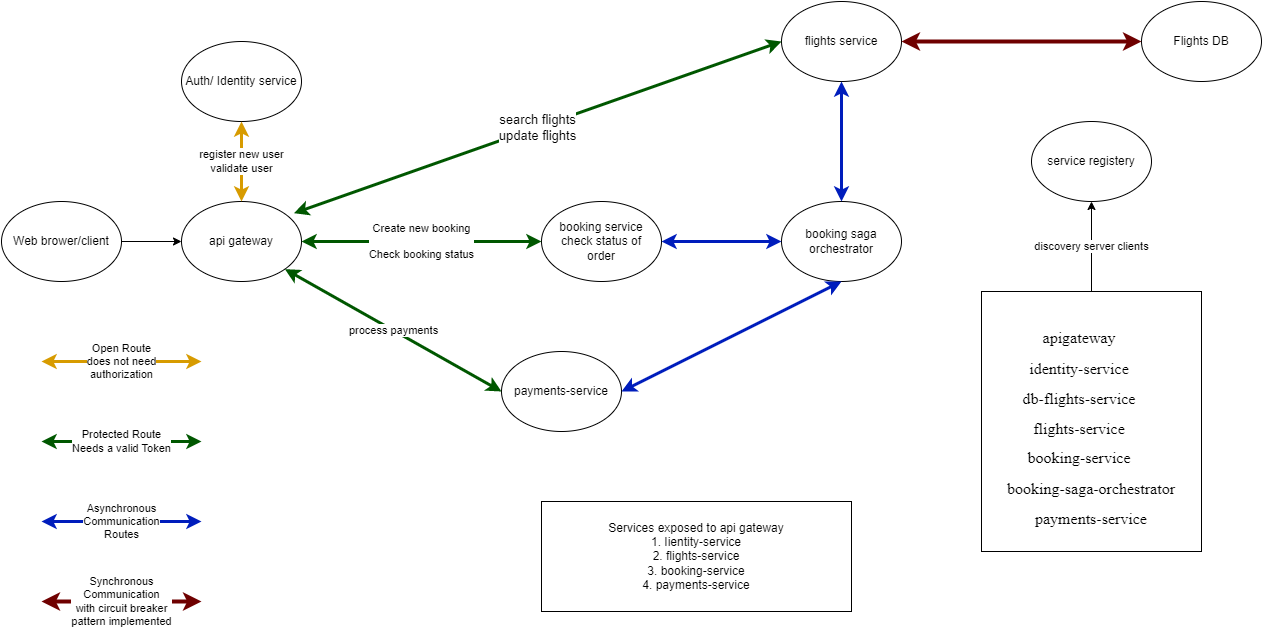
**Flight Booking**

**Services identified for flight booking system**

1. discovery-server
2. apigateway
3. identity-service
4. db-flights-service
5. flights-service
6. booking-service
7. booking-saga-orchestrator
8. payments-service

The above mentioned services are elaborated in next sections.

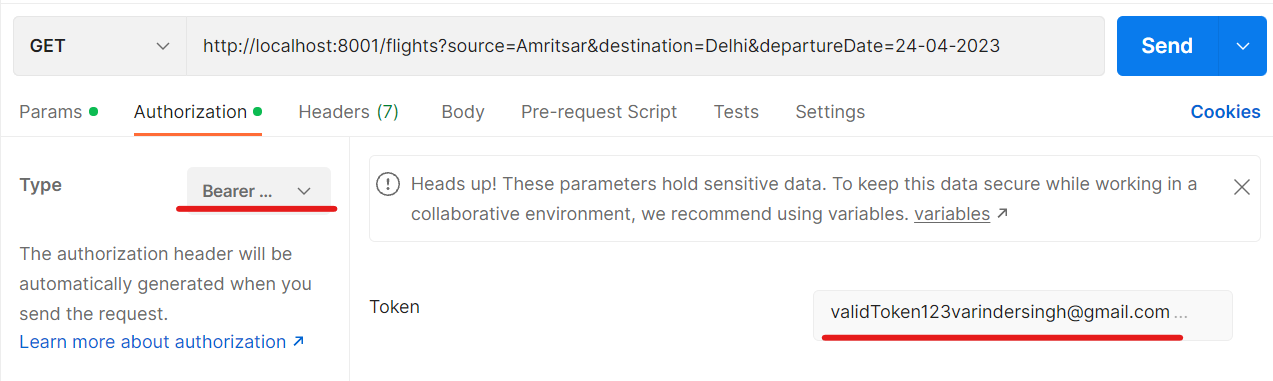
**A data flow diagram of the proposed system is shown below**

****

**Assumptions made for the above identified services.**

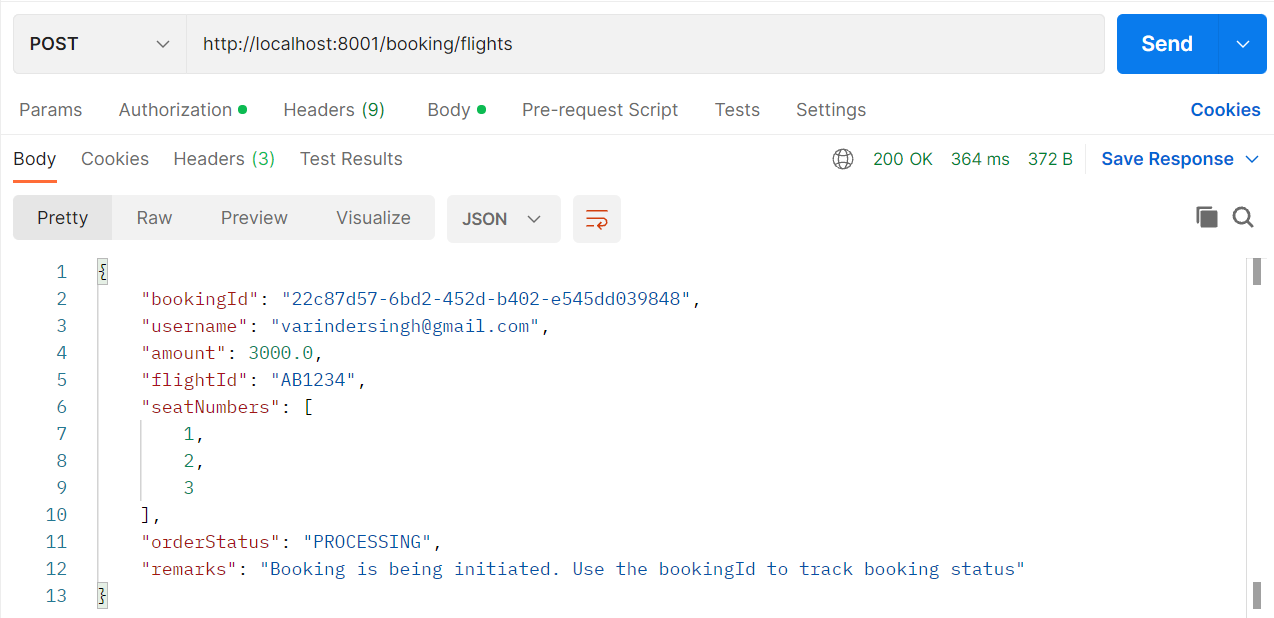
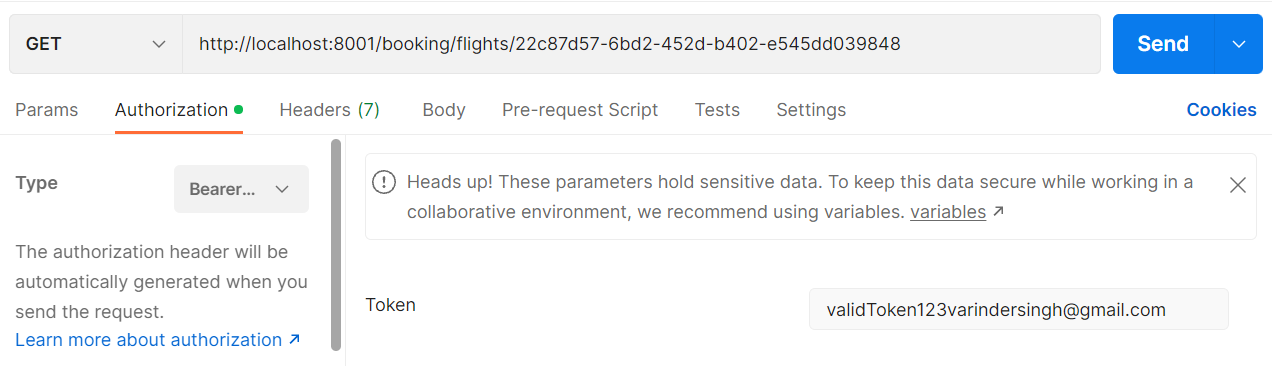
1. identity-service :  
   This service will perform users related tasks like  
   a. registering new user  
   b. logging in the user and providing a valid token
2. db-flights-service :  
   This service will act as a database for all the flights details. It will provide the functionality to  
   a. read flights data  
   b. update flights data
3. flights-service :  
   This service is created to consume the db-flights-service such that the db-flights-service is not exposed publicly. Because the user may be able to directly modify the flights data in database if db-flights-service is exposed to api-gateway/publicly.  
   This service uses circuit breaker pattern to consume db-flights-service and activemq to communicate asynchronously.  
   This service provide functionality like  
   a. view flights where we can filter flights using source, destination, departure date.  
   b. view complete data for a specific flight
4. booking-service :  
   This service is created to create a booking order for a user. This service will get a post request for creating a booking. Then it will initiate the booking and will dispatch various events like check seat availability, payment request etc. to keep the booking order up to date.
5. booking-saga-orchestrator :  
   This service is based on the saga pattern used in microservices to carry out the asynchronous requests in a systematic order.  
   This service manages the events between booking-service, payments-service, flights-service.
6. payments-service :  
   This service is created to mock the payment requests in the booking transactions.

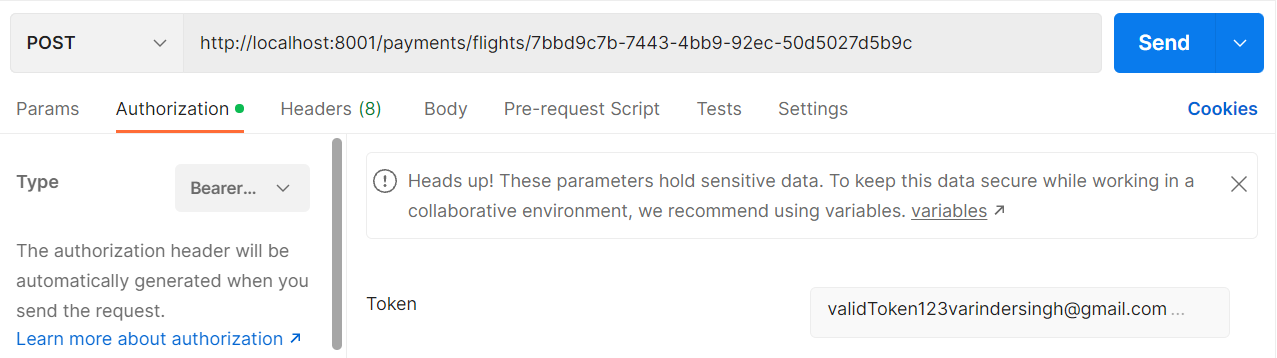
**Description and usage of various microservices implemented.**

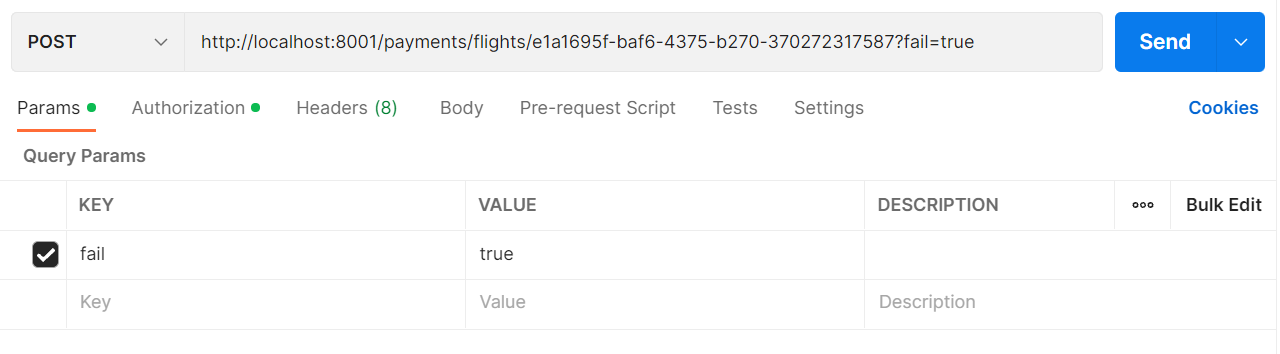
1. **Api gateway :**This service is used to map endpoints of different microservices under a single domain.  
     
   Base Uri : <http://localhost:8001>  
   Various microservices are mapped to api gateway as show below  
   1. **Register a new user ( identity-service)**  
      Uri : <http://localhost:8001/users>  
      Method : Post  
      All fields are required  
      Request Body : JSON  
      Result :  
      
   2. **Login user (identity service)**  
      Uri : <http://localhost:8001/users/login>   
      Method : Post  
      All fields are required  
      Request Body : JSON  
      Result :  
      
   3. **Search Flight (filghts-service)**Uri : http://localhost:8001/flights?source=Amritsar&destination=Delhi&departureDate=24-04-2023  
      Method : Get  
      Authentication is Required, kindly provide valid token under Bearer Token in Postman.  
        
      Query params (all query params are optional, if not provided all flights will be shown)  
      1. source = name of the source city  
      2. destination = name of the destination city  
      3. departureDate = date of travel must be in DD-MM-YYY format  
        
      Request  
        
        
        
        
        
        
        
        
        
        
        
      Result  
      Graphical user interface, text, application, email

      Description automatically generated
   4. **Search a specific flight (flights-service)**  
      Uri : [http://localhost:8001/flights/{flightd}](http://localhost:8001/flights/%7bflightd%7d)  
      Method : Get  
      Authentication is Required, kindly provide valid token under Bearer Token in Postman.  
        
      Path Variable (required)  
      1. flightId = Flight Id of a flight.  
        
      Request  
      Graphical user interface, text, application

      Description automatically generated  
        
        
        
        
        
        
        
        
        
        
        
        
        
        
        
        
      Response  
      Graphical user interface, text, application, email

      Description automatically generated
   5. **Book a Flight (booking-service)**Uri : http://localhost:8001/booking/flights  
      Method : Post  
      Authentication is Required, kindly provide valid token under Bearer Token in Postman.  
        
      Request Body : Json (All fields required)  
      1. username = username of the user booking flight  
      2. flightId = Flight Id of the flight to be booked  
      3. seatNumbers = Array of seat numbers user wish to book. (seats available 1 to 30)   
      Response :  
      
   6. **Check Booking status (booking-service)**Uri : http://localhost:8001/booking/flights/{bookingId}  
      Method : Get  
      Authentication is Required, kindly provide valid token under Bearer Token in Postman.  
        
      Path Variable (required)  
      1. bookingId = bookingId is unique id to specify the booking order.  
        
      Request:  
        
        
      Response:  
      Graphical user interface, text, application

      Description automatically generated
   7. **Send Payment (payment-service)**It is payment mocking service and can be used to perform two tasks  
      1. To mock payment success  
      2. To mock payment failure  
        
      **I. Payment Success Mocking**Uri : http://localhost:8001/payments/flights/{bookingId}   
      Method : Post  
      Authentication is Required, kindly provide valid token under Bearer Token in Postman.  
        
      Path Variable (required)  
      1. bookingId = bookingId is unique id to specify the booking order.  
        
        
        
        
        
        
        
        
        
      Request:  
        
        
      Response:  
      **Graphical user interface, text, application

      Description automatically generated**  
        
        
      2. **Payment Failure Mocking**Uri : http://localhost:8001/payments/flights/{bookingId}?fail=true   
      Method : Post  
      Authentication is Required, kindly provide valid token under Bearer Token in Postman.  
        
      Path Variable (required)  
      1. bookingId = bookingId is unique id to specify the booking order.  
        
      To make the payment fail add the following query parameter in the URI.  
      Query Param (Required)  
      1. fail=true  
        
      Request:  
        
        
      Response:  
      Graphical user interface, text, application

      Description automatically generated  
       **To Be Continued…**