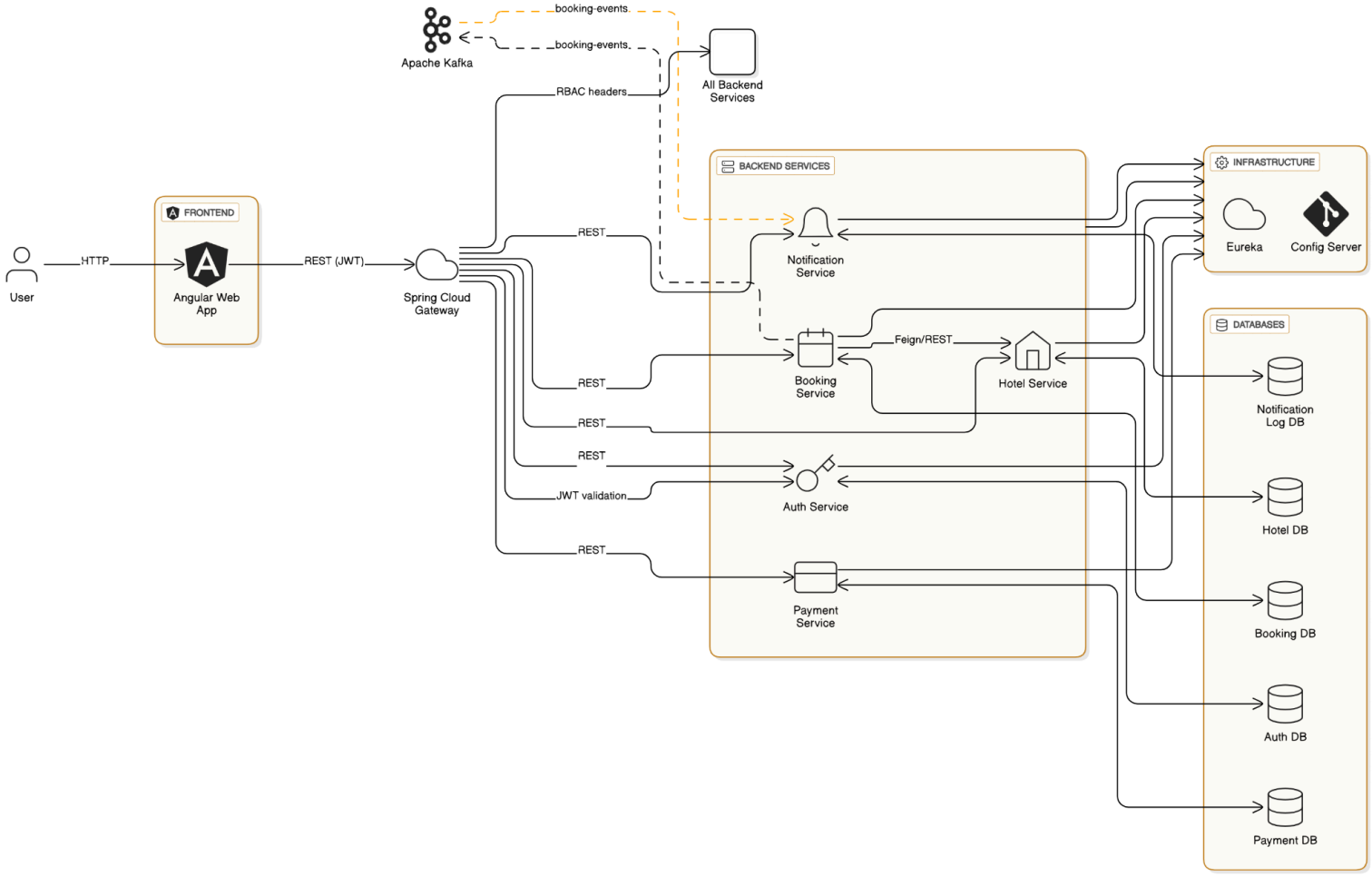
Hotel Reservation System - Microservices Architecture

ARCHITECTURE DIAGRAM:



Core Infrastructure Services (Mandatory):

1. Service Registry (Eureka) - Service discovery
2. Config Server - Centralized configuration
3. API Gateway - Single entry point implementing RABC

Business Microservices:

1. Auth Service (8081) - Users & Authentication
2. Hotel Service (8082) - Hotels, Rooms, Inventory
3. Booking Service (8083) - Booking + Check-in/Check-out
4. Payment Service (8084) - Invoices + Payments + Extra Charges
5. Notification Service (8085) - Communications
6. Analytics Service (8086) - Reports & Dashboards

DATABASE SCHEMA

1. Auth Service

USER DATABASE

ATTRIBUTE	TYPE	DESCRIPTION
id	Long	Primary Key
username	String	Unique login name
email	String	Unique email
password	String	Encrypted password
firstName	String	User first name
lastName	String	User last name
phone	String	Contact number
roles	Set<Role>	Assigned user roles
hotelIds	Set<Long>	Linked hotel IDs

2. Hotel Service

HOTEL DATABASE

ATTRIBUTE	TYPE	DESCRIPTION
id	Long	Primary Key
name	String	Hotel name
address	String	Hotel address
city	String	Hotel city
state	String	Hotel state
country	String	Hotel country
starRating	Integer	Star classification
active	Boolean	Active status
createdAt	LocalDateTime	Creation timestamp
updatedAt	LocalDateTime	Last update timestamp

ROOM DATABASE

ATTRIBUTE	TYPE	DESCRIPTION
id	Long	Primary Key
hotel	Hotel	Linked hotel
category	RoomCategory	Room category
roomNumber	String	Room identifier
status	RoomStatus	Availability status
active	Boolean	Active flag
createdAt	LocalDateTime	Creation timestamp
updatedAt	LocalDateTime	Last update timestamp

ROOM CATEGORIES DATABASE

ATTRIBUTE	TYPE	DESCRIPTION
id	Long	Primary Key
hotel	Hotel	Linked hotel
name	String	Category name
capacity	Integer	Guest capacity
description	String	Category details
active	Boolean	Active flag
createdAt	LocalDateTime	Creation timestamp
updatedAt	LocalDateTime	Last update timestamp

3. Booking Service

BOOKING DATABASE

ATTRIBUTE	TYPE	DESCRIPTION
id	Long	Primary Key
userId	Long	Booking user ID
hotelId	Long	Hotel reference
roomId	Long	Room reference
checkInDate	LocalDate	Check-in date
checkOutDate	LocalDate	Check-out date
totalAmount	BigDecimal	Booking price
bookingStatus	BigDecimal	Booking state
createdAt	LocalDateTime	Creation time
updatedAt	LocalDateTime	Last update time

4. Payment Service

PAYMENT DATABASE

ATTRIBUTE	TYPE	DESCRIPTION
id	Long	Primary Key
bookingId	Long	Linked booking ID
userId	Long	Payment owner
amount	BigDecimal	Payment amount
status	PaymentStatus	Payment state
createdAt	LocalDateTime	Creation timestamp
updatedAt	LocalDateTime	Last update timestamp

API ENDPOINTS:

1. Auth Service (Port 8081)

HTTP METHOD	ENDPOINT	DESCRIPTION	PAYLOAD

POST	/api/auth/register	Register new guest user	{
			"username":"Debashrita123",
			"email": "debashrita@email.com",
			"password": "password123",
			"firstName": "Debashrita",
			"lastName": "Mandal",
			"phone": "+1234567890"
			}

POST	/api/auth/login	Authenticate user and return JWT token	{
			"username": "debashrita",
			"password": "password123"
			}

GET	/api/auth/validate	Validate JWT token (used by API Gateway)	

POST	/api/user	Create user with specific role (Admin only)	{
			"username": "receptionist_01",
			"email": "receptionist01@hotel.com",
			"password": "Receptionist@123",
			"firstName": "Hotel",
			"lastName": "Receptionist",
			"phone": "9876541234",
			"role": "ROLE_RECEPTIONIST",
			"hotelIds": [1]
			}

2. Hotel Service (Port 8083)

HTTP METHOD	ENDPOINT	DESCRIPTION	PAYLOAD

POST	/api/hotels	Create new hotel property	{
			"name": "Grand Hotel",
			"address": "RG Road, Near Star Mall",
			"city": "Bengaluru",
			"state": "Karnataka",
			"country": "India",
			"starRating": 5
			}

GET	/api/hotels	List all hotels	

POST	/api/hotels/{hotelId}/categories	Create room category for hotel	{
			"name": "Deluxe",
			"capacity": 2,
			"description": "Deluxe room with balcony and city view"
			}

GET	/api/hotels/{hotelId}/categories	List room categories for hotel	

POST	/api/hotels/{hotelId}/rooms	Add rooms to inventory	{
			"roomNumber": "101"
			}

GET	/api/hotels/{hotelId}/rooms	Get rooms	
PUT	/api/hotels/{hotelId}/rooms/{id}/status	Update the room status	

POST	/api/categories/{categoryId}/pricing	Set base pricing	{
			"basePrice": 4500.00,
			"currency": "INR"
			}

GET	/api/categories/{categoryId}/pricing	Get the base price	
POST	/api/categories/{categoryId}/se	Set the seasonal	{ "startDate": "2025-12-20",

	asonal-pricing	pricing	"endDate": "2025-12-31", "price": 6500.00 }
GET	/api/categories/{categoryId}/seasonal-pricing	Get the seasonal pricing	

3. Booking Service (Port 8084)

HTTP METHOD	ENDPOINT	DESCRIPTION	PAYLOAD

POST	/api/booking	Create new booking	{
			"hotelId": 1,
			"roomId": 1,
			"checkIn": "2025-12-20",
			"checkOut": "2025-12-22"
			}

GET	/api/booking/{id}	List the bookings by id	
GET	/api/booking/user/{userId}	List all booking by user	
GET	/api/bookings	List all bookings (for admin)	
DELETE	/api/bookings/{id}	Delete booking by booking Id	

4. Payment Service (Port 8085)

HTTP METHOD	ENDPOINT	DESCRIPTION	PAYLOAD
POST	/api/payments/{bookingId}	Initiate the payment	
PUT	/api/payments/{bookingId}/pay	Complete the payment to get status as PAID	
GET	/api/payments/booking/{bookingId}	Get the payment status using bookingId	

5. Report Service (Port 8086)

HTTP METHOD	ENDPOINT	DESCRIPTION	RESULT PAYLOAD

POST	api/reports/revenue	Get the revenue report	{
			"totalRevenue": 13000.00,
			"currency": "INR"
			}

PUT	/api/reports/booking/summary	Get the booking summary report	{
			"totalBookings": 3,
			"confirmedBookings": 1,
			"cancelledBookings": 2
			}

Business Rules

1. Authentication and Authorization

- All users must authenticate using valid credentials, and every protected request must include a valid JWT token.
- Role-Based Access Control (RBAC) is enforced to restrict system actions based on user roles.
- Admin users have full access, while Managers, Receptionists, and Guests have limited role-specific permissions.
- Unauthorized or insufficiently authorized requests are rejected by the system.

---

2. Hotel and Room Management

- Hotels and rooms can be created, updated, or disabled only by Admin users.
- Each room is associated with exactly one hotel and one room category.
- Room categories define capacity and base pricing for rooms.
- Rooms marked under Maintenance are excluded from availability and booking.

---

3. Room Search and Availability

- Guests can search rooms using filters such as city, date range, capacity, and price.
- Only rooms available for the complete selected date range are displayed.
- Availability checks are performed both during search and booking creation.
- The system strictly prevents double booking of rooms.

---

4. Booking and Reservation Lifecycle

- Only authenticated users are allowed to create bookings with valid check-in and check-out dates.
- Users can view or cancel only their own bookings, while Admins can view all bookings.
- Each booking follows a controlled lifecycle from booking to completion or cancellation.

- Invalid booking status transitions are not allowed.

---

5. Check-in and Check-out Operations

- Check-in is allowed only on or after the scheduled check-in date.
- Check-in and check-out are handled by Managers or Receptionists.
- During check-in, booking and room statuses are updated accordingly.
- During check-out, rooms are released and made available for future bookings.

---

6. Pricing, Billing, and Payments

- Booking price is calculated based on room pricing and length of stay.
- Seasonal pricing may be applied when configured by Admin users.
- Each booking is linked to exactly one payment record.
- A booking is confirmed only after successful payment.

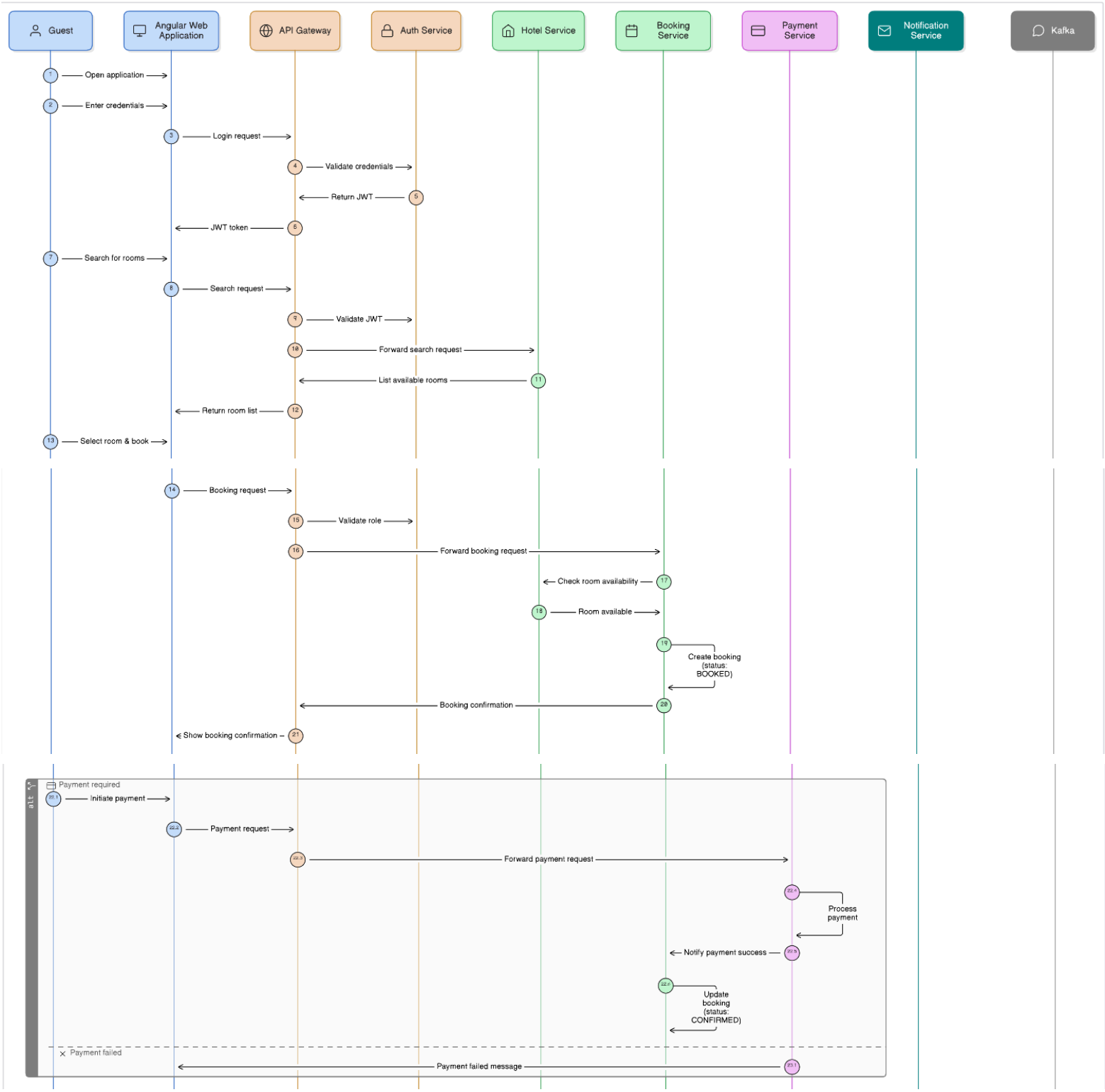
---

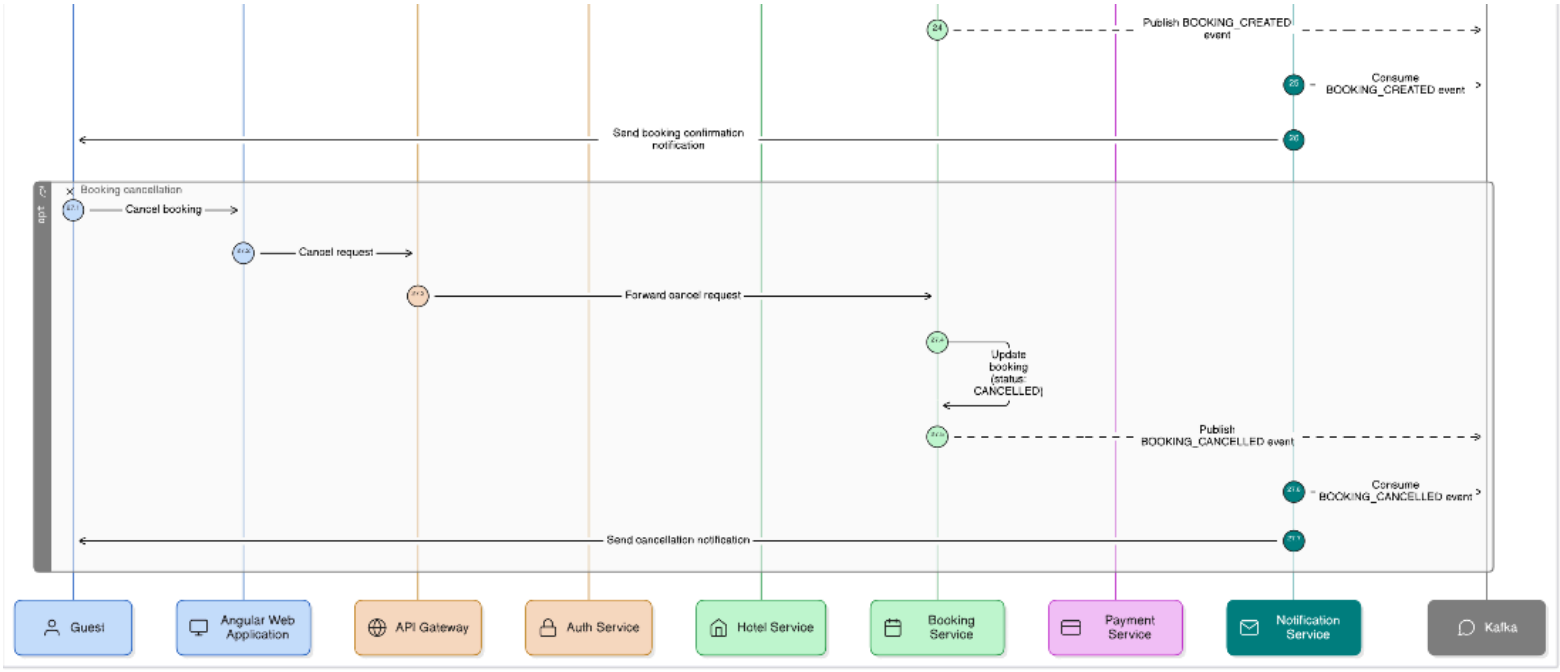
7. Notifications, Reports, and Data Integrity

- Booking events are published asynchronously to trigger notifications.
- Notification failures do not affect booking or payment processes.
- Reports such as occupancy and revenue are accessible only to Admin and Manager roles.
- Each microservice owns its data and communicates through APIs or events only.

SEQUENCE DIAGRAM:

This sequence diagram illustrates the end-to-end flow of a hotel room booking, starting from user authentication and room search, followed by booking creation, payment processing, and asynchronous notification delivery. It shows how the Angular frontend interacts with backend microservices through an API Gateway, while Kafka enables event-driven notifications without affecting the booking flow.




Publish BOOKING_CREATED event
Consume BOOKING_CREATED event
Send booking confirmation notification
alt
Booking cancellation
Cancel booking
Cancel request
Forward cancel request
Update booking (status: CANCELLED)
Publish BOOKING_CANCELLED event
Consume BOOKING_CANCELLED event
Send cancellation notification
Guest
Angular Web Application
API Gateway
Auth Service
Hotel Service
Booking Service
Payment Service
Notification Service
Kafka