

# Hotel Reservation System

A Modern Full-Stack Web Application using Spring Boot & Angular

**Presented by:** Debashrita Mandal

**University:** KIIT University, Bhubaneswar

**University Roll No:** 22052810

**Instructor:** Mr. Parivesh Gupta

**Technology Stack:** Spring Boot, Angular,

# Problem Statement & Objectives

This project replaces manual reservation processes with a centralized system for real-time availability, easy booking, and improved hotel operations.

## Problem Statement

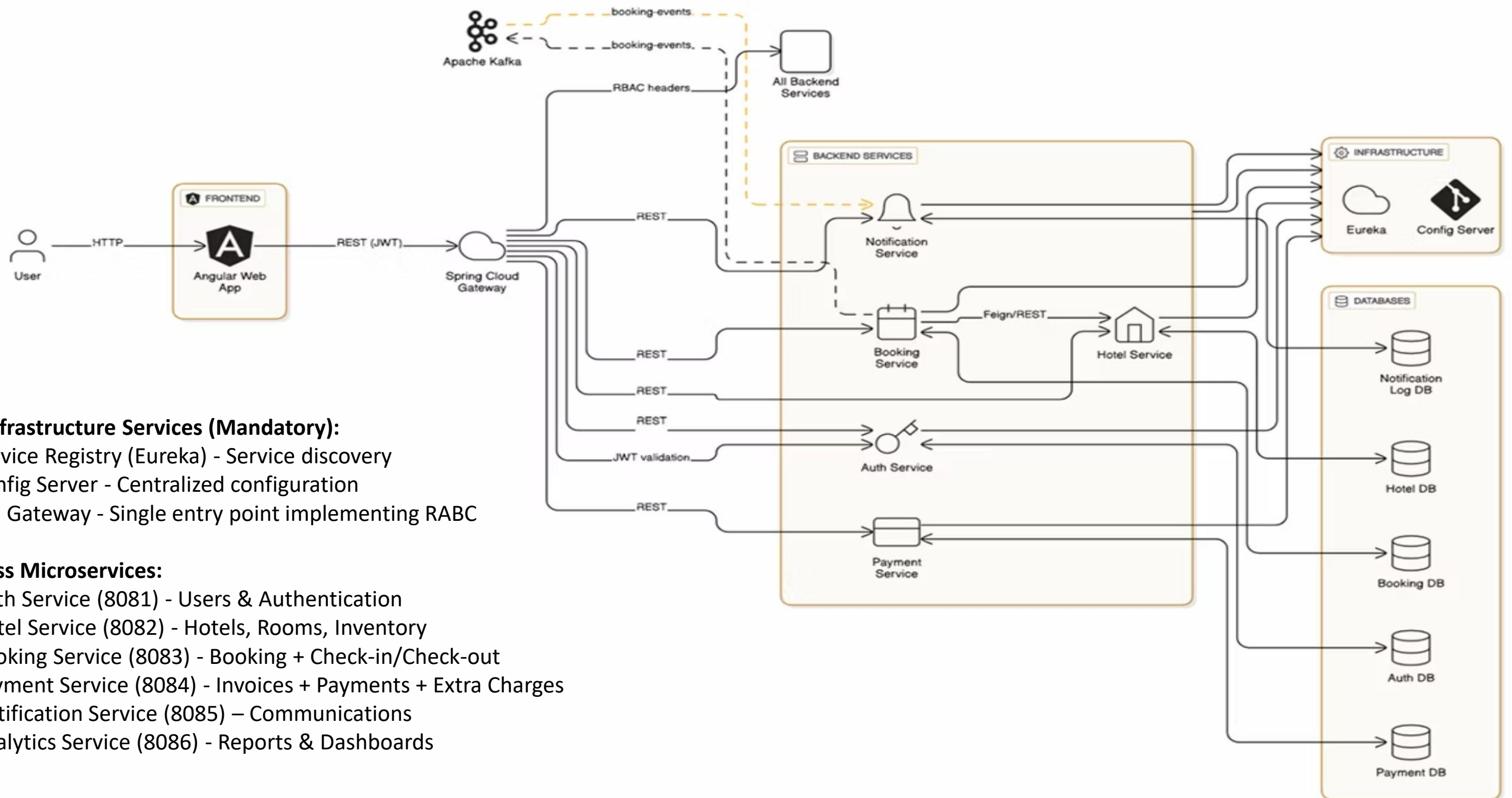
- **Overbooking** or **underutilization** of rooms
- Difficulty tracking **real-time** room availability
- **Inefficient reservation** handling and cancellations
- **Limited visibility** into occupancy rates and revenue
- **Poor customer experience** for booking and management

## Project Objectives

To design and develop a Full-Stack Enterprise Application:

- Uses **Java Spring Boot** as the backend
- Uses **Angular** for frontend development
- Implements **JWT-based authentication and role-based authorization**
- Follows **clean architecture and enterprise best practices**
- Demonstrates real-world **booking, billing, and reporting workflows**

# Microservices Architecture



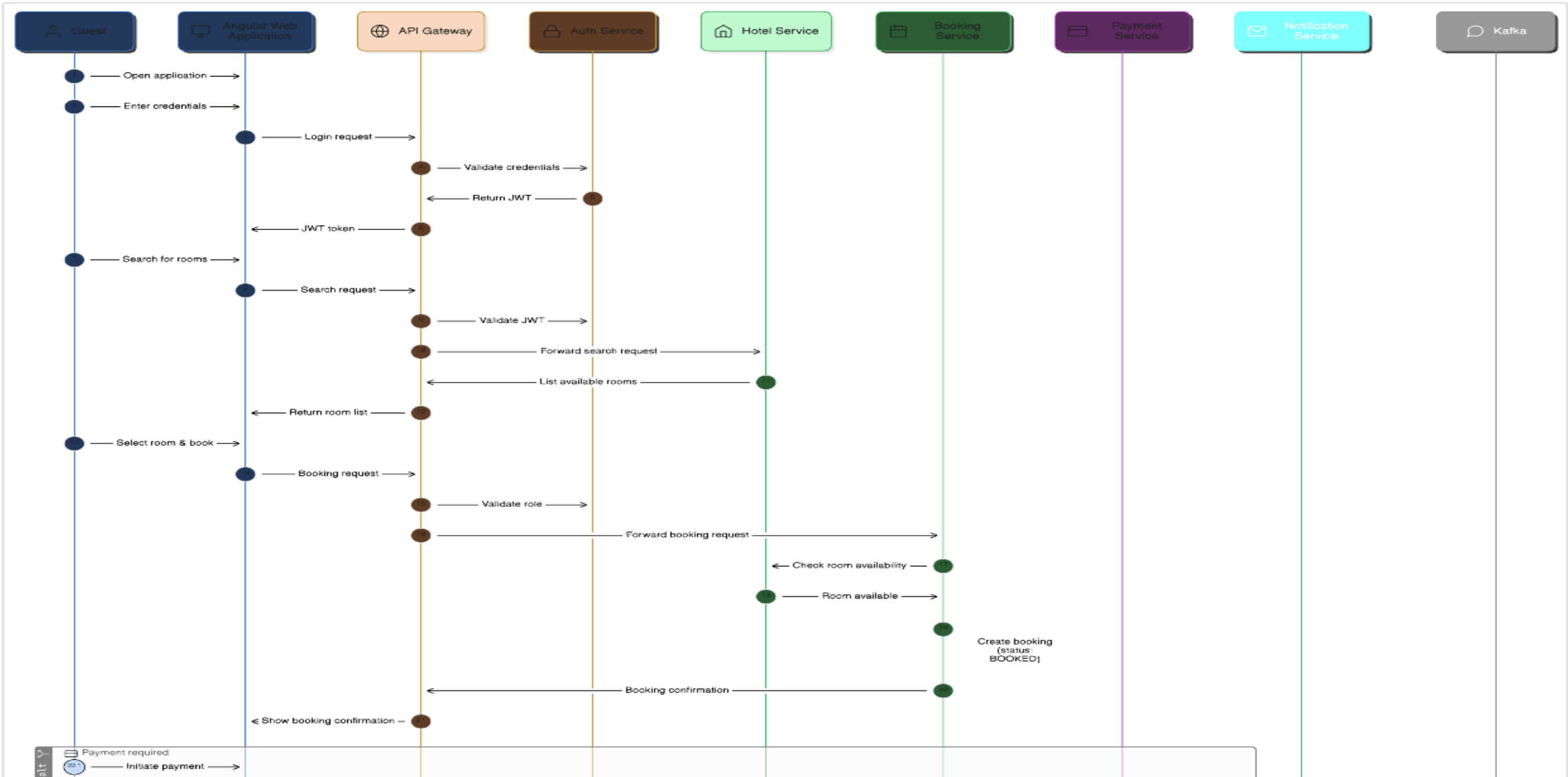
## Core Infrastructure Services (Mandatory):

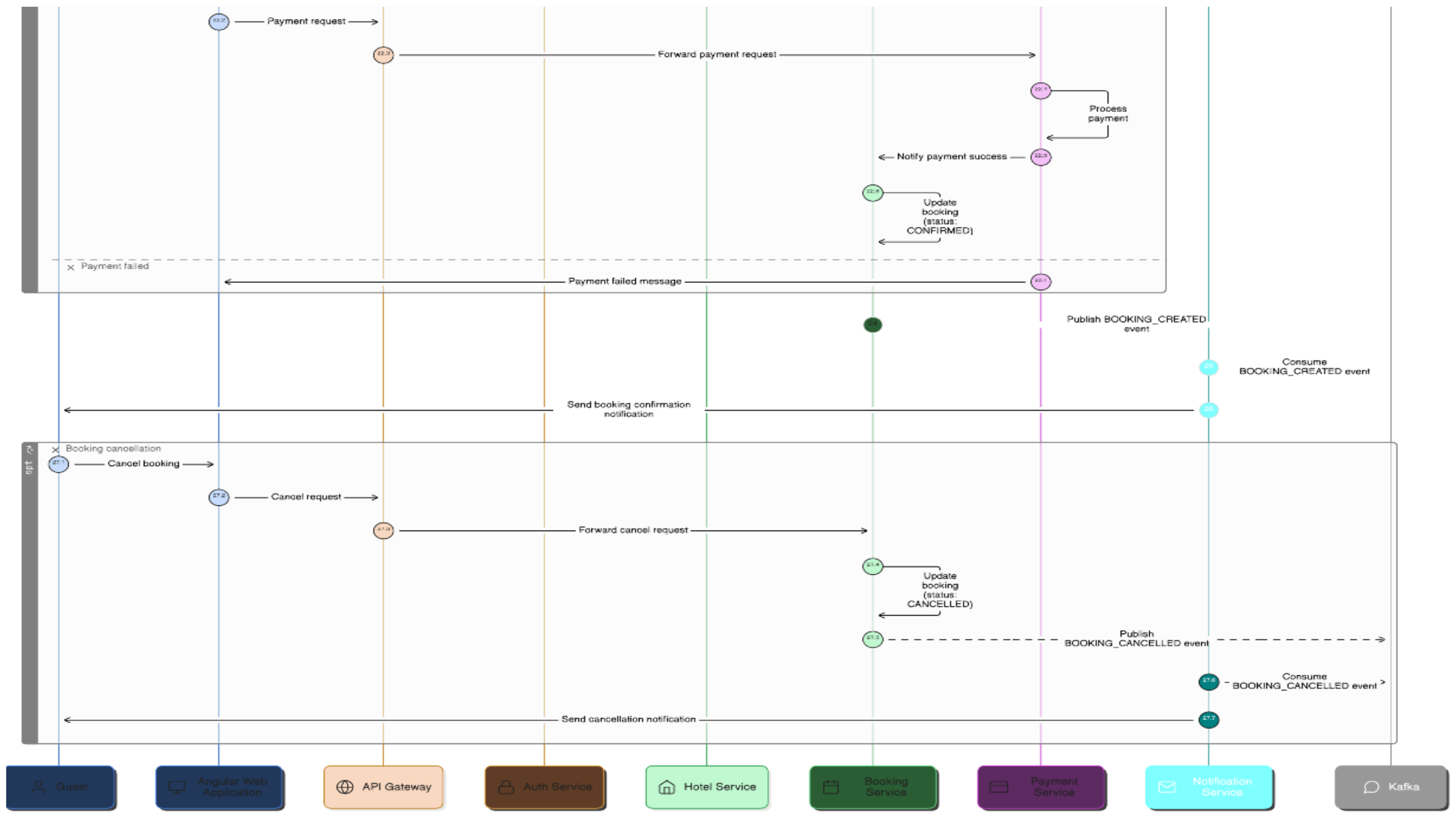
1. Service Registry (Eureka) - Service discovery
2. Config Server - Centralized configuration
3. API Gateway - Single entry point implementing RBAC

## 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

# Sequence Diagram





# Complex Problem 1 — Automated No-Show Detection & Booking State Management

- Implemented a scheduler that runs daily at **12:30 AM**
- Scheduler scans all **confirmed bookings**
- Verifies whether the **check-in date has passed** without check-in
- Automatically marks such bookings as **NO\_SHOW**
- Updates the **booking status** in the system without manual intervention

# Complex Problem 2 — Event-Driven Check-In/ Check-Out Reminder System

- Implemented a scheduler that runs daily at **9:00 AM**
- Identifies **upcoming check-in and check-out schedules**
- Generates reminder notifications for guests
- Publishes reminder messages as **Kafka events**
- Sends emails **asynchronously** by consuming Kafka events

# 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	

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}/seasonal-pricing	Set the seasonal pricing	{ "startDate": "2025-12-20", "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 }

# 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



**THANK YOU!**