**Project Title: SmartTelecom Management System**

**Tech Stack**

* **Backend:** Java 17, Spring Boot
* **Security:** Spring Security with JWT
* **API Gateway:** Spring Cloud Gateway
* **Service Discovery:** Eureka Server
* **Database:** PostgreSQL (for user & plans), MongoDB (for usage logs & analytics)
* **Frontend:** (Optional) Angular/React
* **Build Tool:** Maven
* **Other:** Lombok, ModelMapper, Docker (optional)

**Modules**

**1. API Gateway**

* Routes requests to appropriate microservices
* JWT validation at gateway level

**2. Eureka Server**

* Service discovery for dynamic routing of microservices

**3. Authentication Service**

* User login & registration
* Password encryption using BCryptPasswordEncoder
* JWT token generation
* Role-based access: Admin / Customer

**4. User Service**

* CRUD for customer profiles
* Update mobile plans
* Get plan usage summary

**Uses PostgreSQL**

**5. Plan Management Service**

* Admin adds/modifies/removes mobile plans (prepaid/postpaid/data-only)
* List all plans with filtering/sorting
* Plan recommendation system (simple logic based on usage)

**Uses PostgreSQL**

**6. Usage Tracking Service**

* Tracks call/data usage per customer
* Saves data in MongoDB
* Simulates logs via scheduled multithreaded jobs

**7. Billing Service**

* Generates bills at the end of month
* Retrieves usage logs from MongoDB
* Applies charges based on subscribed plan
* Sends notifications (simulated)

**Security - JWT + Spring Security**

* Secure endpoints using JWT
* Define roles with method-level security using @PreAuthorize

**Key Concepts Integrated**

**SOLID Principles**

* Single Responsibility: Services separated by responsibility
* Open-Closed: Use interfaces for billing logic
* Liskov, Interface Segregation: DTOs for different client roles
* Dependency Inversion: Use interfaces + service injection

**Design Patterns**

* **Singleton:** JWT Token utility
* **Factory Pattern:** PlanFactory for different plan types
* **Observer Pattern:** Notification service (SMS/email observer)
* **Strategy Pattern:** Billing strategies for different plans
* **Template Method:** Monthly billing template

**Collections**

* HashMaps for quick user-plan mapping
* Lists for usage logs
* TreeMap for sorted plan recommendations

**Multithreading**

* Use ExecutorService to simulate parallel user usage logging
* Scheduled job to aggregate logs from MongoDB

**Exception Handling**

* Custom exceptions (e.g., UserNotFoundException, InvalidPlanException)
* Global handler using @ControllerAdvice
* Use ResponseEntity for custom error responses

**Database Design**

**PostgreSQL:**

* users: id, name, email, phone, role, plan\_id
* plans: id, name, type, price, validity, features

**MongoDB:**

* usage\_logs: \_id, user\_id, date, call\_duration, data\_used (MB), sms\_count

Folder Structure:

smart-telecom/

├── gateway-service/

├── eureka-server/

├── auth-service/

├── user-service/

├── plan-service/

├── usage-service/

├── billing-service/

├── common-utils/ (DTOs, Enums, Constants)

└── config-server/ (optional)

**Sample API Endpoints**

**Auth Service**

* POST /auth/register
* POST /auth/login

**User Service**

* GET /users/me
* PUT /users/update
* GET /users/plan-usage

**Plan Service**

* GET /plans
* POST /plans (Admin only)

**Billing Service**

* POST /billing/generate/{userId}
* GET /billing/history