**Notification Service High-Level Design (HLD)**

**1. Overview**

The Notification Service handles all outbound communications in a Task Management System. It listens for task-related events and sends notifications through various channels (Email, SMS, In-App). The service ensures timely delivery and respects user preferences.

**2. Objectives**

* Real-time and scheduled notifications
* Support for Email, SMS, and In-App
* User-specific preferences and delivery options
* Logging and retry mechanisms

**3. Architecture Components**

**3.1 Event Producer**

* Task Management System publishes events (e.g., task assignment)

**3.2 Message Broker**

* Kafka or RabbitMQ queues events

**3.3 Notification Processor**

* Consumes events from the broker
* Coordinates fetching user preferences, templates
* Dispatches messages to appropriate channels

**3.4 Template Engine**

* Loads and personalizes templates for each notification

**3.5 User Preferences Service**

* API to retrieve user settings (e.g., enable SMS, disable email)

**3.6 Channel Senders**

* Email Sender (via SMTP)
* SMS Sender (e.g., Twilio)
* In-App Notifier (e.g., via WebSocket)

**3.7 Notification Database**

* Stores metadata and delivery status

**3.8 Retry Manager**

* Retries failed messages with exponential backoff

**4. Workflow**

1. Task Management System emits an event to the message broker
2. Notification Processor consumes the event
3. User Preferences are checked
4. Template is rendered
5. Message is sent via preferred channel(s)
6. Delivery status is logged
7. Retry Manager handles failures if needed

**5. Plain UML - Component Diagram**

@startuml

package "Notification System" {

[Notification Processor] --> [Template Engine]

[Notification Processor] --> [User Preferences Service]

[Notification Processor] --> [Email Sender]

[Notification Processor] --> [SMS Sender]

[Notification Processor] --> [In-App Notifier]

[Notification Processor] --> [Notification DB]

[Notification Processor] <-- [Message Broker]

[Retry Manager] --> [Notification Processor]

}

package "External Systems" {

[Message Broker] --> [Task Management System]

[Email Sender] --> [SMTP Server]

[SMS Sender] --> [SMS Gateway]

[In-App Notifier] --> [WebSocket Server]

}

@enduml

**6. External Dependencies**

* SMTP Server (for emails)
* SMS Gateway (Twilio, etc.)
* WebSocket infrastructure (for in-app)

**7. Future Considerations**

* Add push notifications (mobile)
* Support for localized templates
* Analytics dashboard for notification stats