# Second Smaller Project: ****Notification Microservice****

### ****Overview****

Build a focused microservice that handles **user notifications**—sending emails, SMS, or push notifications triggered by events from other services.

### Why Notification Microservice?

* Real-world use case: many companies isolate notifications to a dedicated microservice.
* You’ll learn async messaging, event-driven architecture, and integration with third-party APIs.
* Show understanding of service boundaries, inter-service communication, and scalability.

**High-Level Technical Flow & Components**

Other services (e.g., Task API) → Message Broker (RabbitMQ / Kafka) → Notification Microservice → External Providers (SendGrid, Twilio, FCM)

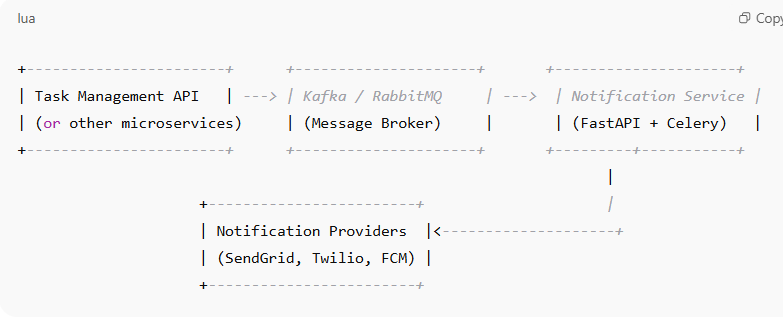
### Key Features & Responsibilities

* **Consume notification events** from a message broker (e.g., Kafka or RabbitMQ).
* **Validate** incoming event payloads with Pydantic models.
* **Support multiple channels**: email (SendGrid), SMS (Twilio), push notifications (Firebase Cloud Messaging).
* **Retry & failure handling**: retry failed sends with exponential backoff.
* **Idempotency**: avoid duplicate notifications for same event.
* **Expose REST API** for managing user notification preferences.
* **Logging & metrics** for observability.
* **Rate limiting & throttling** per user or API key.
* **Deployable as a containerized microservice**, easily scalable.

**Technical Stack**

| **Layer** | **Technology** |
| --- | --- |
| Backend Framework | FastAPI (async) |
| Message Broker | Kafka or RabbitMQ |
| DB for Preferences | PostgreSQL (async) |
| ORM | SQLAlchemy (async) |
| Background Task Processing | Celery or RQ with Redis broker |
| Notification Providers | SendGrid, Twilio, Firebase Cloud Messaging (FCM) |
| Caching / Rate Limit | Redis |
| Observability | Prometheus + Grafana, JSON logging |
| Deployment | Docker + Kubernetes or ECS Fargate |

**Architecture Diagram**



**Quick flow:**

1. **Task API** publishes a "task\_completed" event with user info to Kafka.
2. **Notification Service** consumes event asynchronously.
3. Validates event data via Pydantic.
4. Checks user preferences from PostgreSQL.
5. Sends email/SMS/push notification using external APIs.
6. Logs success/failure and retries if needed.
7. Exposes an API for users/admins to update notification preferences.
8. Handles rate limiting and throttling via Redis.

**What you'll learn:**

* Event-driven microservice design
* Message queue integration (Kafka/RabbitMQ)
* Async task processing with Celery/RQ
* Working with multiple external APIs reliably
* Microservice-specific concerns: idempotency, retries, rate-limits
* Service observability & metrics
* Containerization + deployment (Kubernetes or AWS ECS)

After completing the **main Task Management API** (7-day deep dive covering everything from async DB, auth, caching, rate-limiting, background tasks, deployment, etc.) **plus** this **Notification Microservice** (microservices, event-driven architecture, message brokers, retries, external APIs, scaling), your FastAPI expertise will be **truly production-grade** and **interview-ready** for roles expecting 3–5 years of experience.

You will have hands-on mastery of:

* Core FastAPI development with async SQLAlchemy, Alembic migrations, Pydantic validation
* Authentication, role-based access control, API key validation
* Advanced API design: pagination, caching, rate limiting, dependency injection
* Background tasks, worker queues (Celery/RQ) and external API integrations
* Resilience patterns: retries, idempotency, circuit breakers
* Observability: logging, metrics, tracing
* Deployment: containerization, AWS ECS/Fargate or Lambda
* Microservices concepts: event-driven architecture, message queues, inter-service communication
* Scalability, concurrency handling, failover and recovery

This combined learning path **fully covers** what interviewers typically expect in senior backend FastAPI roles, including production-grade system design and operational best practices.

If you want, I can also help you prepare a compact interview Q&A bank covering all these topics — so you can answer questions confidently in interviews.