Challenge: Fullstack Staff Engineer

Contamos con un ecosistema de microservicios que reciben y emiten eventos en tiempo real. Necesitamos un servicio que procese mensajes entrantes, los cuente por intervalos de tiempo y exponga la información a otros sistemas internos y externos.

El objetivo del reto es evaluar cómo estructuras tu código, cómo piensas la solución a nivel de algoritmia y cómo manejas la conexión con fuentes externas (bases de datos, APIs).

Caso de uso

Existe un servicio que emite **mensajes** a través de un **webhook**. Queremos construir un nuevo servicio que:

1. **Reciba el webhook del mensaje**
    - El mensaje contendrá al menos: `message_id`, `account_id`, `created_at`.
    - Cada `message_id` es único y no debe contarse dos veces en caso de que por algún motivo llegue duplicado

Estructura del mensaje de webhook:

```
{
  "message_id": "msg_01HX9FZ2E0KJ8C3Q9XP",
  "account_id": "acc_12345",
  "created_at": "2025-09-18T10:42:17Z",
  "metadata": {
    "channel": "whatsapp",
    "source": "inbound",
    "tags": ["camp:septiembre", "mx"]
  },
}
```

2. **Cuente mensajes por hora en una base de datos**
    - Se deben agrupar por `account_id` y por hora (ej. `2025-07-18T15:00:00Z`).
    - Cada hora debe tener un contador acumulado.
3. **Emita un total diario a un servicio externo**
    - Cada vez que llega un mensaje, después de actualizar el conteo por hora, tu servicio debe calcular el **total de mensajes del día** para esa cuenta y enviarlo vía **POST** a un servicio externo (puedes simularlo con un endpoint local).
    - Diseñar una estructura de json (contrato) que cumpla con los requerimientos para emitir el evento con la información necesaria

4. **Exponga un endpoint de consulta**
    - Endpoint: `GET /counts`
    - Parámetros: `account_id`, `from`, `to` (ISO8601).

Debe devolver una lista de conteos por hora con el formato:

```
 [
  { "account_id": "acc_123", "datetime": "2025-09-18T10:00:00Z",
"count_messages": 12 },
  { "account_id": "acc_123", "datetime": "2025-09-18T11:00:00Z",
"count_messages": 34 }
]
```

Requisitos técnicos

- Puedes implementar el servicio en el lenguaje de programación que mejor te acomodes
- Base de datos: puedes usar cualquier base de datos relacional o no relacional
- El endpoint externo al que envías el total diario puede ser un mock simple

---

Entregables

1. **Código fuente** en un repositorio público
2. **Instrucciones de ejecución** (`README.md`) indicando cómo levantar el servicio y todo lo necesario para probar
3. **Explicación breve** (`Decisions.md`) donde describas:
    - Cómo diseñaste la solución (estructura, modelo de datos, endpoints).
    - Cómo aseguraste idempotencia y consistencia.
    - Supuestos que tomaste y qué mejorarías con más tiempo.
    - Si usaste IA, cómo fue que la usaste.
4. (Opcional) **Unit test**