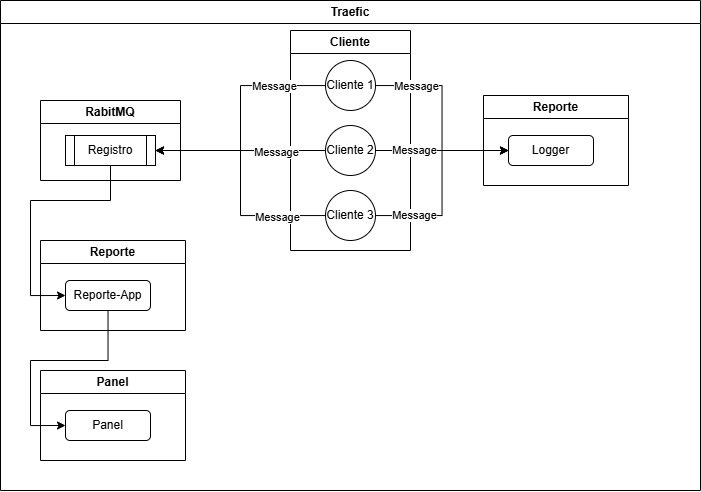
**Taller RabitMQ**

**Nombres:** Alex Duvan Hernandez Buitrago

Yesid Alejandro Martinez Guerrero

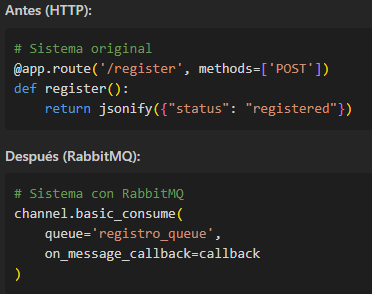
**1. Arquitectura con RabitMQ**



**2. Cambios respecto a la versión original**

Se realizaron principalmente los siguientes cambios con respecto al proyecto original:

1. Se dejo de lado el uso de un HTTP sincrónico para la comunicación entre los servicios y se reemplazó por un servicio asincrónico (RabitMQ) de mensajería
2. Se creo un servicio dedicado para logging
3. La implementación de RabitMQ mejoro la tolerancia a errores por su naturaleza de guardar los mensajes, pero adicional se implementaron manejo de errores como reintentos automáticos.
4. Se implemento el monitoreo del servicio RabitMQ lo que mejora su seguimiento



**3. Justificación del diseño propuesto.**

El diseño actual realizado implementado RabitMQ se justifica debido a las ventajas que obtuvo el uso de esta tecnología:

1. Al no depender los servicios directamente entre si facilita su integración y la implementación de manejo de errores independientes
2. Como mencionado previamente el monitoreo del servicio RabitMQ lo que mejora su seguimiento lo que facilita la verificación de funcionalidad de cada cliente y el logger y en general los servicios del sistema
3. A diferencia del uso de un HTTP sincrónico impide la perdida de mensajes lo que da mas confiabilidad al sistema

**4. Lecciones aprendidas sobre RabbitMQ.**

1. RabitMQ es un servicio que facilita la comunicación entre servicios dentro de un mismo sistema

2. Facilita la distribución de las funcionalidades en un programa lo que mejora su rendimiento

3. Permite una integración mas limpia y de fácil aplicación.

4. Su naturaleza asincrónica mejora el rendimiento y el manejo de errores dentro de las comunicaciones de los servicios.

5. Servicios de esta índole (no solo RabitMQ) tienen una amplia aplicación en distintos programas y se puede decir que es fundamental su implementación para el manejo de aplicaciones con múltiples peticiones