RabbitMQ e Kafka

- Comparação, Conceitos e Aplicações

Objetivo

- - Entender mensageria e streaming de eventos
- - Conhecer RabbitMQ e Kafka
- - Comparar funcionalidades, semelhanças e diferenças
- - Exemplos de uso em microserviços

Conceitos Básicos

- - Mensageria: comunicação assíncrona entre serviços
- - Broker: componente que recebe, armazena e distribui mensagens
- - Produtor: envia mensagens
- - Consumidor: recebe mensagens

RabbitMQ

- - Tipo: Message Broker
- - Baseado em Filas (Queues) e Exchanges
- - Entrega de mensagens confiável
- - Persistência opcional
- - Uso típico: tarefas assíncronas, notificações, pedidos

Kafka

- - Tipo: Event Streaming Platform
- - Baseado em Tópicos (Topics) e Partições (Partitions)
- - Mensagens persistentes (log imutável)
- - Escalabilidade horizontal elevada
- - Uso típico: logs, métricas, analytics, pipelines de eventos

Modelo de Mensageria

- RabbitMQ: mensagem consumida → removida
- Kafka: mensagem permanece (log)
- Consumidores: RabbitMQ → 1 mensagem por consumidor, Kafka → múltiplos consumidores
- Ordem: RabbitMQ → por fila, Kafka → por partição
- Reprocessamento: RabbitMQ difícil, Kafka fácil

Persistência e Retenção

- RabbitMQ: Persistência opcional, retenção até consumir, volume moderado
- Kafka: Persistência sempre, retenção configurável, alto volume de mensagens

Escalabilidade e Performance

- RabbitMQ: baixa latência, escalabilidade limitada
- Kafka: alto throughput, escalabilidade horizontal, ideal para streaming de eventos

Casos de Uso

- RabbitMQ: Processamento de tarefas, Fila de pedidos, Notificações e e-mails
- Kafka: Streaming de eventos, Logs e métricas em tempo real, Pipelines de dados e integração de sistemas

Semelhanças

- - Assíncronos
- - Baseados em mensagens/eventos
- - Escaláveis e distribuídos
- - Suportam confirmação de entrega
- - Open Source

Quando usar cada um

- RabbitMQ: sistema simples, jobs assíncronos
- Kafka: armazenar/reprocessar eventos, alto volume, streaming e integração pipelines

Stream Queue (RabbitMQ 3.9+)

- - Combina modelo tradicional de fila com streaming
- - Mantém histórico completo de eventos
- - Consumidores podem ler desde o início ou só novas mensagens
- - Semelhante ao modelo de tópicos e partições do Kafka

Conclusão

- - RabbitMQ ⟶ tarefas e filas confiáveis
- - Kafka ⟶ streaming, logs e alto volume
- - Ambos são valiosos para microserviços; escolha depende do caso de uso