# ADR-002: Uso de Message Broker para Sistema de Notificações por E-mail

## Status

Aceito

## Contexto

Nossa plataforma de e-commerce precisa enviar diferentes tipos de notificações por e-mail para os clientes (confirmação de pedido, atualização de status, recuperação de senha, etc). Atualmente, o envio é feito de forma síncrona dentro do fluxo principal de cada operação, causando os seguintes problemas:

* Atrasos na resposta ao usuário devido ao tempo de processamento do envio de e-mail
* Falhas em operações importantes quando o serviço de e-mail está indisponível
* Dificuldade em escalar o sistema durante picos de demanda
* Alto acoplamento entre os serviços e a lógica de envio de e-mails

### Métricas relevantes:

* Volume médio: 50.000 e-mails/dia
* Picos de até 200.000 e-mails em datas especiais
* Tempo médio atual de envio: 2-3 segundos por e-mail
* Taxa de falha atual: 5% devido a timeouts

## Decisão

Implementar um sistema de message broker (Apache Kafka) para gerenciar o envio assíncrono de notificações por e-mail.

Arquitetura proposta:

1. Serviços produtores publicam mensagens em tópicos específicos do Kafka

2. Um serviço consumidor dedicado processa as mensagens e realiza o envio dos e-mails

3. Implementação de dead-letter queue para tratamento de falhas

4. Uso de múltiplas partições para permitir processamento paralelo

## Consequências

### Benefícios

- Desacoplamento entre serviços e lógica de envio de e-mails

- Melhor experiência do usuário com respostas mais rápidas

- Capacidade de escalar horizontalmente o processamento de e-mails

- Resiliência a falhas temporárias do serviço de e-mail

- Sistema de retry automático para mensagens com falha

- Melhor observabilidade do processo de envio

### Desafios

- Necessidade de manter infraestrutura adicional (Kafka)

- Complexidade adicional no monitoramento

- Possibilidade de duplicação de mensagens em cenários de retry

- Necessidade de treinamento da equipe em message brokers

## Alternativas Consideradas

### RabbitMQ

- Prós:

- Mais simples de configurar e manter

- Menor consumo de recursos

- Contras:

- Menor capacidade de escala

- Menos recursos para reprocessamento de mensagens

- Motivo da não escolha: Necessitamos da capacidade de replay de mensagens e maior escala que o Kafka oferece

### Redis Pub/Sub

- Prós:

- Já utilizamos Redis na infraestrutura

- Baixa latência

- Contras:

- Sem persistência de mensagens

- Limitações no padrão de consumo

- Motivo da não escolha: Falta de persistência e garantias de entrega

### Sistema de Filas próprio com banco de dados

- Prós:

- Controle total sobre a implementação

- Sem necessidade de nova infraestrutura

- Contras:

- Alto esforço de desenvolvimento e manutenção

- Risco de problemas de escalabilidade

- Motivo da não escolha: Custo de desenvolvimento e manutenção muito alto

## Referências

- Documentação do Apache Kafka: https://kafka.apache.org/documentation/

- Patterns of Enterprise Application Architecture - Martin Fowler

- RFC 2821 - Simple Mail Transfer Protocol

- ADR-001: Escolha do Provedor de Serviço de E-mail

## Notas

- A implementação será feita de forma gradual, começando com e-mails de confirmação de pedido

- Necessário alinhar com a equipe de infraestrutura sobre o sizing dos clusters Kafka

- Definir estratégia de monitoramento e alertas

- Criar documentação sobre padrões de mensagens e tratamento de erros

* Autor original: Renan Torres
* Data de aprovação: 10/02/2025
* Aprovado por: Ciro
* Data de substituição: --
* Data da última modificação: 12/02/2025
* Modificado por: Renan Torres
* Última modificação: Incrementos de metadata