Documento de Arquitetura de Software (DAS)

**Sistema de Controle e Consolidação de Fluxo de Caixa**

Autor: Fernando Pinheiro Gomes

Data: 31/08/2025

Versão: 1.0

# 1. Introdução

Este documento apresenta a arquitetura de software proposta para o controle e consolidação do fluxo de caixa de um comerciante. O sistema permitirá o registro de lançamentos financeiros diários (débitos e créditos) e a geração de relatórios consolidados com o saldo diário. A solução deve ser precisa, escalável, resiliente e segura, atendendo aos requisitos funcionais e não funcionais definidos.

# 2. Objetivos e Escopo

O objetivo principal é desenvolver um sistema que:

* Permita o controle de lançamentos financeiros (débito e crédito).
* Disponibilize relatórios de saldo consolidado diário.
* Garanta resiliência entre os serviços de lançamento e consolidação.
* Seja escalável para suportar picos de até 50 requisições/segundo no serviço de consolidação.
* Implemente práticas de segurança, monitoramento e auditoria.

# 2.1 Microserviços propostos

O escopo contempla a implementação de um conjunto de microserviços desacoplados que se comunicam de maneira assíncrona por meio de um barramento de mensageria, garantindo que o serviço de lançamentos não dependa da disponibilidade imediata do serviço de consolidação.

A arquitetura proposta é baseada em microsserviços orientados a eventos. O **Serviço de Lançamentos** será responsável por registrar os débitos e créditos informados pelos usuários, persistindo-os em um banco de dados transacional e publicando eventos em um barramento de mensageria. Esses eventos serão consumidos pelo **Serviço de Consolidação Diária**, que processará as informações recebidas e calculará o saldo consolidado do dia, armazenando o resultado em uma base analítica destinada a relatórios e consultas posteriores. Para disponibilizar os resultados ao comerciante exporá uma API de consulta e também permitirá a exportação de informações em formatos como PDF e CSV, possibilitando a integração com dashboards ou sistemas de terceiros. Complementarmente, o **Serviço de Notificações** será responsável por informar o comerciante sobre o fechamento do saldo diário por meio de e-mail, webhooks ou mensageria, enquanto o **Serviço de Autenticação e Autorização** será simulado para assegurar o acesso seguro ao sistema, definindo papéis específicos como administrador, operador financeiro e auditor. Por fim, o Serviço de Monitoramento e Observabilidade garantirá a coleta de métricas, logs e rastreamento distribuído, viabilizando a gestão proativa do ambiente com ferramentas especializadas.

Os principais microserviços que compõem a solução são:

* **Serviço de Lançamentos**: responsável pelo registro de débitos e créditos e publicação de eventos.
* **Serviço de Consolidação Diária**: realiza o processamento dos lançamentos e gera o saldo consolidado.
* **Serviço de Notificações**: comunica o fechamento do saldo diário por e-mail, webhook ou mensageria.
* **Serviço de Autenticação e Autorização**: garante acesso seguro e controle de permissões.
* **Serviço de Monitoramento e Observabilidade**: coleta métricas, logs e rastreamento distribuído. Serão utilizadas as ferramentas disponibilizadas pelo próprio framework para coleta das métricas

# 3. Arquitetura Proposta (Visão Geral)

A arquitetura proposta segue o padrão de microserviços orientados a eventos, garantindo desacoplamento entre o serviço de controle de lançamentos e o serviço de consolidação diária. A comunicação assíncrona via mensageria (ex: Apache Kafka) assegura que os lançamentos continuem funcionando mesmo se o serviço de consolidação estiver temporariamente indisponível. Para os efeitos deste desafio, imagina-se que será disponibilizada uma nuvem pública para implementação dos serviços, garantindo escalabilidade, resiliência e facilidade de integração entre os componentes da arquitetura. Como exemplo de implementação, serão utilizados serviços da **Amazon Web Services** (AWS), tais como **Amazon EKS** para orquestração de containers, **Amazon RDS (PostgreSQL)** para persistência transacional, **Amazon MSK (Kafka)** para mensageria, **Amazon S3** para armazenamento analítico e **Amazon CloudWatch** para monitoramento e observabilidade.

**Componentes principais:**

1. API Gateway: entrada segura para clientes.

2. Serviço de Lançamentos: responsável por registrar débitos e créditos.

3. Serviço de Consolidação: processa os lançamentos do dia e gera saldo consolidado.

4. Banco de Dados Transacional: armazena lançamentos (ex: PostgreSQL). Poderá ser utilizada esta base de dados por conta da sua eficiência e baixo custo.

6. Serviço de Notificação: responsável por enviar relatórios ou disponibilizar APIs para consulta.

7. Monitoramento e Observabilidade: coleta métricas e logs (Prometheus, Grafana, ELK ou CloudWatch).

Diagrama

O conteúdo gerado por IA pode estar incorreto.

# 4. Arquitetura de Backend

Os microserviços serão desenvolvidos em Java 21 com Spring Boot, garantindo robustez e produtividade. O **Serviço de Lançamentos** processa requisições síncronas para registrar dados em banco relacional, enquanto o **Serviço de Consolidação** consome eventos da fila de mensageria e gera o saldo diário.

# 5. Arquitetura de Frontend

Será desenvolvido um frontend web responsivo utilizando Angular. Este frontend permitirá: a) registrar lançamentos, b) visualizar relatórios diários consolidados, c) acionar reprocessamentos. A comunicação com o backend será via APIs REST seguras.

# 6. Tecnologias e Frameworks

* Frontend: Angular + TypeScript
* Backend: Java 21 + Spring Boot 3.x
* Banco de Dados: PostgreSQL para lançamentos
* Mensageria: Apache Kafka (Amazon MSK)
* Cache: Redis para dados de alta consulta
* Infraestrutura: Kubernetes (EKS) + Docker

# 7. Infraestrutura

Os serviços serão containerizados com Docker e orquestrados via Kubernetes (EKS). Será adotada uma arquitetura multi-AZ para garantir resiliência e alta disponibilidade. O API Gateway, Kafka e bancos de dados estarão distribuídos em múltiplas zonas de disponibilidade.

# 8. Escalabilidade, Alta Disponibilidade e Resiliência

A arquitetura permitirá escalabilidade horizontal dos microserviços com autoscaling em Kubernetes. O serviço de lançamentos funcionará mesmo se o serviço de consolidação estiver indisponível. O banco de dados terá replicação e failover automático, e o Kafka assegura retenção de eventos para reprocessamento. Para rodar localmente a solução, será disponibilizadado um docker compose, que por sis ó não garante alta resiliência e disponibilidade. Para ambientes de testes e desenvolvimento, isto não será necessário.

# 9. Segurança

Medidas de segurança incluem:

- Autenticação e autorização via OAuth2/OpenID Connect (Keycloak ou Cognito).  
- Criptografia em trânsito (TLS 1.2+) e em repouso (KMS).  
- Proteção contra ataques comuns com WAF e validação de payloads.  
- Segregação de redes (VPC, sub-redes públicas e privadas).

# 10. DevSecOps

Serão implementadas pipelines CI/CD (GitHub Actions ou GitLab CI) para build, testes automatizados, análise de qualidade (SonarQube), segurança de containers (Trivy), deploy automatizado em Kubernetes, e observabilidade com Prometheus/Grafana.

# 11. Evoluções Futuras

Possíveis evoluções: integração com sistemas de contabilidade, exportação de relatórios fiscais, uso de machine learning para previsões de fluxo de caixa, e suporte a multi-lojas.