Documentação Técnica - Integração com HubSpot

1. Introdução

Este documento descreve o desenvolvimento de uma API REST que integra com a API do HubSpot, implementando autenticação via OAuth 2.0 (authorization code flow) e a criação de contatos. A solução foi implementada utilizando Java e Spring Boot, com foco em segurança, escalabilidade e boas práticas de código.

O desafio foi realizado em um prazo apertado devido ao contexto de final de sprint no Itaú, o que impactou diretamente na elaboração e refinamento da solução. No entanto, as decisões tomadas consideram os requisitos técnicos e boas práticas de desenvolvimento.

---

2. Tecnologias Utilizadas

2.1 Java 17

A escolha do Java 17 foi baseada na sua versão LTS (Long-Term Support), o que garante estabilidade e segurança por um período prolongado, além de melhorias de desempenho e suporte a novas funcionalidades. O Java é a linguagem de desenvolvimento mais apropriada para o desafio, considerando sua robustez e a facilidade de integração com o Spring Boot.

2.2 Spring Boot

O Spring Boot foi a escolha para o framework devido à sua flexibilidade, simplicidade e alto grau de adoção no mercado. Ele permite a criação de APIs RESTful de forma rápida e eficiente, com forte suporte para integração com sistemas externos, como o HubSpot. O Spring Boot também simplifica o gerenciamento de dependências e configurações.

2.3 OAuth 2.0

Para garantir a segurança na troca de dados entre a aplicação e o HubSpot, optou-se pelo uso do OAuth 2.0 com o fluxo de autorização "authorization code flow". Isso permite uma autenticação segura sem a necessidade de armazenar as credenciais do usuário diretamente na aplicação.

2.4 WebClient (Spring WebFlux)

A escolha do WebClient, em vez do tradicional RestTemplate, foi feita devido à sua capacidade de realizar chamadas HTTP assíncronas e não bloqueantes. Isso é essencial para garantir a escalabilidade da aplicação, especialmente em sistemas com grande volume de requisições. A utilização do Spring WebFlux com WebClient melhora o desempenho e reduz o uso de recursos.

2.5 H2 Database

Para a fase de desenvolvimento, utilizamos o H2 Database, pois ele é um banco de dados em memória que facilita o processo de testes e desenvolvimento, permitindo uma configuração simples e rápida. Contudo, para a produção, o

banco de dados será substituído por um banco mais robusto, como o PostgreSQL, que oferece mais funcionalidades, como suporte a transações complexas e maior escalabilidade.

2.6 Docker e Kubernetes

Docker foi utilizado para criar containers que garantem a portabilidade e consistência da aplicação em diferentes ambientes. Já o Kubernetes foi escolhido para orquestrar a aplicação, garantindo alta disponibilidade e escalabilidade. O uso desses recursos permite que a aplicação seja facilmente gerenciada e dimensionada.

2.7 Kafka

A arquitetura orientada a eventos foi implementada com o Kafka, um sistema de mensageria altamente escalável. O Kafka foi escolhido para garantir uma comunicação assíncrona eficiente entre os serviços, especialmente para o recebimento de webhooks do HubSpot.

---

3. Decisões de Arquitetura e Design

3.1 Arquitetura Hexagonal

A arquitetura hexagonal (ou Ports and Adapters) foi adotada para garantir uma separação clara entre o núcleo da aplicação e suas dependências externas. Isso facilita a manutenção e extensibilidade do sistema, permitindo a troca de tecnologias externas (como o banco de dados ou o serviço HubSpot) sem impactar diretamente o core da aplicação.

3.2 Event-Driven Architecture

A integração com o HubSpot utiliza um modelo orientado a eventos, no qual a criação de contatos gera eventos que são consumidos por outros sistemas. Essa abordagem permite uma comunicação desacoplada, onde os componentes do sistema podem atuar de maneira independente, o que melhora a escalabilidade e a resiliência.

3.3 SOLID e Clean Code

As boas práticas de Clean Code e os princípios SOLID foram seguidos durante o desenvolvimento da API. A ênfase foi colocada na criação de um código limpo e legível, com responsabilidades bem definidas. Além disso, a aplicação de SOLID garante que o código seja fácil de manter e extensível para futuras melhorias.

3.4 Webhooks e Kafka

O uso de webhooks do HubSpot, junto com o Kafka, permite que eventos de criação de contatos sejam consumidos de maneira assíncrona e eficiente. Isso reduz a carga de trabalho do sistema principal e permite a integração em tempo real com o HubSpot.

4. Requisitos de Segurança

A segurança foi uma prioridade durante o desenvolvimento da API, especialmente considerando a troca de tokens de acesso com o HubSpot. As práticas de segurança adotadas incluem:

- **Armazenamento seguro de tokens**: Os tokens de acesso e refresh são armazenados de forma segura, seguindo as melhores práticas de criptografia e segurança.
- **Validação de entradas**: Todos os dados de entrada são validados para evitar injeções de SQL ou outros tipos de ataques.
- **Uso de HTTPS**: A comunicação entre a aplicação e o HubSpot é realizada sobre HTTPS para garantir a confidencialidade e integridade dos dados.

---

5. Melhorias Futuras

5.1 Renovação Automática de Tokens

Uma das melhorias futuras será a implementação de um processo automatizado para renovar os tokens de acesso, garantindo que a aplicação não precise realizar o login manualmente após cada expiração do token.

5.2 Suporte a Outros Endpoints do HubSpot

Atualmente, a API suporta apenas a criação de contatos. No futuro, será interessante adicionar suporte para outras operações da API do HubSpot, como a atualização e exclusão de contatos, além da criação de negócios ou empresas.

5.3 Banco de Dados

Para produção, o H2 Database será substituído por um banco de dados mais robusto, como o PostgreSQL, que oferece melhor suporte a transações e maior escalabilidade.

5.4 Observabilidade

A implementação de monitoramento e logging mais detalhado será essencial para garantir que a aplicação possa ser diagnosticada e otimizada em tempo real. Ferramentas como OpenTelemetry e Prometheus podem ser integradas para esse fim.

5.5 Testes

O projeto precisa de uma cobertura de testes, incluindo testes unitários e de integração, além de validações de performance. Testes de carga e estresse também serão importantes para garantir que a aplicação consiga lidar com altos volumes de requisições, especialmente ao consumir webhooks em tempo real.

---

6. Conclusão

Este projeto foi desenvolvido para integrar com a API do HubSpot, implementando um fluxo de autenticação OAuth 2.0 e a criação de contatos através de uma API REST. O foco foi em criar uma solução segura, escalável e de fácil manutenção, utilizando as melhores práticas de desenvolvimento e arquitetura.

As decisões técnicas foram tomadas levando em consideração a complexidade do desafio, a necessidade de entrega rápida e o cenário de produção. Com a implementação de melhorias futuras, como renovação de tokens e suporte a mais funcionalidades do HubSpot, a solução pode ser expandida e otimizada conforme as necessidades do negócio.

T