Documentação Técnica - Integração com HubSpot

1. Introdução

Este documento tem como objetivo descrever tecnicamente a implementação de uma API REST desenvolvida com o framework Spring Boot para realizar a integração com os serviços da plataforma HubSpot. O sistema permite autenticação via OAuth 2.0, criação e listagem de contatos e consumo de eventos via Webhooks.

O desenvolvimento desta API foi realizado como parte de um desafio técnico com o intuito de avaliar a capacidade de integração com APIs externas, estruturação de código limpo e uso de boas práticas na construção de serviços web modernos.

2. Arquitetura e Tecnologias

A aplicação foi construída utilizando os seguintes recursos tecnológicos:

- **Java 21**: versão moderna da linguagem com melhorias de performance e sintaxe.
- **Spring Boot 3.4.4**: plataforma principal para construção do serviço, fornecendo robustez e agilidade.
- **Spring Security OAuth2 Client**: responsável por gerenciar o fluxo de autenticação com a HubSpot.
- **Lombok**: utilizado para reduzir a verbosidade do código com geração automática de getters, setters e construtores.
- **Jackson**: biblioteca utilizada para conversão entre JSON e objetos Java.
- **SpringDoc OpenAPI (Swagger UI)**: ferramenta de documentação interativa da API acessível via `/docs`.
- **Ngrok**: ferramenta para expor a aplicação local para a internet, necessária para testes com Webhooks.

3. Fluxo de Autenticação

O fluxo de autenticação foi implementado utilizando o protocolo OAuth 2.0. A aplicação inicia o processo redirecionando o usuário para a tela de login da HubSpot através do endpoint `/oauth/authorize`. Após o consentimento, a HubSpot redireciona para o endpoint `/oauth/callback` com um código de autorização.

Esse código é então trocado por um `access_token` e um `refresh_token` através de uma requisição à API da HubSpot. O `access_token` obtido é armazenado temporariamente em memória (simulando uma estrutura simples de armazenamento).

4. Manipulação de Contatos

A API possui endpoints para criação e listagem de contatos utilizando os serviços da HubSpot. O endpoint `POST /contacts` permite a criação de novos contatos informando nome, sobrenome e e-mail. O endpoint `GET /contacts` retorna uma lista de contatos previamente cadastrados.

Todos os dados trafegam em formato JSON e são devidamente convertidos para objetos Java por meio do `ObjectMapper`. DTOs específicos foram criados para representar tanto as requisições quanto as respostas, promovendo clareza e manutenção facilitada.

5. Webhooks

O endpoint `POST /webhook` é utilizado para receber notificações automáticas da HubSpot, especificamente para o evento `contact.creation`. Essa funcionalidade permite que a aplicação reaja a novos cadastros realizados diretamente no CRM.

O conteúdo do webhook é recebido como uma lista de objetos JSON contendo informações como ID do evento, tipo de evento, e o identificador do contato criado. Essa estrutura é parseada e registrada para fins de acompanhamento e testes.

6. Documentação da API

Foi integrada à aplicação a biblioteca **SpringDoc OpenAPI**, que gera automaticamente a documentação da API baseada nos controllers e DTOs existentes. A interface Swagger UI está disponível em `/docs`, permitindo visualizar e testar os endpoints de forma interativa.

A configuração foi realizada por meio do arquivo `application.yml` com personalização do caminho e do título da documentação.

7. Instalação, Execução e Uso da Aplicação

7.1 Requisitos de Ambiente

Para executar a aplicação localmente, é necessário garantir que o ambiente esteja preparado com os seguintes softwares:

- **Java 21**: versão utilizada para compilar e executar o projeto.
- **Maven 3.8+**: ferramenta de build e gerenciamento de dependências.
- **Ngrok**: para expor localmente a aplicação a fim de receber webhooks.
- **Conta de Desenvolvedor HubSpot**: para registrar o app e obter `client_id` e `client_secret`.

7.2 Clonando o Projeto

O código-fonte está disponível em um repositório Git. Para baixá-lo, execute o comando abaixo em seu terminal:

```
git clone https://github.com/andregnicoletti/meetime-case-hubspot.git
cd meetime-case-hubspot
```

7.3 Configuração da Aplicação

A configuração da aplicação é feita através de variáveis de ambiente definidas dentro de um script chamado <code>run.sh</code>, presente na raiz do projeto. Este script é responsável por exportar as variáveis necessárias e iniciar a aplicação de forma automatizada.

Dentro do script `run.sh`, você deve preencher os valores das variáveis conforme abaixo:

```
APPLICATION_PORT=8080
HUBSPOT_CLIENT_ID="sua_client_id_aqui"
HUBSPOT_CLIENT_SECRET="sua_client_secret_aqui"
```

Essas variáveis são utilizadas pela aplicação para determinar a porta de execução, bem como as credenciais necessárias para autenticação com o HubSpot. O script `run.sh` também realiza uma verificação da instalação do `ngrok` e oferece a opção de iniciá-lo automaticamente, facilitando os testes de webhooks.

7.4 Executando a Aplicação

Certifique-se de dar permissão de execução ao script antes da primeira execução com o comando:

```
chmod +x run.sh
```

```
./run.sh
```

Para executar a aplicação, basta rodar o script `run.sh` no terminal. Ele cuidará da exportação das variáveis, verificação do ngrok e execução do JAR:

7.5 Testando a Aplicação

Para interagir com a API, acesse a interface Swagger UI em:

```
http://localhost:8080/docs
```

Nela, você poderá executar os endpoints de forma visual, como:

- `/oauth/authorize`: inicia o fluxo OAuth.
- `/contacts` (POST): cria um novo contato.
- `/contacts` (GET): lista os contatos criados.
- `/webhook` (POST): recebe notificações da HubSpot (necessita ngrok).

7.6 Simulando Webhooks

Para testar o endpoint `/webhook`, execute o Ngrok para expor sua API:

Caso o ngrok já tenha sido iniciado automaticamente pelo script `run.sh`, você pode descobrir a URL pública gerada acessando o painel local em **http://localhost:4040** ou utilizando o comando abaixo no terminal para executar o ngrok:

```
ngrok http 8080
```

Em seguida, configure o webhook no painel do HubSpot apontando para a URL fornecida pelo Ngrok, como:

```
https://random-id.ngrok.io/webhook
```

Você pode também simular manualmente uma chamada via ferramentas como Postman, com o corpo:

```
[
  {
    "eventId": 123456,
    "subscriptionType": "contact.creation",
    "objectId": 987654321
  }
]
```

8. Possíveis Melhorias Futuras

- Implementação de um mecanismo de persistência para os tokens, substituindo o armazenamento em memória.
- Renovação automática do `access_token` utilizando o `refresh_token` antes do vencimento.
- Criação de testes unitários e de integração, utilizando `MockMvc` e `TestRestTemplate`.
- Implementação de persistência de dados de contato em banco de dados.
- Implementação de métricas com Actuator e integração com Prometheus ou Grafana.

9. Conclusão

A aplicação foi construída com foco em clareza, boas práticas de engenharia de software e organização de código. Foram aplicadas abordagens modernas de autenticação, consumo de APIs externas e documentação. O projeto está pronto para evoluções futuras, com base sólida para integrações mais robustas com a HubSpot e adaptação a novos requisitos.