Desafio Técnico LinkApi - Pleno

Instruções

Leia atentamente as instruções abaixo para realização do teste proposto.

- Desenvolva e versione os projetos usando git.
- Utilize o GitHub para hospedar o código em um repositório privado.
- Crie a documentação de como executar e consumir seu projeto.
- Compartilhe este repositório privado como o e-mail **people@linkapi.com.br**.
- Enviar o link do repositório para **people@linkapi.com.br**.

⚠️ IMPORTANTE - CRÍTÉRIOS DE DESCLASSIFICAÇÃO ⚠️

- **O teste em questão deve ser feito utilizando a engine NodeJS (Vanilla JavaScript ou TypeScript), caso contrário, o candidato será desclassificado automaticamente.**
- **O repositório deve estar com o compartilhamento privado, caso contrário, o candidato será desclassificado automaticamente.**

O que será validado?

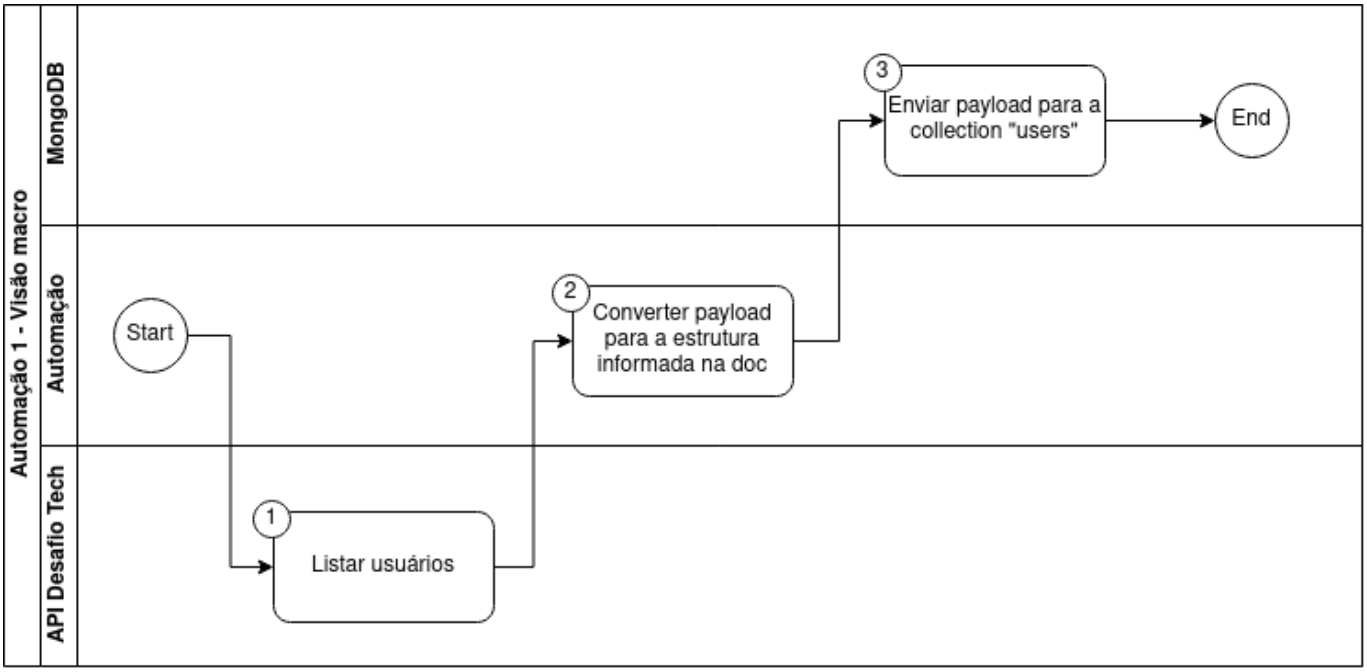
- Requisitos realizados.
- Desacoplamento de código.
- Legibilidade de código.
- Boas práticas de desenvolvimento de API RESTful.
- Performance das aplicações.
- Documentação das aplicações.
- Exposição de dados sensíveis no código

Projeto 1.1 - Automação de Conversão (Obrigatório)

Objetivo

Desenvolvimento de uma automação agendada em NodeJs para realizar o consumo de dados dos usuários em uma API, converter a estrutura de cada usuário e envia-lo para um banco de dados MongoDB na collection “users“.

Fluxograma



Requisitos

1. Criar automação com as regras de negócio e funcionalidade dos steps posteriores.
2. Seu script deverá inicialmente realizar a busca de todos os usuários na API mencionada no final da descrição deste projeto através da rota **"/users"**.
3. Com a resposta da requisição anterior, deverá ser criado um subprocesso para cada usuário retornado.

4. Converter a estrutura de cada usuário para a estrutura abaixo:

```
{
  "fullName":"Kapi Kaperson", // Junção dos campos firstName e lastName do
usuário
  "email":"kapi.kapeira@gmail.com",// Campo email do usuário
  "address":"Rua das Kapivaras",// Campos street do primeiro indice da
listagem de endereços
  "addressNumber":123,// Campo number do primeiro indice da listagem de
endereços
  "phoneNumber":"941-123-555"// Campo phoneNumber do primeiro indice da
listagem de contatos
}
```

5. Criar banco de dados MongoDB.
6. Feito isso deverá ser criada uma collection chamada "users" para inclusão dos usuários.
7. Para cada conversão, realizar a inserção dos dados do usuário na collection "users".

Pontos de atenção

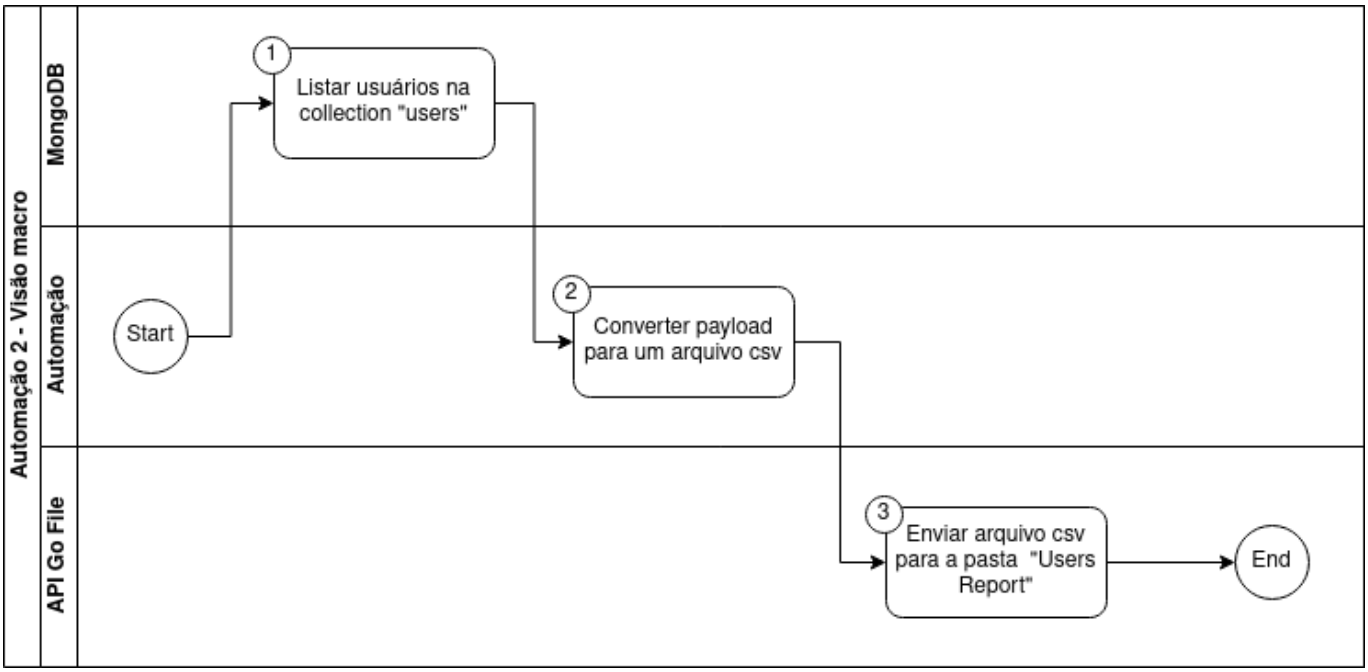
- Para a criação do banco de dados MongoDB, é preferível a utilização de Docker mas existem serviços como MongoDB Atlas para criação de uma instancia gratuitamente que atendem a este requisito.
- Documentação da API Mencionada: Desafio Tech - API Docs
- A API em questão possui recursos de paginação (parâmetros page e limit) e rate limit de 30 reqs/min
- A API possui autenticação do tipo Basic, para a geração do token deverão ser utilizados o seguinte usuário e senha:
    - **Usuário**: 17b271f2-2c76-4240-a0d7-46f57e919ca3
    - **Senha**: 741d5db9-c596-41b4-8785-1d50367224c8

Projeto 1.2 - Automação de envio de arquivos (Opcional)

Objetivo

Desenvolvimento de uma automação agendada em NodeJs para realizar o envio de arquivos gerados a partir dos documentos populados no projeto anterior.

Fluxograma



Requisitos

1. Criar automação com as regras de negócio e funcionalidade dos steps posteriores.
2. Seu script deverá inicialmente realizar a busca de todos os usuários na collection "users" do MongoDB.
3. Com a resposta da busca anterior deverá ser criado um arquivo csv com as seguintes colunas "id", "fullName" (Junção do campo firstName e lastName do usuário), "email".
4. Mandar o arquivo csv gerado para o GoFile, em uma pasta nomeada por "Users Report"

Pontos de atenção

- Para utilizar os recursos da API do GoFile, é necessária a criação de uma conta utilizando e-mail
- A documentação da API GoFile pode ser encontrada em GoFile - API Docs.

Projeto 2 - API (Obrigatório)

Objetivo

Construção de uma API RESTful que realize o gerenciamento de arquivos na API do GoFile.

Endpoints

Endpoint - Criar pastas

Este endpoint deverá receber um nome de pasta, realizar a criação da mesma na GoFile e armazenar o nome e id da pasta criada no banco de dados MongoDB na collection "Folders".

Endpoint - Upload de arquivo

Este endpoint deverá receber um arquivo qualquer e o nome da pasta onde será salvo, e fazer o upload do mesmo na pasta em questão no GoFile, armazenando nome, id do arquivo e id da pasta gerados na collection "Files".

Pontos de atenção

Caso o usuário passe um nome de pasta que ainda não foi criada através da rota de criação de pastas, retornar um erro solicitando que o mesmo execute esta operação

Endpoint - Deleção de arquivo

Este endpoint deverá receber o nome da pasta e o nome do arquivo e deve excluir o mesmo da GoFile.

Pontos de atenção

Caso o usuário passe um nome de pasta ou de arquivo que ainda não existir no banco, retornar um erro solicitando que o mesmo execute estas operações.