Documentação do Backend

Eduardo Garcia Mendonça
Hector Fernando Tarasiuk
Jair Eugenio Ferreira
João Pedro Beltrão Cortes

Índice

1 Mas afinal, o que o back end vai fazer?

- Receber currículo do front end
- Cadastrar um novo currículo
- Deletar um currículo cadastrado
- Mandar currículo pro front end
- Verificar se um currículo já existe no banco de dados
- Verificar se um usuário digitou CPF e senha corretos

1.1 Cadastrar um novo usuário

Um usuário registra todos os seus dados no sistema e então o front end envia esses dados na forma de JSON pro back end. O front end deve verificar se o CPF já foi cadastrado antes de enviar os dados pro backend, ele faz isso pela função verUsuario.

Isso é feito no endpoint nov.

1.2 Mandar dados pro front end

O frontend pode consultar por um currículo já cadastrado. O frontend faz uma busca usando o id do currículo. Se o currículo existe, o backend retorna como JSON o currículo do banco de dados. Se não existe, o backend retorna NULL.

Isso é feito no endpoint cur.

1.3 Editar um usuário

Primeiramente, no front end o usuário digita CPF e senha. O back end então tem que dizer se esse usuário tem ou não autorização de editar os dados. Se o back dizer pro front que sim, o frontend pede os dados do currículo associado ao CPF e vai pra tela de edição, onde da pra editar tudo menos CPF e senha. O front então manda esses dados de volta pro backend pra ser salvos.

Isso é feito no endpoint edit.

1.4 deletar um usuário

Igual no editar o usuário, primeiramente o usuário entra CPF e senha. O frontend verifica se o currículo associado existe no banco de dados e verifica se a senha ta certa. Se existir e a senha estiver correta, o banco de dados retorna autorização pro front end. Quando o front end recebe a autorização, ele faz um pedido pro banco de dados deletar o arquivo.

Isso é feito no endpoint del.

1.5 Verificar se um usuário já existe no banco de dados

Um pedido é enviado ao backend pedindo confirmação sobre se já existe um usuário no banco de dados associado a um CPF.

Isso é feito no endpoint verU.

1.6 Autenticar um usuário

Autentica um usuário pelo CPF e pela senha pra conceder permissão de editar dados do currículo ou de deletar seu currículo do banco de dados.

Isso é feito no endpoint auth.

1.7 Endpoints da API:

1.7.1 nov (POST)

recebe dados de um currículo pra gravar ele no disco.

1.7.2 cur (GET)

retorna os dados de um currículo para o frontend

1.7.3 edit (POST)

edita um currículo já existente com dados novos

1.7.4 del (POST)

diz pro backend deletar um currículo se o CPF e senha fornecidos forem válidos.

1.7.5 auth (POST)

verifica se um usuário existe e forneceu a senha correta.

1.7.6 verU (GET)

verifica se um usuário já existe no banco de dados

2 Documentação do código fonte do backend

O backend é uma API RESTful feita com spring-boot. Ele faz uso da biblioteca simplesDB para interagir com o banco de dados.

Para rodar compilar o backend, é necessário primeiro instalar a biblioteca simplesDB. isso pode ser feito pelo conveniente script shell instala_biblioteca.sh no diretório backend. Rode:

```
./instala_biblioteca.sh
```

Pode se rodar o backend pelo comando em seu diretório:

```
./mvnw spring-boot:run
```

ou pelo arquivo jar no diretório principal do trabalho final. pode-se compilar um novo .jar executável por meio do comando

```
./mvnw clean package
```

O novo executável aparecerá no diretório targets.

Segue agora uma descrição dos conteúdos e funcionalidades de cada arquivo do código fonte.

2.1 curriculoBackApplication.java

A main do backend está neste arquivo. Ele não faz muito além de definir o diretório onde estão armazenados os dados.

2.2 authController.java

Cria e controla o endpoint POST responsável pela autenticação (/auth). Faz uso da função autenticaUsuario() de simplesDB para isso. Retorna um booleano que pode ser usado pelo frontend para saber se a operação foi um sucesso ou um fracasso.

2.3 cpfSenha.java

Define um record publico cpfSenha, representa os dados que o frontend enviaria para o backend para operações que necessitam de autenticação.

2.4 curController.java

Cria e controla o endpoint GET responsável por mandar os dados relacionados a um ID de usuário ao frontend (/cur). Faz uso das funções carregaCurriculo() e curriculoPraEnviar() de simplesDB para isso. Retorna um objeto json com os dados do currículo ou um null.

2.5 delController.java

Cria e controla o endpoint POST responsável por deletar um usuário (/del). Remove um currículo do banco de dados se o CPF e Senha providos pelo request forem válidos. Faz uso da função delUsuario() de simplesDB. Retorna um booleano para o frontend.

2.6 editController.java

Cria e controla o endpoint POST responsável por editar dados de um usuário (/edit). faz uso da função editaUsuario() e simplesDB. retorna um booleano.

2.7 novController.java

Cria e controla o endpoitn POST responsável por criar um novo currículo no banco de dados (/nov). Faz uso da função novoUsuario() de simplesDB. Retorna um booleano.

2.8 verUcontroller.java

Cria e controla o endpoint GET responsável por verificar se já existir algum currículo no banco de dados associado ao CPF fornecido. faz uso da função noDiscoCPF() de simpesDB. Retorna um booleano.