Seleção Dev Java

Se você chegou até aqui é porque se interessou em fazer parte do nosso quadro de funcionários. Como temos muitas oportunidades para você colocar a mão na massa, queremos ver como você se sai com o cenário abaixo, por meio do qual conseguiremos avaliar várias de suas competências.

A demanda

Deverá ser criada uma aplicação de gerenciamento de **autores** e **obras**, seguindo as regras de relacionamento abaixo: - Cada autor poderá ter 0 (zero) ou n obra(s); - Cada obra deverá ter 1 (um) ou n autor(es); - A partir de uma obra deverá ser posível acessar o(s) autor(es); - A partir de um autor deverá ser possível acessar a(s) obra(s).

1) Back-end

A aplicação, a ser desenvolvida em Java, deverá expor uma API REST de cadastro, alteração, remoção e consulta de **autores** e **obras** com as seguintes propriedades básicas definidas para cada entidade:

Autor

- Nome - obrigatório
- Sexo
- E-mail - não obrigatório, deve ser validado caso preenchido (não pode haver dois cadastros com mesmo e-mail)
- Data de nascimento - obrigatório, deve ser validada
- País de origem - obrigatório (deve ser um país existente)
- CPF - somente deve ser informado caso país de origem seja o Brasil, desta forma torna-se obrigatório. Deve ser validado (formatado e não pode haver dois cadastros com mesmo CPF)

Obra

- Nome - obrigatório
- Descrição - obrigatório (deve conter no máximo 240 caracteres)
- Data de publicação - obrigatória caso a data de exposição não seja informada (é utilizada mais para livros e demais publicações escritas)
- Data de exposição - obrigatória caso a data de publicação não seja informada (é utilizada mais para obras que são expostas, como pinturas, esculturas e demais)

Regra(s)

- A data de publicação e a data de exposição não podem ser nulas ao mesmo tempo, devendo sempre uma ou outra ser informada.

2) Front-end

A aplicação deverá ser acessível via navegador e possuir telas com formulários para cadastro de autores e obras.

Autor

- **Cadastro**: na tela de cadastro e edição dos **autores**, além de informar seus atributos básicos, deverá ser possível associar qual(is) **obra(s)** (cadastradas no sistema) são de sua autoria.
- **Consulta**: na tela de consulta dos **autores** deverá haver uma tabela onde serão listados os autores cadastrados. #### Obra
- **Cadastro**: na tela de cadastro e edição das **obras**, além de informar seus atributos básicos, deverá ser possível associar seu(s) **autor(es)** (cadastrados no sistema).
- **Consulta**: na tela de consulta das **obras** deverá haver uma tabela onde serão listadas as obras cadastradas.

Regras de exclusão

- **Autor**: somente pode ser excluído caso não possua **obras** associadas.
- **Obra**: não há restrição para exclusão.

A sua aplicação front-end deverá consumir sua API back-end. Não há restrição em relação à tecnologia para o desenvolvimento do front-end.

3) Instalação

Um manual com as instruções de instalação do projeto deve ser produzido. O manual deve conter os processos necessários para executar a aplicação. **Este passo não se faz necessário caso o Extra 5 seja implementado**.

Extras

Se você possui conhecimento um pouco mais avançado, pode implementar as tarefas abaixo que serão consideradas como diferencial na sua análise. Atente-se para o prazo de entrega ;) * **Extra 1 (autenticação/segurança):** implementar algum tipo de autenticação, como a **basic** ou outra de sua preferência (o usuário pode ser fixo no banco, caso prefira, não precisando que a aplicação envolva a criação de uma conta). * **Extra 2 (testes):** implementar testes de unidade utilizando JUnit. Os testes devem contemplar as operações abaixo validando as restrições informadas anteriormente: * **Inserção**: validar regras informadas anteriormente (ex.: obrigatoriedade dos campos em cada situação). * **Edição**: manter a consistência garantida anteriormente na inserção, impedindo que a entidade seja modificada para um estado inválido (ex.: **obra** sem **autor**). * **Consulta**: validar os dados retornados do service (pode-se utilizar mock's). * **Exclusão**: validar regras de exclusão informadas anteriormente. * **Extra 3 (padrão de projeto):** implementar algum padrão de projeto que

auxilie o desenvolvimento, de forma a melhorar a manutenabilidade do código, diminuir a replicação e aumentar o reuso. * **Extra 4 (filtragem e paginação):** Implementar paginação e filtragem de **obra(s)**. Deve ser possível realizar a filtragem das **obra(s)** pelos atribiutos ***nome*** e ***descrição***. Deve também ser criado no front-end da consulta das **obra(s)** o *input* para realizar a filtragem. * **Extra 5 (deploy/hospedagem):** tornar a aplicação disponível em algum ambiente de nuvem, como o Heroku: * Não deverá ser necessária nenhuma configuração por parte do usuário para usar a aplicação, bastando apenas acessar a URL onde o sistema está hospedado. * O sistema deve estar funcional ao ser acessado. ### Instruções - Faça o fork do desafio; - Crie um repositório privado no Gitlab para o projeto e adicione como colaborador o usuário **selecao.stefanini.cpg**; - Desenvolva. Você terá 4 (quatro) dias a partir da data do envio do desafio; - Após concluir seu trabalho faça um push para o repositório que você criou; - Envie um e-mail a equipe de seleção da Stefanini Campina Grande (selecao.cpg@stefanini.com) notificando a finalização do desafio para validação.

Bom desenvolvimento!