Módulo 7 - AngularJS

Nível 3 - Programa de Estágio - Mirante Tecnologia

Referências

- Spring Boot
- Spring Framework
- Spring Data JPA
- AngularJS
- AngularJS Style Guide
- AngularJS UI Router
- Restangular
- Frontend Maven Plugin

Instruções

1. O exercício consiste em criar um cadastro de automóveis.
2. Utilize o esqueleto do projeto como base para o projeto, mas seja livre para alterar as classes e arquivos existentes ou criar novos. Apenas siga a arquitetura indicada.
3. Configure as bibliotecas de *JavaScript* (*AngularJS*, *Bootstrap*, ...) utilizando o *Frontend Maven Plugin* (configurado para trabalhar com *NPM*).
4. Construa um cadastro de carros com um formulário e uma tabela.
5. O formulário deve possuir os campos obrigatórios **modelo**, **placa**, **tração**, **categoria** e **fabricante**.
6. O campo **modelo** deve ser um *autocomplete* que mostra os modelos já cadastrados para serem selecionados. Se o usuário digitar um modelo que não existe, deve ser cadastrado um novo modelo ao salvar os dados carro.
7. O campo **fabricante** deve mostrar o nome do fabricante, o país do fabricante (somente leitura) e um botão **pesquisar**.
8. Ao clicar no botão **pesquisar** do fabricante, deve ser aberta uma *modal* com um *textfield* de **filtro** e uma tabela mostrando os fabricantes (nome e país). A tabela deve mostrar om botão **selecionar** para cada fabricante. Ao digitar no *textfield* de **filtro**, a tabela deve ser filtrada pelo nome do fabricante. Se o usuário digitar um fabricante que não existe, deverá ser mostrado um campo **país** e um botão **adicionar**, que irá acionar o fabricante e selecioná-lo.
9. A tração deve ser um *radio button*, com os valores **combustão** e **elétrico**.
10. A categoria deve ser um *combobox*, com os valores **particular**, **aluguel** e **oficial**.
11. A tabela de consulta deve mostrar os carros cadastrados com os campos **modelo**, **placa**, **tração**, **categoria** e **fabricante** (nome e país).
12. Acrescente na página de consulta um filtro com os campos **modelo**, **placa**, **tração**, **categoria** e **fabricante** (nome). Esse filtro deverá ser aplicado à tabela de consulta automaticamente, sem clicar em nenhum botão.

13. A **placa** deve ser única, isto é, não devem ser cadastrados carros com placas iguais.
14. A **placa** deve estar no formato **XXX9999** (três letras e quatro números).
15. Não devem ser cadastrados carros com placas com formato inválido.
16. A tabela deve mostrar uma opção de **alterar** e **excluir** para cada carro.
17. Ao clicar em **alterar**, o formulário deve ser carregado com os dados do carro e, ao clicar em salvar no formulário, os dados do carro devem ser atualizados na tabela.
18. Ao clicar em **excluir**, os dados do carro devem ser retirados do cadastro.
19. O formulário deve mostrar o botão **cadastrar** quando for inclusão e **salvar** quando for alteração.
20. Quando não houver carros cadastrados, mostre a mensagem **nenhum carro cadastrado** ao invés da tabela.
21. Mostre mensagens bem definidas e individuais de todos os erros que ocorrerem no cadastro.
22. Crie uma página de template para fazer o roteamento.
23. Não inclua arquivos na pasta **src/main/webapp**.
24. Adicione uma máscara ao campo **placa** para que o usuário só consiga digitar os números e as letras na posição desejada.
25. O programa deve funcionar com a partir das seguintes *URLs*:
    1. http://localhost:8080/#/home

       Home page.
    2. http://localhost:8080/#/carros

       Consulta de carros cadastrados.
    3. http://localhost:8080/#/carro

       Cadastro de um novo carro.
    4. http://localhost:8080/#/carro/1

       Alteração do carro com id 1. O número pode variar na *URL*.
26. O projeto deve utilizar o *Lombok*.

    http://projectlombok.org
27. Não utilize a diretiva **ng-controller** nas páginas.
28. Trabalhe com componentização dos elementos utilizando a função **angular.component()**.
29. Organize seu código de acordo com o *AngularJS Style Guide* do John Papa.
30. Utilize o *AngularJS UI Router* para criar as rotas (URLs) do sistema.
31. Crie os *web services* no servidor pelo *Spring MVC* através da anotação **@RestController**.
32. Utilize o *Restangular* para acessar os *web services* no servidor.