

Bootcamp: Desenvolvimento Front End

Desafio do Módulo

Módulo 1	Fundamentos
-----------------	--------------------

Objetivos

Exercitar os seguintes conceitos trabalhados no Módulo:

- ✓ Declarar elementos HTML como títulos, parágrafos, inputs, divs, spans etc.
- ✓ Estilizar o app com CSS.
- ✓ Mapear elementos do DOM para serem manipulados com JavaScript.
- ✓ Realizar diversos cálculos com array methods como map, filter, some, forEach e includes.
- ✓ Realizar requisições HTTP com o comando fetch e utilização de promises ou async/await.
- ✓ Sair da zona de conforto e pensar fora da caixa.




Enunciado

Criar uma aplicação para pesquisar desenvolvedores (dev's) com opções de filtragem.

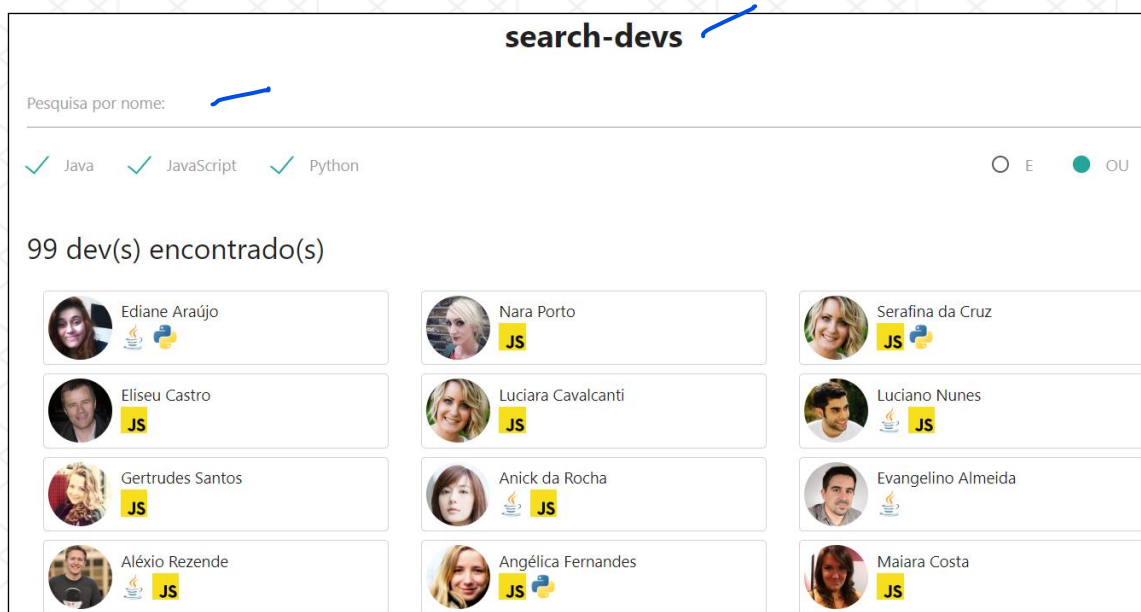
Atividades

Os alunos deverão desempenhar as seguintes atividades:

1. "Subir" o backend, que será fornecido pelo professor e contém um arquivo JSON com **99** dev's já definidos.
2. No frontend, carregar os dados dos dev's em memória através de acesso ao backend com o comando **fetch**.

3. Permitir a filtragem de dev's a partir de trechos do nome, através de um input de texto com interação do usuário, considerando o texto em minúsculo sem espaços em branco e desconsiderando acentos. Assim, o termo de busca "dre", deve trazer os dev's Andréia e André, por exemplo.
4. Permitir a filtragem de dev's através das linguagens de programação. Há dev's cadastrados que conhecem Java, JavaScript e/ou Python. Essa filtragem deve considerar os operadores lógicos E ou OU. Assim, se eu pesquiso por dev's que conhecem Java E Python, somente os dev's que conhecem ambas as linguagens devem ser retornados. Se eu pesquiso com OU, os dev's que conhecem somente uma das linguagens também devem ser retornados.
5. Para representar as linguagens de programação dos dev's, foram utilizados ícones, que também serão fornecidos pelo professor, sendo:
 - a. Java. 
 - b. JavaScript. 
 - c. Python. 
6. As imagens abaixo ilustram exemplos de interface da aplicação em algumas situações possíveis. Foi utilizado o [Materialize](#) como ferramenta de CSS.

Tela inicial da aplicação (sugestão):



Considerando os dados fornecidos, não existe nenhum dev que conheça **Java E JavaScript E Python**:

search-devs

Pesquisa por nome:

☒ Java
 ☒ JavaScript
 ☒ Python
 ● E ○ OU

0 dev(s) encontrado(s)


Dev's que conhecem **Java E JavaScript** (são mais que 3):

search-devs


Pesquisa por nome:


☒ Java
 ☒ JavaScript
 ☐ Python
 ● E ○ OU

dev(s) encontrado(s)





Luciano Nunes






Anick da Rocha





Aléxio Rezende




Dev's que conhecem **Java OU JavaScript** (são mais que 3):

search-devs



Pesquisa por nome:


☒ Java
 ☒ JavaScript
 ☐ Python
 ○ E ● OU

dev(s) encontrado(s)





Ediane Araújo





Nara Porto





Serafina da Cruz

Perceba também a variação da combinação de linguagens de programação, pois há dev's que conhecem somente JavaScript e dev's que conhecem JavaScript E Python, por exemplo. Isso é válido quando **OU** está selecionado.

Busca com o texto "or", que filtrou Nara Porto, Aléxioo Rezende (desconsiderando espaços em branco) e Loreta Nascimento, por exemplo:

The screenshot shows a web application titled "search-devs". At the top, there is a search bar with the placeholder text "Pesquisa por nome:" and the input "or". Below the search bar, there are three checkboxes for programming languages: "Java" (checked), "JavaScript" (checked), and "Python" (unchecked). To the right of these checkboxes are two radio buttons: "E" (selected) and "OU" (unselected). Below the search bar, there is a red square icon followed by the text "dev(s) encontrado(s)". At the bottom, there are three profile cards. The first card is for "Nara Porto" with a profile picture and a "JS" tag. The second card is for "Aléxio Rezende" with a profile picture and a "JS" tag. The third card is for "Loreta Nascimento" with a profile picture and a "JS" tag.

Dicas e sugestões

1. Lembre-se sempre de que há várias formas de se implementar um problema.
2. Após executar a requisição à API, faça uma transformação de dados com **map** e crie um campo adicional para usar como **busca**. Esse campo deve **converter o nome para minúsculas e retirar os espaços em branco**. Use esse campo adicional para localizar os dev's quando o usuário digitar no input.
3. Dê **prioridade à funcionalidade** e só depois dedique-se à **interface**.
4. A função **split** pode converter uma **string** em **array**. A função **join** pode converter um **array** em **string**.
5. Utilize o evento **input** para filtrar os dados a partir da digitação do usuário.
6. Na minha opinião, a lógica mais complexa deste app é a filtragem de dev's utilizando **OR**. Para fazer isso, pense em conjuntos. Utilizando **OR**, **pelo menos uma das linguagens marcadas** (**array.some**) deve estar presente no conjunto de todas as linguagens (**array.includes**).
7. Tente ao máximo implementar o desafio **sozinho**, com o apoio do **fórum**. Em **último caso**, deixo um projeto bastante semelhante ao desafio **neste link**, que pode servir

de apoio e sanar algumas dúvidas. **Atenção: tenha certeza de que você vai aprender muito mais se implementar este desafio sozinho.**