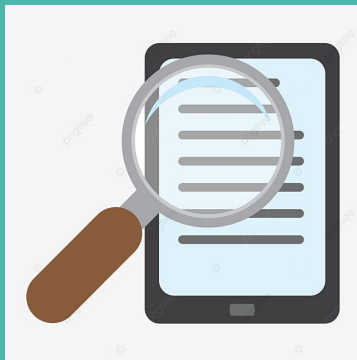

Desenvolvimento WEB

-

Full Stack Completo: Java + React

Implementando Filtros de Pesquisa no Back-end



Implementando Filtros de Pesquisa no Back-end

Objetivo: implementar um filtro na tela de listagem de produtos:

Home

Cliente

Produto ▾

Entregador

Comprador

Cupom Desconto

Produto

Filtrar

Novo

Código do Produto

Filtrar por Código do Produto

Título

de||

Categoria

Filtrar por Categoria

Código	Categoria	Título	Descrição	Valor Unitário	Tempo Mínimo de Entrega	Tempo Máximo de Entrega	Ações
005	Notebook	Notebook Dell	Linha 2023	1500	30	120	<div><div></div><div></div></div>

Implementando Filtros de Pesquisa no Back-end

No arquivo `ProdutoRepository`, implemente 4 novas funções para representar as opções de pesquisa no filtro da tela de listagem:

```
public interface ProdutoRepository extends JpaRepository<Produto, Long> {

    //Exemplo de uma busca exata
    @Query(value = "SELECT p FROM Produto p WHERE p.codigo = :codigo")
    List<Produto> consultarPorCodigo(String codigo);

    //Exemplo de uma busca aproximada com ordenação:
    // @Query(value = "SELECT p FROM Produto p WHERE p.titulo like %:titulo% ORDER BY p.titulo")
    // List<Produto> consultarPorTitulo(String titulo);
    List<Produto> findByTituloContainingIgnoreCaseOrderByTituloAsc(String titulo);

    //Exemplo de uma busca exata como um atributo de relacionamento
    @Query(value = "SELECT p FROM Produto p WHERE p.categoria.id = :idCategoria")
    List<Produto> consultarPorCategoria(Long idCategoria);

    //Exemplo de uma busca com mais de um atributo
    @Query(value = "SELECT p FROM Produto p WHERE p.titulo like %:titulo% AND p.categoria.id = :idCategoria")
    List<Produto> consultarPorTituloECategoria(String titulo, Long idCategoria);

}
```

Implementando Filtros de Pesquisa no Back-end

- Para aprender mais opções de métodos de repositório utilizando o `@Query` do Spring Data JPA:

<https://www.baeldung.com/spring-data-jpa-query>

Implementando Filtros de Pesquisa no Back-end

No
implemente o método filtrar:

```
public List<Produto> filtrar(String codigo, String titulo, Long idCategoria) {  
  
    List<Produto> listaProdutos = repository.findAll();  
  
    if ((codigo != null && !"".equals(codigo)) &&  
        (titulo == null || "".equals(titulo)) &&  
        (idCategoria == null)) {  
        listaProdutos = repository.consultarPorCodigo(codigo);  
    } else if (  
        (codigo == null || "".equals(codigo)) &&  
        (titulo != null && !"".equals(titulo)) &&  
        (idCategoria == null)) {  
        listaProdutos = repository.findByTituloContainingIgnoreCaseOrderByTituloAsc(titulo);  
    } else if (  
        (codigo == null || "".equals(codigo)) &&  
        (titulo == null || "".equals(titulo)) &&  
        (idCategoria != null)) {  
        listaProdutos = repository.consultarPorCategoria(idCategoria);  
    } else if (  
        (codigo == null || "".equals(codigo)) &&  
        (titulo != null && !"".equals(titulo)) &&  
        (idCategoria != null)) {  
        listaProdutos = repository.consultarPorTituloECategoria(titulo, idCategoria);  
    }  
  
    return listaProdutos;  
}
```

Implementando Filtros de Pesquisa no Back-end

No arquivo `ProdutoController`, implemente o método `filtrar` que define uma rota a ser utilizada nas telas que precisarem filtrar produtos por código, título e/ou categoria de produto:

```
@PostMapping("/filtrar")
public List<Produto> filtrar(
    @RequestParam(value = "codigo", required = false) String codigo,
    @RequestParam(value = "titulo", required = false) String titulo,
    @RequestParam(value = "idCategoria", required = false) Long idCategoria) {

    return produtoService.filtrar(codigo, titulo, idCategoria);
}
```

Implementando Filtros de Pesquisa no Front-end

Home Cliente Produto ▾ Entregador Comprador Cupom Desconto

Produto

Filtrar

Novo

Código do Produto



Filtrar por Código do Produto

Título

de

Categoria

Filtrar por Categoria

Código	Categoria	Título	Descrição	Valor Unitário	Tempo Mínimo de Entrega	Tempo Máximo de Entrega	Ações
005	Notebook	Notebook Dell	Linha 2023	1500	30	120	 

Implementando Filtros de Pesquisa no Front-end

1) No arquivo `ListProduto.jsx`, acrescente o seguinte código **em vermelho**:

```
export default function ListProduto () {  
  
  const [listaProdutos, setListaProdutos] = useState([]);  
  const [openModal, setOpenModal] = useState(false);  
  const [idRemover, setIdRemover] = useState();  
  const [menuFiltro, setMenuFiltro] = useState();  
  const [codigo, setCodigo] = useState();  
  const [titulo, setTitulo] = useState();  
  const [idCategoria, setIdCategoria] = useState();  
  const [listaCategoriaProduto, setListaCategoriaProduto] = useState([]);
```

Implementando Filtros de Pesquisa no Front-end

2) No arquivo `ListProduto.jsx`, acrescente o seguinte código **em vermelho**:

```
function carregarLista () {  
  
    axios.get("http://localhost:8080/api/produto")  
    .then((response) => {  
        setListaProdutos(response.data);  
    })  
  
    axios.get("http://localhost:8080/api/categoriaproduto")  
    .then((response) => {  
  
        const dropDownCategorias = [];  
        dropDownCategorias.push({ text: '', value: '' });  
        response.data.map(c => (  
            dropDownCategorias.push({ text: c.descricao, value: c.id })  
        ))  
  
        setListaCategoriaProduto(dropDownCategorias)  
  
    })  
}
```

Implementando Filtros de Pesquisa no Front-end

3) No
acrescente as seguintes funções:

```
function handleMenuFiltro() {  
  
    if (menuFiltro === true) {  
        setMenuFiltro(false);  
    } else {  
        setMenuFiltro(true);  
    }  
}  
  
function handleChangeCodigo(value) {  
  
    filtrarProdutos(value, titulo, idCategoria);  
}  
  
function handleChangeTitulo(value) {  
  
    filtrarProdutos(codigo, value, idCategoria);  
}  
  
function handleChangeCategoriaProduto(value) {  
  
    filtrarProdutos(codigo, titulo, value);  
}
```

Implementando Filtros de Pesquisa no Front-end

4) No
acrescente a seguinte função:

```
async function filtrarProdutos(codigoParam, tituloParam, idCategoriaParam) {  
  
    let formData = new FormData();  
  
    if (codigoParam !== undefined) {  
        setCodigo(codigoParam)  
        formData.append('codigo', codigoParam);  
    }  
  
    if (tituloParam !== undefined) {  
        setTitulo(tituloParam)  
        formData.append('titulo', tituloParam);  
    }  
  
    if (idCategoriaParam !== undefined) {  
        setIdCategoria(idCategoriaParam)  
        formData.append('idCategoria', idCategoriaParam);  
    }  
  
    await axios.post("http://localhost:8080/api/produto/filtrar", formData)  
    .then((response) => {  
        setListaProdutos(response.data)  
    })  
}
```

Implementando Filtros de Pesquisa no Front-end

5) No arquivo `ListProduto.jsx`, acrescente o seguinte código **em vermelho**:

```
...  
  
    <Container textAlign='justified' >  
  
        <h2> Produto </h2>  
  
        <Divider />  
  
        <div style={{marginTop: '4%'}}>  
  
            <Menu compact>  
                <Menu.Item  
                    name='menuFiltro'  
                    active={menuFiltro === true}  
                    onClick={() => handleMenuFiltro()}  
                >  
  
                    <Icon name='filter' />  
  
                    Filtrar  
  
                </Menu.Item>  
            </Menu>  
  
        </div>  
  
    </Container>  
  
...
```

Implementando Filtros de Pesquisa no Front-end

6) No arquivo `ListProduto.jsx`, acrescente o seguinte código **em vermelho**:

```
...  
<Button  
  label='Novo'  
  circular  
  color='orange'  
  icon='clipboard outline'  
  floated='right'  
  as={Link}  
  to='/form-produto'  
>  
  
{ menuFiltro ?  
  
  <Segment>  
    <Form className="form-filtros">  
  
      <Form.Input  
        icon="search"  
        value={codigo}  
        onChange={e => handleChangeCodigo(e.target.value)}  
        label='Código do Produto'  
        placeholder='Filtrar por Código do Produto'  
        labelPosition='left'  
        width={4}  
      />  
    </Form>  
  </Segment>  
}
```

```
    <Form.Group widths='equal'>  
      <Form.Input  
        icon="search"  
        value={titulo}  
        onChange={e => handleChangeTitulo(e.target.value)}  
        label='Titulo'  
        placeholder='Filtrar por título'  
        labelPosition='left'  
      />  
      <Form.Select  
        placeholder='Filtrar por Categoria'  
        label='Categoria'  
        options={listaCategoriaProduto}  
        value={idCategoria}  
        onChange={ (e, {value}) => {  
          handleChangeCategoriaProduto(value)  
        }}  
      />  
    </Form.Group>  
  </Form>  
</Segment>: ""  
}  
<br/><br/>  
<Table color='orange' sortable celled>  
  ...
```

Implementando Filtros de Pesquisa no Front-end

7) Faça o teste do filtro:

Home

Cliente

Produto ▾

Entregador

Comprador

Cupom Desconto

Produto

🔼 Filtrar

Novo

Código do Produto

Filtrar por Código do Produto

🔍

Título

dell

🔍

Categoria

Filtrar por Categoria

▾

Código	Categoria	Título	Descrição	Valor Unitário	Tempo Mínimo de Entrega	Tempo Máximo de Entrega	Ações
005	Notebook	Notebook Dell	Linha 2023	1500	30	120	<div><div>✎</div><div>🗑</div></div>

© 2023 - Projeto WEB III - IFPE Jaboatão dos Guararapes

Dúvidas



Exercício

Implemente:

- 1) Um filtro de pesquisa de acordo com o exemplo visto na aula para a tela de listagem de produtos;
- 2) Um filtro na tela de listagem de cliente que possibilita filtrar os clientes pelo nome e pelo CPF;



The background features two large, solid green abstract shapes. One is a semi-circle on the left side, and the other is a more complex, organic shape on the right side.

Obrigado !