🫡 Guia 1

## 

## 

## 

4°Bimestre (SPOLBP1)

## 

## 

## 

#### 

#### 

#### 

#### 

#### [**Sumario**](#_31p8get12ull)

#### 

[**Atividade de Pesquisa e Desenvolvimento: Coleção de Itens, Paginação e Página de Detalhes com Interações sem Carregamento de Página 3**](#_exn65f33di5e)

[1. Coleção de Itens (Listagem de Produtos) 3](#_lbq94jhlgtw2)

[2. Paginação (Exibindo Páginas de Itens) 4](#_8p9szmp5zy5q)

[3. Página de Detalhes de um Item (Visualizar Informações Detalhadas) 5](#_c5rjcp25az54)

[4. Interações Sem Carregamento de Página (AJAX) 6](#_8myt7ipg1zg)

## 

## 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

### 

### 

# 

# Atividade de Pesquisa e Desenvolvimento: Coleção de Itens, Paginação e Página de Detalhes com Interações sem Carregamento de Página

## 1. Coleção de Itens (Listagem de Produtos)

* **O que é?**A coleção de itens é uma lista de produtos ou dados que será apresentada ao usuário. Cada item da coleção deve ter informações como nome, descrição e preço. A ideia é mostrar vários itens em uma única página.
* **Implementação no Flask:** No Flask, criamos uma rota que retorna uma lista de produtos (ou itens) como JSON. Esses dados são então consumidos por JavaScript para exibição na página.

**Exemplo de Código:**

from flask import Flask, jsonify

app = Flask(\_\_name\_\_)

produtos = [

{"id": 1, "nome": "Produto 1", "descricao": "Descrição do Produto 1", "preco": 100.0},

{"id": 2, "nome": "Produto 2", "descricao": "Descrição do Produto 2", "preco": 150.0},

{"id": 3, "nome": "Produto 3", "descricao": "Descrição do Produto 3", "preco": 200.0},

# ... mais produtos

]

@app.route('/produtos', methods=['GET'])

def listar\_produtos():

return jsonify(produtos)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

## 2. Paginação (Exibindo Páginas de Itens)

* **O que é?**A paginação divide a lista de itens em várias páginas, para melhorar a navegação e a desempenho da aplicação. Em vez de carregar todos os itens de uma vez, a aplicação exibe apenas um número limitado de itens por vez.
* **Implementação no Flask:** Para a paginação, podemos usar a função paginate do SQLAlchemy ou simplesmente implementar a lógica de páginas manualmente. Aqui, vamos simular a paginação.

**Exemplo de Código:**  
@app.route('/produtos/<int:pagina>', methods=['GET'])

def listar\_produtos\_paginados(pagina):

produtos\_por\_pagina = 5

inicio = (pagina - 1) \* produtos\_por\_pagina

fim = inicio + produtos\_por\_pagina

produtos\_pagina = produtos[inicio:fim]

return jsonify(produtos\_pagina)

JS

**JavaScript para Navegação entre Páginas:**  
let paginaAtual = 1;

function carregarProdutos(pagina) {

fetch(`/produtos/${pagina}`)

.then(response => response.json())

.then(data => {

let listaProdutos = document.getElementById("listaProdutos");

listaProdutos.innerHTML = "";

data.forEach(produto => {

listaProdutos.innerHTML += `<li>${produto.nome} - ${produto.preco}</li>`;

});

});

}

document.getElementById("proximaPagina").addEventListener('click', () => {

paginaAtual++;

carregarProdutos(paginaAtual);

});

document.getElementById("paginaAnterior").addEventListener('click', () => {

if (paginaAtual > 1) {

paginaAtual--;

carregarProdutos(paginaAtual);

}

});

carregarProdutos(paginaAtual); // Carrega os produtos da página 1

## 3. Página de Detalhes de um Item (Visualizar Informações Detalhadas)

* **O que é?**A página de detalhes exibe mais informações sobre um produto quando o usuário clica em um item da lista. Isso pode ser feito sem recarregar a página, usando JavaScript para fazer uma requisição AJAX e mostrar os detalhes no mesmo layout.
* **Implementação no Flask:** A rota /produto/<id> retorna os detalhes de um item específico em formato JSON.

**Exemplo de Código:**

@app.route('/produto/<int:id>', methods=['GET'])

def detalhes\_produto(id):

produto = next((p for p in produtos if p['id'] == id), None)

if produto:

return jsonify(produto)

return jsonify({"erro": "Produto não encontrado"}), 404

**JavaScript para Carregar Detalhes do Produto:**

document.querySelectorAll(".verDetalhes").forEach(button => {

button.addEventListener('click', function() {

const produtoId = this.getAttribute('data-id');

fetch(`/produto/${produtoId}`)

.then(response => response.json())

.then(produto => {

if (produto.erro) {

alert(produto.erro);

} else {

// Atualizar o modal com os detalhes do produto

document.getElementById("produtoNome").textContent = produto.nome;

document.getElementById("produtoDescricao").textContent = produto.descricao;

document.getElementById("produtoPreco").textContent = produto.preco;

document.getElementById("modalProduto").style.display = "block";

}

});

});

});

#### 

## 4. Interações Sem Carregamento de Página (AJAX)

* **O que é?**Com AJAX (JavaScript Assíncrono), podemos atualizar partes da página sem precisar recarregar a página inteira. Isso é útil para atualizar listas de itens, detalhes de produtos e outros componentes de forma dinâmica.
* **Exemplo de Interação Sem Recarregamento:** No código acima, a requisição para carregar os detalhes do produto ou listar os produtos é feita de forma assíncrona, o que permite que a página não seja recarregada ao exibir os detalhes ou trocar de página na lista de produtos.

**Exemplo de Exibição de Modal:** Ao clicar no botão "Ver Detalhes" de um produto, o modal exibe as informações sem recarregar a página, como mostrado abaixo:

<div id="modalProduto" style="display:none;">

<h2 id="produtoNome"></h2>

<p id="produtoDescricao"></p>

<p id="produtoPreco"></p>

<button onclick="document.getElementById('modalProduto').style.display='none'">Fechar</button>

</div>