**🛒 Atividade: Gerenciamento de Estado com Vuex para a Loja Virtual da Empresa Q**

A empresa Q está desenvolvendo uma loja virtual com **Vue.js** e precisa de uma forma eficiente de gerenciar o **estado global da aplicação**, como:

* Carrinho de compras 🛍️
* Detalhes do usuário 👤
* Inventário de produtos 📦

A solução escolhida: **Vuex**, a biblioteca oficial de gerenciamento de estado para Vue.js.

**🎯 Objetivos**

* Configurar e integrar o Vuex ao projeto
* Criar a estrutura base com state, mutations, actions e getters
* Implementar o gerenciamento do carrinho de compras
* Testar e validar a funcionalidade com um componente Vue

**🧰 Materiais Necessários**

* 💻 Computador com acesso à internet
* 📓 Caderno para anotações
* 🖊️ 1 caneta

**🗂️ Estrutura do Projeto**

css

CopiarEditar

src/

├── components/

│ └── ShoppingCart.vue

├── store/

│ └── index.js

├── App.vue

└── main.js

**🛠️ Parte 1: Configurando o Vuex**

**1.1 Instalação**

No terminal, dentro do seu projeto Vue:

bash

CopiarEditar

npm install vuex

# ou, se preferir:

yarn add vuex

**1.2 Criando a Store**

Crie uma pasta chamada store dentro de src, e adicione o arquivo index.js com a estrutura abaixo:

js

CopiarEditar

// src/store/index.js

import Vue from 'vue';

import Vuex from 'vuex';

Vue.use(Vuex);

export default new Vuex.Store({

state: {

cart: [],

},

mutations: {

ADD\_TO\_CART(state, product) {

state.cart.push(product);

},

},

actions: {

addToCart({ commit }, product) {

commit('ADD\_TO\_CART', product);

},

},

getters: {

cartItemCount: state => state.cart.length,

}

});

**1.3 Integração com o Vue**

No main.js, importe a store e integre-a ao Vue:

js

CopiarEditar

// src/main.js

import Vue from 'vue';

import App from './App.vue';

import store from './store';

new Vue({

render: h => h(App),

store, // Integrando Vuex

}).$mount('#app');

**🧩 Parte 2: Implementando Funcionalidades**

**2.1 Estado Inicial**

js

CopiarEditar

state: {

cart: [], // Armazena os itens do carrinho

}

**2.2 Mutation para adicionar item ao carrinho**

js

CopiarEditar

mutations: {

ADD\_TO\_CART(state, product) {

state.cart.push(product);

},

}

**2.3 Ação para disparar a mutation**

js

CopiarEditar

actions: {

addToCart({ commit }, product) {

commit('ADD\_TO\_CART', product);

},

}

**2.4 Getter para total de itens**

js

CopiarEditar

getters: {

cartItemCount: state => state.cart.length,

}

**🧪 Parte 3: Testando e Validando**

**3.1 Criar botão para adicionar ao carrinho**

No componente ShoppingCart.vue, adicione:

vue

CopiarEditar

<template>

<div>

<button @click="addProductToCart(product)">Adicionar ao carrinho</button>

<p>Total de itens no carrinho: {{ cartItemCount }}</p>

</div>

</template>

<script>

export default {

data() {

return {

product: { id: 1, name: 'Produto Exemplo', price: 100 }

}

},

computed: {

cartItemCount() {

return this.$store.getters.cartItemCount;

}

},

methods: {

addProductToCart(product) {

this.$store.dispatch('addToCart', product);

}

}

}

</script>

**3.2 Validando**

* ✅ Ao clicar no botão, o produto deve ser adicionado ao carrinho.
* ✅ O valor exibido de cartItemCount deve aumentar.
* ✅ As funções actions, mutations e getters devem estar conectadas corretamente.