LABORATÓRIO ADICIONAL II

Gerenciando dados

Nesse passo, vamos criar um carrinho de compras com as seguintes fases:

- Atualizar a visualização dos detalhes dos produtos para incluir um botão de Comprar, o qual vai adicionar o produto atual para uma lista de produtos que o serviço de carrinho vai gerenciar
- Adicionar o componente de carrinho, o qual vai mostrar os itens que estão no carrinho
- Adicionar um componente de entrega, que recupera os preços de entrega para os itens no carrinho usando o componente HttpClient para recuperar dados de entrega de um arquivo json

Criando um serviço de carrinho de compras

No Angular, um serviço é uma instância de classe que você pode disponibilizar para qualquer parte da aplicação usando o sistema de injeção de dependência do Angular.

Atualmente, usuários podem visualizar informações de produtos, e a aplicação consegue simular compartilhamento e notificações sobre mudanças produtos.

Vamos agora adicionar produtos no carrinho

Definindo um serviço de carrinho

1. Gere um serviço de carrinho utilizando o seguinte comando:

```
$ ng generate service carrinho
```

2. Importe a interface Produto no novo serviço. Defina também, dentro da classe, uma propriedade itens para guardar o array de produtos no carrinho.

```
$ export class CarrinhoService{
    itens: Produto[] = [];
  /* . . . */
  }
```

3. Defina métodos para adicionar itens no carrinho, retornar itens no carinho e limpar itens no carrinho.

```
$ adicionarCarrinho(produto: Produto) {
    this.itens.push(produto);
   }

   pegarItens() {
```

```
    return this.itens;
  }

  limparCarrinho() {
    this.itens= [];
    return this.itens;
  }
```

- Método adicionarCarrinho adiciona um produto no array
- Método pegarItens coleta os itens adicionados no carrinho e retorna cada item com a sua quantidade associada
- Métoda limparCarrinho retorna um array vazio de itens, que esvazia o carrinho

Usando o serviço de carrinho

1. Importe o serviço no Component de Detalhes dos Produtos
2. Injete esse serviço adicionando no constructor()

```
$       private carrinhoService: CarrinhoService
```

3. Defina o método de adicionar ao Carrinho, que vai adicionar o produto atual no carrinho

```
$       adicionarCarrinho(produto: Produto) {
            this.carrinhoService. adicionarCarrinho (produto);
            window.alert('Seu produto foi adicionado ao carrinho!');
        }
```

O método adicionarCarrinho() faz:

- Pega o produto atual como um argumento
- Usa o serviço de carrinho para adicionar o produto
- Mostra uma mensagem que você adicionou um produto no carrinho

4. No HTML, adicione um botão de Compra, e vincule o evento de click() para o método de adicionarCarrinho(). Esse código atualiza os detalhes dos produtos com um botão.

```
$       <button type="button" (click)=" adicionarCarrinho (produto)">Comprar</button>
```

Criando a visualização do carrinho

Crie o Componente do Carrinho

1. Gere um novo componente do Carrinho no terminal

2. Note que o novo Componente é adicionado nas declarações do app.module
3. Defina a nova rota de carrinho:

```
$       { path: 'carrinho', component: CarrinhoComponent },
```

4. Atualize o botão de Checkout na Barra Superior. Adicione um routerLink apontando para /carrinho

```
$       routerLink="/carrinho"
```

Mostrando os itens do carrinho

1. Importe o serviço do Carrinho no componente do Carrinho
2. Injete o serviço do Carrinho no constructor
3. Define uma propriedade itens para guardar os produtos no carrinho

```
$       itens= this.carrinhoService.pegarItens();
```

4. Atualize o HTML do Componente com um título e use uma com um *ngFor para mostrar cada item do carrinho com o nome e o preço

```
$       <h3>Cart</h3>


            <div class="cart-item" *ngFor="let item of items">
              <span>{{ item.name }}</span>
              <span>{{ item.price | currency }}</span>
            </div>
```

Recuperando preços de entrega

Servidores frequentemente retornam dados no formato de fluxo de dados. Fluxos de dados são úteis porque eles facilitam a transformação dos dados retornados e fazer modificações na maneira como você requisita os seus dados.

HttpClient é uma forma embutida para recuperar dados de uma API externa e providenciar eles para sua aplicação em formato de fluxo de dados

Inicialmente, vamos adicionar um json na pasta assests, com o nome entrega.json

```
$       [
            {
              "tipo": "Overnight",
              "preco": 25.99
            },
```

```
  {
    " tipo ": "2-Day",
    "preco": 9.99
  },
  {
    " tipo ": "Postal",
    "preco": 2.99
  }
]
```

Configure o AppModule para usar o HttpClient

HttpClienteModule registra os provedores que a sua aplicação precisa para usar o serviço HttpCliente por toda a aplicação

1. Importe HttpClientModule no AppModule:

```
$ import { HttpClientModule } from '@angular/common/http';
```

2. Adicione esse module nos imports do @NgModule

Configure o serviço de Carrinho para utilizar o HttpClient

O próximo passo vai ser injetar o serviço de HttpClient no seu serviço de carrinho para que você consiga pegar os dados e interagir com APIs externas e recursos

1. No serviço do carrinho, importe HttpClient:

```
$ import { HttpClient } from '@angular/common/http';
    import { Produto} from './produtos;
    import { Injectable } from '@angular/core';
```

2. Injete o HttpClient no constructor()

```
$ @Injectable({
    providedIn: 'root'
  })
  export class CarrinhoService {
    itens: Produto[] = [];
```

```
  constructor(
    private http: HttpClient
  ) {}
/* . . . */
}
```

Configure o serviço do Carrinho para pegar os preços de entrega

Para pegar os dados de entrega, você pode usar o método get() do HttpClient

1. No serviço do carrinho, adicione um novo método pegarPrecosEntrega():

```
$       pegarPrecosEntrega () {
            return this.http.get<{tipo: string, preco: number}[]>('/assets/entrega.json');
        }
```

Criando um componente de entrega

1. Crie um novo componente do carrinho chamado entrega no terminal
2. No Router, adicione uma rota para entrega.

```
$       { path: 'entrega', component: EntregaComponent },
```

Não há um link específico para esse Component ainda. Porém nós podemos verificar o sucesso da rota direto no navegador

Configurando o componente de entrega para usar o serviço do carrinho

1. No componente de entrega, importe o serviço do carrinho
2. Injete o serviço no contructor do Componente.
3. Defina a propriedade custosEntrega e sete essa variável utilizando o pegarPrecosEntrega do serviço de carrinho

```
$       custosEntrega!: Observable<{ tipo: string, preco: number } []>;
ngOnInit(): void {
            This.custosEntrega = this.carrinhoService.pegarPrecosEntrega();
            }
```

4. Atualize o template para mostrar o tipo de entrega e os preços usando a pipe async

```
$       <h3>Preços de Entrega</h3>
```

```
<div class="shipping-item" *ngFor="let entrega of custosEntrega | async">
  <span>{{ entrega.tipo}}</span>
  <span>{{ entrega.preco| currency }}</span>
</div>
```

A pipe async retorna o último valor de um fluxo de dados e continua a fazer pelo ciclo de vida de um dado componente. Quando o Angular destroi o componente, a pipe async automaticamente para.

5. Adicione um link no Componente do carrinho para visualizar o Componente de entrega

$
```
<p>
  <a routerLink="/shipping">Shipping Prices</a>
</p>
```

Fim do laboratório adicional 2