Primer 2: Slanje parametara serveru

Ovaj primer se nastavlja na primer 1.

1. U klasi *PizzaListComponent* dodat je objekat *params* koji sadrži polja koja odgovaraju parametrima koje ćemo da šaljemo serveru sa zahtevom. Za sada, imamo samo dva parametra *sort* i *sortDirection* koji nam trebaju za sortiranje:

```
export class PizzaListComponent implements OnInit {
    pizzaList: Pizza[] = [];
    pizzaCount: number = -1;

    params: any = {
        "sort": "name",
        "sortDirection": "asc"
    }
```

2. U templejt PizzaListComponent:
    a. Dodat select sa tri opcije sortiranja: *name*, *grade* i *price*. Primetite da *value* atributi u <option> elementima imaju vrednosti koje odgovaraju poljima pizza objekata koje vraća server. Želimo da putem ovog selecta korisnik može da odabere vrednost *sort* parametra koji će poslati serveru. Na primer, ako odabere *Name*, serveru ćemo da pošaljemo *sort=name*. Putem *ngModel* direktive, kažemo da će ono što korisnik odabere na ovom selektu biti sačuvano u *params.sort*
    b. Slično, dodat je select sa dve opcije koje regulišu smer sortiranja: *Ascending* i *Descending*. Putem ngModel kažemo da će se ono što korisnik odabere u ovom selectu čuvati u *params.sortDirection*. Server može da primi *sortDirection=desc* – ovo mu je znak da treba da sortira u opadajućem redosledu ili *sortDirection=asc* – server će onda sortirati u rastućem redosledu (poslali smo mu nešto što nije *desc*)

```
<div>{{params | json}}</div>

<div>
    <select [(ngModel)]="params.sort">
        <option value="name">Name</option>
        <option value="grade">Grade</option>
        <option value="price">Price</option>
    </select>

    <select [(ngModel)]="params.sortDirection">
        <option value="asc">Ascending</option>
        <option value="desc">Descending</option>
    </select>
</div>

<app-pizza-details *ngFor="let pizza of pizzaList" [pizza]="pizza"></app-pizza-details>
```

3. U klasi *PizzaListComponent* smo kod koji dobavlja listu pica izdvojili u posebnu funkciju *refreshList:*

```
export class PizzaListComponent implements OnInit {
    pizzaList: Pizza[] = [];
    pizzaCount: number = -1;

    params: any = {
        "sort": "name",
        "sortDirection": "asc"
    }

    constructor(private pizzaService :PizzaService) { }

    ngOnInit() {
        this.refreshList();
    }

    refreshList(): void {
        this.pizzaService.getAll(this.params).subscribe({
            next: data => {
                this.pizzaList = data.pizzas;
                this.pizzaCount = data.count;
                console.log("Retreived pizzas: ", this.pizzaCount);
            }
        });
    }

}
```

Razlog je što ćemo svaki put kada želimo drugačije sortiranje liste, morati da uputimo novi zahtev serveru, pa ne želimo da ovaj kod bude dupliran – dovoljno je da pozovemo ovu funkciju (u narednim koracima ćemo omogućiti da ona šalje parametre).

4. U templejtu *PizzaListComponent* ćemo namestiti da se ova funkcija (korak 3) poziva svaki put kada korisnik odabere novu opciju u jednom od select elemenata:

```
<div>{{params | json}}</div>

<div>
    <select [(ngModel)]="params.sort" (change)="refreshList()">
        <option value="name">Name</option>
        <option value="grade">Grade</option>
        <option value="price">Price</option>
    </select>

    <select [(ngModel)]="params.sortDirection" (change)="refreshList()">
        <option value="asc">Ascending</option>
        <option value="desc">Descending</option>
    </select>
</div>

<app-pizza-details *ngFor="let pizza of pizzaList" [pizza]="pizza"></app-pizza-details>
```

Testirajte svoju aplikaciju: s obzirom na to da se svaki put kada se pozove funkcija *refreshList*() u konzoli ispiše koliko je pica dobavljeno sa servera, svaki put kada promenite vrednost u jednom od select elemenata, trebao bi da se u konzoli ispiše broj dobavljenih pica.

5. Ispravićemo funkciju *refershList*() tako da se u *getAll* metodu *PizzaService*a šalju podešeni parametri (*getAll* metodu ćemo morati u narednim koracima popraviti da ove parametre šalje serveru).

```
refreshList(): void {
    this.pizzaService.getAll(this.params).subscribe({
        next: data => {
            this.pizzaList = data.pizzas;
            this.pizzaCount = data.count;
            console.log("Retreived pizzas: ", this.pizzaCount);
        }
    });
}
```

6. U servisu *PizzaService* ćemo ispraviti *getAll* metodu tako da bude u stanju da primi od komponente parametre koje treba da pošalje serveru i pošalje ih sa zahtevom. Namestili smo da parametre od komponente očekujemo u vidu običnog *JavaScript* objekta, kao i da je ovaj parametar funkcije *getAll* opcion (komponenta ne mora da nam pošalje parametre, u tom slučaju joj vraćamo sve pice sa servera, bez ikakvih podešavanja).

```
import { Injectable } from '@angular/core';
import { HttpClient, HttpParams } from '@angular/common/http';
import { Observable } from 'rxjs';
import { map } from 'rxjs/operators';

import { Pizza } from '../model/pizza.model';
import { PizzaSearchResult } from '../model/pizzaSearchResult.model';

const baseUrl = "http://localhost:3000/api/pizzas";

@Injectable({
    providedIn: 'root'
})
export class PizzaService {

    constructor(private http :HttpClient) { }

    getAll(params? :any) :Observable<PizzaSearchResult>{
        let queryParams = {};
        if(params){
            queryParams = {params : new HttpParams()
                .set("pageSize", params.pageSize && params.pageSize.toString() || "")
                .set("page", params.page && params.page.toString() || "")
                .set("filter", params.filter && JSON.stringify(params.filter) || "")
                .set("sort", params.sort && params.sort.toString() || "")
                .set("sortDirection", params.sortDirection && params.sortDirection.toString() || "")
            }
        }

        return this.http.get(baseUrl, queryParams).pipe(map(
            data => { return new PizzaSearchResult(data) }
        ));
    }
}
```

Na osnovu parametara koje nam je poslala komponenta, pravimo *JavaScript* objekat *queryParams* koji ima polje *params* tipa *HttpParams* (definisan u *@angular/common/http*). U objektu *params* smo napravili sve parametre koje server može da primi (znamo koji su na osnovu datog API-ja servera).

Isprobajte svoju aplikaciju.