Pagination - Keyword Search

Pagination

© luv2code LLC

Pagination

© luv2code LLC

Development Process

Step-By-Step

© luv2code LLC

Development Process

Step-By-Step

1. Add pagination support to ProductService

© luv2code LLC

Development Process

Step-By-Step

1. Add pagination support to ProductService
2. Update ProductListComponent to handle pagination

© luv2code LLC

Step1: Add pagination support to ProductService

© luv2code LLC

Step1: Add pagination support to ProductService

File: product.service.ts

```
searchProductsPaginate(thePage: number,
                       thePageSize: number,
                       theKeyword: string): Observable<GetResponseProducts> {

  const searchUrl = `${this.baseUrl}/search/findByNameContaining`
    + `?name=${theKeyword}&page=${thePage}&size=${thePageSize}`;

  return this.httpClient.get<GetResponseProducts>(searchUrl);
}
```


© luv2code LLC

Step1: Add pagination support to ProductService

File: product.service.ts

Pass in parameters for pagination

```
searchProductsPaginate(thePage: number,
                       thePageSize: number,
                       theKeyword: string): Observable<GetResponseProducts> {

  const searchUrl = `${this.baseUrl}/search/findByNameContaining`
    + `?name=${theKeyword}&page=${thePage}&size=${thePageSize}`;

  return this.httpClient.get<GetResponseProducts>(searchUrl);
}
```


© luv2code LLC

Step1: Add pagination support to ProductService

File: product.service.ts

Pass in parameters for pagination

```
searchProductsPaginate(thePage: number,
                       thePageSize: number,
                       theKeyword: string): Observable<GetResponseProducts> {

  const searchUrl = `${this.baseUrl}/search/findByNameContaining`
    + `?name=${theKeyword}&page=${thePage}&size=${thePageSize}`;

  return this.httpClient.get<GetResponseProducts>(searchUrl);
}
```

Spring Data REST supports pagination out of the box.
Just send the parameters for page and size

© luv2code LLC

Step 2: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  ...

  handleSearchProducts() {

    const theKeyword: string = this.route.snapshot.paramMap.get('keyword');
    ...
    // now get the products for the given category id
    this.productService.searchProductsPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               theKeyword).subscribe(this.processResult());
  }
```


© luv2code LLC

Step 2: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  ...

  handleSearchProducts() {

    const theKeyword: string = this.route.snapshot.paramMap.get('keyword');
    ...
    // now get the products for the given category id
    this.productService.searchProductsPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               theKeyword).subscribe(this.processResult());
  }
```

Call service and pass in parameters

© luv2code LLC

Step 2: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  ...

  handleSearchProducts() {

    const theKeyword: string = this.route.snapshot.paramMap.get('keyword');
    ...
    // now get the products for the given category id
    this.productService.searchProductsPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               theKeyword).subscribe(this.processResult());
  }
```

Call service and pass in parameters

© luv2code LLC

Step 2: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  ...

  handleSearchProducts() {

    const theKeyword: string = this.route.snapshot.paramMap.get('keyword');
    ...
    // now get the products for the given category id
    this.productService.searchProductsPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               theKeyword).subscribe(this.processResult());

  }
```

Call service and pass in parameters

Pagination component: pages are 1 based

Spring Data REST: pages are 0 based

© luv2code LLC

Step 2: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  ...

  handleSearchProducts() {

    const theKeyword: string = this.route.snapshot.paramMap.get('keyword');
    ...
    // now get the products for the given category id
    this.productService.searchProductsPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               theKeyword).subscribe(this.processResult());
  }

  private processResult() {
    return data => {
      this.products = data._embedded.products;
      this.thePageNumber = data.page.number + 1;
      this.thePageSize = data.page.size;
      this.theTotalElements = data.page.totalElements;
    };
  }

}
```

Same method
from previous videos

© luv2code LLC