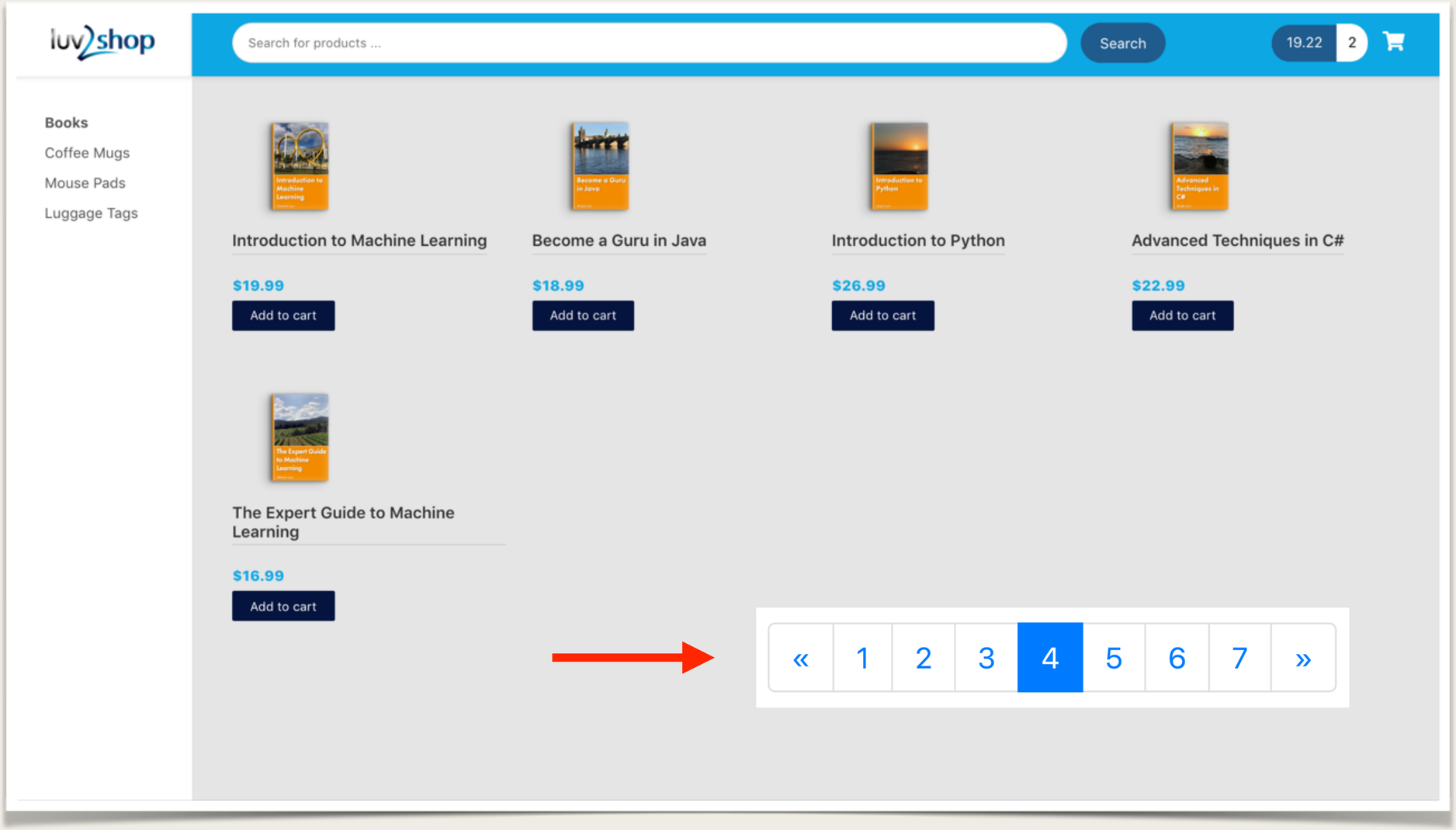
Pagination - Development Process

Pagination



Development Process

1. Install ng-bootstrap
2. Refactor the interface for: GetResponseProducts
3. Add pagination support to ProductService
4. Update ProductListComponent to handle pagination
5. Enhance HTML template to use ng-bootstrap pagination component

Step 1: Install ng-bootstrap

- Run the following commands in your Angular project directory

```
> ng add @angular/localize

> npm install @ng-bootstrap/ng-bootstrap
```

Dependency for Angular 9+

Installs ng-bootstrap

https://ng-bootstrap.github.io/#/getting-started

Step 1: Install ng-bootstrap

- Import the module for ng-bootstrap

File: app.module.ts

```
import { NgbModule } from '@ng-bootstrap/ng-bootstrap';

@NgModule({
  ...
 imports: [
    RouterModule.forRoot(routes),
    BrowserModule,
    HttpClientModule,
    NgbModule
  ],
  ...
})
export class AppModule { }
```

ng-bootstrap module

Step 2: Refactor the interface for GetResponseProducts

- We currently use the interface `GetResponseProducts`
- Maps JSON data from REST API to our TypeScript objects

JSON

```
{
  "_embedded" : {
    "products" : [ {
      "id" : 1,
      "sku" : "BOOK-TECH-1000",
      "name" : "Crash Course in Python",
    ...
    }, {
      "id" : 2,
      "sku" : "BOOK-TECH-1001",
      "name" : "Become a Guru in JavaScript",
    ...
    } ]
  }
}
```

```
interface GetResponseProducts {
  _embedded: {
    products: Product[];
  }
}
```

Spring Data REST - Response Meta Data

- The response meta data also has valuable information for pagination

```
<< array of products >>
...
"page" : {
  "size" : 20,
  "totalElements" : 100,
  "totalPages" : 5,
  "number" : 0
}
```

Size of this page

Grand total of ALL elements in the database.
But we are NOT returning all of the elements.
Just the "count" for informational purposes only.

Total pages available

Current page number

Step 2: Refactor the interface for GetResponseProducts

- Refactor the interface to support the pagination meta-data

JSON

```
{
  "_embedded" : {
  ...
 },
  "page" : {
    "size" : 20,
    "totalElements" : 100,
    "totalPages" : 5,
    "number" : 0
  }

}
```

```
interface GetResponseProducts {
  _embedded: {
    products: Product[];
  },
  page: {
    size: number,
    totalElements: number,
    totalPages: number,
    number: number
  }
}
```

Step 3: Add pagination support to ProductService

File: product.service.ts

Pass in parameters for pagination

```
getProductListPaginate(thePage: number,
                       thePageSize: number,
                       theCategoryId: number): Observable<GetResponseProducts> {

  const url = `${this.baseUrl}/search/findByCategoryId`
    + `?id=${theCategoryId}&page=${thePage}&size=${thePageSize}`;

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

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

Step 4: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  // new properties for pagination
  thePageNumber: number = 1;
  thePageSize: number = 10;
  theTotalElements: number = 0;
  ...

  handleListProducts() {
    ...
    // now get the products for the given category id
    this.productService.getProductListPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               this.currentCategoryId).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;
    };
  }

}
```

When data arrives from product service ... then set properties based on the data

Everything on right-hand side of assignment is data from Spring Data REST JSON

Step 4: Update ProductListComponent to handle pagination

File: product-list.component.ts

```
export class ProductListComponent implements OnInit {

  // new properties for pagination
  thePageNumber: number = 1;
  thePageSize: number = 10;
  theTotalElements: number = 0;
  ...

  handleListProducts() {
    ...
    // now get the products for the given category id
    this.productService.getProductListPaginate(this.thePageNumber - 1,
                                               this.thePageSize,
                                               this.currentCategoryId).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;
    };
  }

}
```

Pagination component: pages are 1 based

Spring Data REST: pages are 0 based