Lab 04 – RouterLink and Services

Shopping Cart

ในแลปนี้เราจะกลับไปยังแลปแรกสุดที่เราได้สร้างไว้ ซึ่งเราทำไว้แค่แสดงสินค้าพร้อมรายละเอียด แอพนี้ไม่มี State หรือ Navigation เนื่องจากมีแค่หน้า My Store ในแลปนี้เราจะสร้างเพจเฉพาะของรายละเอียดสินค้า รวมถึงการใช้ Cart Service ด้วย

1. เปิดแอพที่สร้างไว้ตอนแลปแรก จากนั้นกด Fork แล้วตั้งชื่อ Repository เป็น *lab04
2. สร้าง Component ใหม่โดยใช้ชื่อ product-details
3. ใน app.module.ts เพิ่ม route ให้กับ ProductDetailsComponent ดังนี้ ความหมายคือถ้าเจอ URL ในรูปแบบที่กำหนดไว้จะเรียกการใช้งานจาก Component ที่ระบุไว้

```
@NgModule({
  imports: [
    BrowserModule,
    ReactiveFormsModule,
    RouterModule.forRoot([
      { path: '', component: ProductListComponent },
      { path: 'products/:productId', component:
      ProductDetailsComponent}
    ])
  ],
```

4. สร้าง Link ด้วย RouterLink เพื่อให้เวลาเราคลิกที่ชื่อสินค้าแล้วเปิดหน้ารายละเอียด
    a. อัพเดต *ngFor ด้วยการเพิ่มค่า Index ให้กับ ProductId
    b. กำหนด [routerLink] ให้เปลี่ยนหน้า URL ไปไหนรูปแบบเดียวกับที่กำหนดไว้ใน Route ของ app.module.ts
    c. ทดสอบการทำงานด้วยการคลิกที่ชื่อสินค้า ถ้ามีคำว่า product-details works! ก็แสดงว่า routerLink ทำงานอย่างถูกต้อง

```
<h2>Products</h2>

<div *ngFor="let product of products; index as productId">

  <h3>
    <a [title]="product.name + ' details'" [routerLink]="['/products', productId]">
      {{ product.name }}
    </a>
  </h3>

</div>
```

5. แก้ไข Product Details Component ให้นำค่า ProductId มาใช้แสดงผล
    a. Import ข้อมูล Product จาก product.ts ในลักษณะเดียวกันกับ Product List Component พร้อมทั้งเพิ่ม Property product เอาไว้รับค่า
    b. Import ActivatedRoute จาก @angular/router จากนั้นประกาศ Property route แบบคลาส ActivatedRoute ใน constructor

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';

import { products } from '../products';

@Component({
  selector: 'app-product-details',
  templateUrl: './product-details.component.html',
  styleUrls: ['./product-details.component.css']
})
export class ProductDetailsComponent implements OnInit {
  product;

  constructor(
    private route: ActivatedRoute,
  ) { }
```

6. เพิ่มฟังก์ชัน ngOnInit เพื่อรับค่า ProductId มาจาก routerLink ใน Product List Component ซึ่งฟังก์ชันนี้จะถูกเรียกโดย Angular หลังจากสร้าง Component เสร็จสิ้น

```
18    ngOnInit() {
19      this.route.paramMap.subscribe(params => {
20        this.product = products[+params.get('productId')];
21      });
22    }
```

7. สร้าง Template แสดงรายละเอียดสินค้า โดยใช้ *ngIf ตรวจสอบว่า Property product มีค่าอยู่หรือไม่

```
1   <h2>Product Details</h2>
2
3   <div *ngIf="product">
4     <h3>{{ product.name }}</h3>
5     <h4>{{ product.price | currency:'THB' }}</h4>
6     <p>{{ product.description }}</p>
7
8   </div>
```

Service ใน Angular เป็นคลาสที่สามารถใช้ได้ในทุก Component ด้วยวิธี Dependency Injection System โดย Service เป็นแหล่งที่แลกเปลี่ยนข้อมูลภายในแอพ ในที่นี้คือระบบ Cart ที่ต้องรับข้อมูลการซื้อจาก Product Page

8. สร้าง Cart Service โดยการคลิกขวาที่ Folder app แล้วเลือก Angular Generator ต่อด้วย Service ตั้งชื่อว่า cart เราจะได้ไฟล์ cart.service.ts
    a. เพิ่ม { providedIn: 'root' } ใน @Injectable เพื่อให้สามารถใช้ได้ทั้งแอพ

```
1   import { Injectable } from '@angular/core';
2
3   @Injectable({
4     providedIn: 'root'
5   })
6   export class CartService {
7
8     constructor() { }
9
10  }
```

9. ในคลาส CartService สร้าง Array ชื่อ items จากนั้นสร้างฟังก์ชัน addToCart, getItems และ clearCart

```
12    items = [];
13
14    addToCart(product) {
15      this.items.push(product);
16    }
17
18    getItems() {
19      return this.items;
20    }
21
22    clearCart() {
23      this.items = [];
24      return this.items;
25    }
```

10. เราต้องการใช้งาน Cart Service ในหน้า Product Detail
    a. อยากแรกคือการ Import Service นี้ใน product-details.component.ts
    b. จากนั้น Inject CartService ไว้ใน constructor ในรูปแบบเดียวกันกับการ Inject ActivatedRoute
11. สร้างฟังก์ชัน addToCart โดยให้แสดงข้อความว่าสินค้าถูกโยนลง Cart พร้อมทั้งเรียกใช้ฟังก์ชันชื่อเดียวกันใน CartService

```
26    addToCart(product) {
27      window.alert('Your product has been added to the cart!');
28      this.cartService.addToCart(product);
29    }
```

12. เพิ่มปุ่ม Buy ในหน้า Product Detail แล้วผูกฟังก์ชัน addToCart ไว้กับ (click)

แบบฝึกหัด

ให้สร้างหน้า Cart เพื่อแสดงสินค้าที่ถูกซื้อมาแล้วทั้งหมด

1. สร้าง Component ที่ชื่อว่า Cart
2. เพิ่ม routing ไปยัง Cart Component
    a. ทดสอบการทำงานด้วยการกดที่ปุ่ม Checkout แล้วต้องเห็น cart works!
3. เพิ่มส่วนแสดงสินค้า
    a. Import Cart Service (ดูตัวอย่างจากข้อ 10a)
    b. Inject Cart Service (ดูตัวอย่างจากข้อ 10b)
    c. สร้าง Property items มาเก็บข้อมูลสินค้า
    d. ในฟังก์ชัน ngOnInit เรียกใช้ฟังก์ชัน getItems ของ Cart Service จากนั้นนำผลลัพธ์ที่ได้มาใส่ไว้ใน Property items (this.items = ???)
    e. ใช้ *ngFor เพื่อแสดงสินค้าจาก items ในไฟล์ HTML
4. เพิ่มปุ่ม Clear Cart เพื่อลบสินค้าทั้งหมดออกจาก Cart
    a. สร้างฟังก์ชัน clearCart ใน Cart Component แล้วเรียกใช้ฟังก์ชัน clearCart ของ Cart Service (อย่าลืม this.items = ??? เพื่อล้าง items ใน Component)
    b. ผูก clearCart ไว้กับ (click) ของปุ่ม
5. (Optional) แสดงราคารวมของสินค้าที่ซื้อทั้งหมดในหน้า Cart