INT204 Server Side II

1.api/products

ProductServices

```
public Page<Product> getProductsWithpaging(int page, int size, String sortBy) {
    Sort sort = Sort.by(sortBy);
    Pageable pageable = PageRequest.of(page, size, sort);
    return productRepository.findAll(pageable);
}
```

ProductController

```
@GetMapping("/pages")
public Page<Product> getProductsPage(
        @RequestParam(defaultValue = "0") Integer page,
        @RequestParam(defaultValue = "10") Integer size,
        @RequestParam(defaultValue = "productCode") String sortBy) {
    return service.getProductsWithpaging(page,size,sortBy);
}
```

ProductServices

เป็นการเอาข้อมูลของProductsโดยใช้ page กับการ sort และตัว method มีการเก็บค่า3ตัว คือ page ,size , sortBy และมี 2 object คือ sort กับ pageable จากนั้นเรียกใช้ method findAll เพื่อขอข้อมูลของProducts (page,size,sortBy)

ProductController

controller จะไปเอา object ของ service มาเก็บ และใช้ parameter 3 ตัว โดยใช้ requestparameter ที่เป็น page ,size ,sortBy

Method getProductPage จะส่งคืนเป็น page โดยใช้ parameter 3ตัว

และการเรียกใช้ method getProductsWithpaging ด้วย parameter 3ตัว และส่งคืนค่า object ตามที่กำหนด

2.api/products/10.0/250.0

ProductService

```
1 usage
public List<Product> getProductsByPriceSetween(Double low, Double high) {
    if (low > high) {
//          double tmp = low;
//          low = high;
//          high = tmp;
        return productRepository.findProductByPriceBetweenOrderByPriceDesc(high, low);
    } else {
        return productRepository.findProductByPriceBetweenOrderByPriceDesc(low, high);
    }
}
```

ProductController

```
@GetMapping("/price/{min}/{max}")
public List<Product> getProductFilter(@PathVariable Double min,@PathVariable Double max){
    return service.getProductsByPriceSetween(min,max);
}
```

ProductService

Method getProductByPriceSetween รับ parameter 2ตัว คือ high , low และทำการ check ถ้าหาก low > high จะทำการเรียกใช้ method findProductByPriceBetweenOrderByPriceDesc เดียวกันโดยมีค่า high เป็น parameter แรก และค่า low เป็นพารามิเตอร์ที่สอง ถ้า high > low จะเอาค่า low ก่อน และตามด้วย high

ProductController

ทำการรับ get request และ parameter 2 ตัวคือ min , max และนำไปใช้ใน method getProductByPriceSetween ซึ่ง method นี้ จะไปรับ min,max จากนั้นจะส่งไป services และเรียกใช้ getProductByPriceSetween ของตัว services เพื่อหา product ตามเงื่อนไขที่กำหนดไว้ และ return

3.api/products/ship

ProductService

```
public List<Product> getProductByProductLine(String productLine,int page,int size,String sortBy){
    Sort sort = Sort.by(sortBy);
    Pageable pageable = PageRequest.of(page, size, sort);

    return productRepository.findProductByProductLine(productLine,pageable);
}
```

ProductController

```
@GetMapping("/{productline}")
public List<Product> getProductByProductLine(
        @PathVariable String productline,
        @RequestParam(defaultValue = "0") Integer page,
        @RequestParam(defaultValue = "10") Integer size,
        @RequestParam(defaultValue = "productLine") String sortBy){
    return service.getProductByProductLine(productline,page,size,sortBy);
}
```

ProductService

เป็นการเอาข้อมูลของProductsโดยใช้ page กับการ sort และตัว method มีการเก็บค่า3ตัว คือ page ,size , sortBy และมี 2 object คือ sort กับ pageable จากนั้นเรียกใช้ method findProductByProductLine เพื่อรับข้อมูลของProducts (page,size,sortBy)

ProductController

controller จะไปเอา object ของ service มาเก็บ และใช้ parameter 3 ตัว โดยใช้ requestparameter ที่เป็น page ,size ,sortBy และ PathVariable

Method getProductByProductLine จะส่งคืนเป็น page โดยใช้ parameter 3ตัว

และการเรียกใช้ method getProductsByProductLine ด้วย parameter 3ตัว และส่งคืนค่า object ตามที่กำหนด รวมถึง productline ที่เป็น pathvariable

4.api/products/S10_1234

ProductServices

```
1 usage
public Product updateProduct(String productCode, Product updateProduct) {
    Product product = productRepository.findById(productCode).map(p -> mapProduct(p, updateProduct))
            .orElseThrow(() -> new RuntimeException(productCode + " does not exist !!"));
    return productRepository.saveAndFlush(product);
}
```

ProductController

```
@PutMapping("/{productCode}")
public Product updateProduct(@RequestBody Product updateProduct,
                             @PathVariable String productCode) {

    return service.updateProduct(productCode, updateProduct);
}
```

ProductServices

Method จะดึงข้อมูลด้วยการfindbyId และทำการmap จากนั้น method map จะส่งคืน ส่วน method orElseThow จะถูกเรียกใช้เพื่อส่ง RuntimeException หากมีข้อมูล แต่ถ้าหากไม่มีจะคืนค่าเป็นข้อความที่กำหนดไว้ และตัวที่อัพเดทจะถูกบันทึกไว้โดยใช้ method saveAndFlush

Productcontroller

จะทำการรับ request เป็นแบบ put และรับparameter 2ตัว คือ productCode,updateProduct ซึ่ง productCode จะเก็บเป็น path และถูกแปลงเป็น object ของ product และจะเรียก updateproduct ของตัว service เพื่ออัพเดทข้อมูล โดยจะส่ง parameter เข้าไปและทำการอัพเดทข้อมูล

5.api/products (POST)

ProductService

```
public Product addNewProduct(Product newProduct) {
    return productRepository.saveAndFlush(newProduct);
}
```

ProductController

```
@PostMapping("")
public Product create(@RequestBody Product newProduct){
    return service.addNewProduct(newProduct);
}
```

ProductService

จะใช้สำหรับเพิ่มnewproduct ซึ่งจะมีการรับข้อมูลเข้ามาในแบบ object ผ่าน parameter และเรียกใช้ method saveAndFlush มาบันทึกข้อมูลและreturn

ProductController

เป็น method สำหรับเพิ่มข้อมูลของสินค้า โดยจะรับ request เป็นแบบ post ซึ่งจะรับ parameter เป็น newProduct และจะเรียกใช้ medthod addNewProduct ของตัว service มาบันทึกข้อมูลและreturn