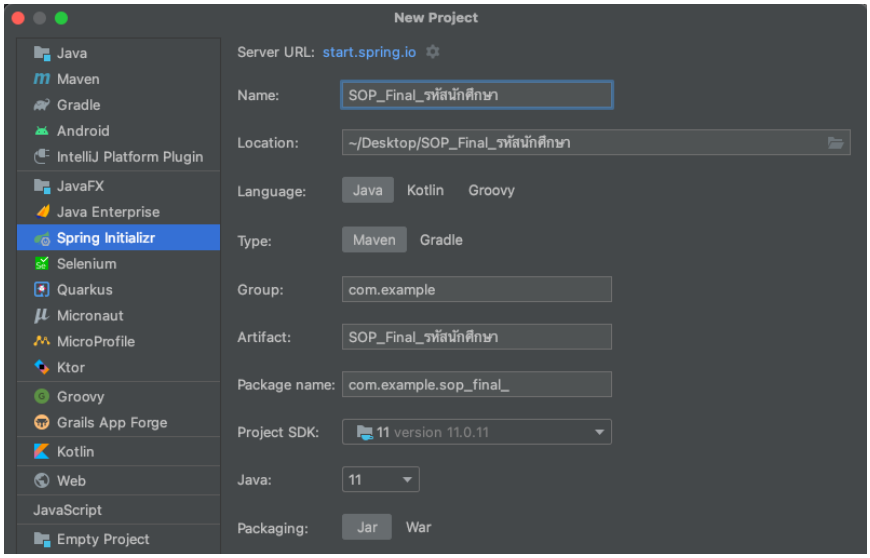
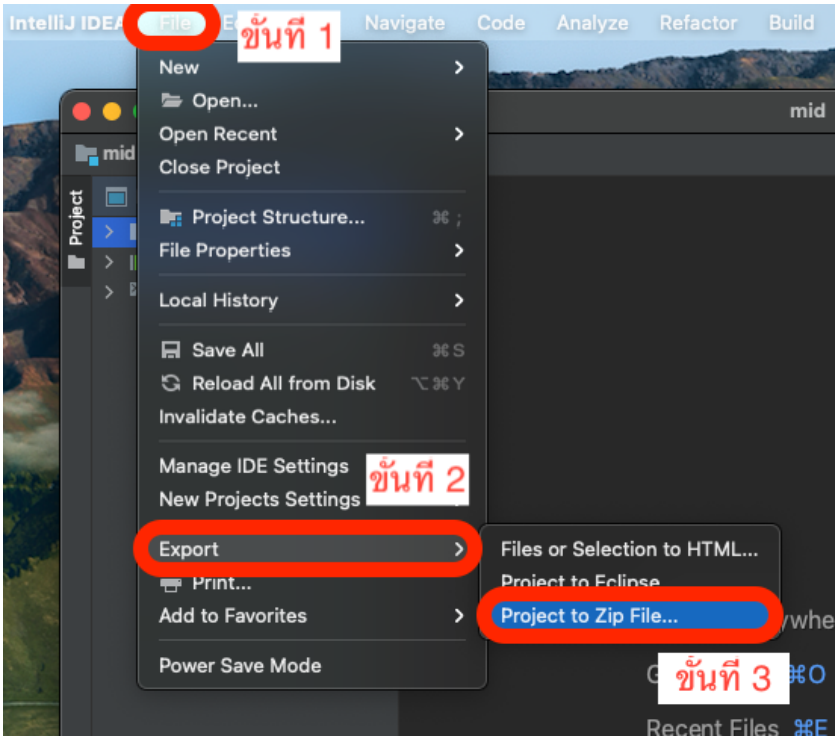
แบบฝึกหัด (เวลาทำ 120 นาที)

คำอธิบาย

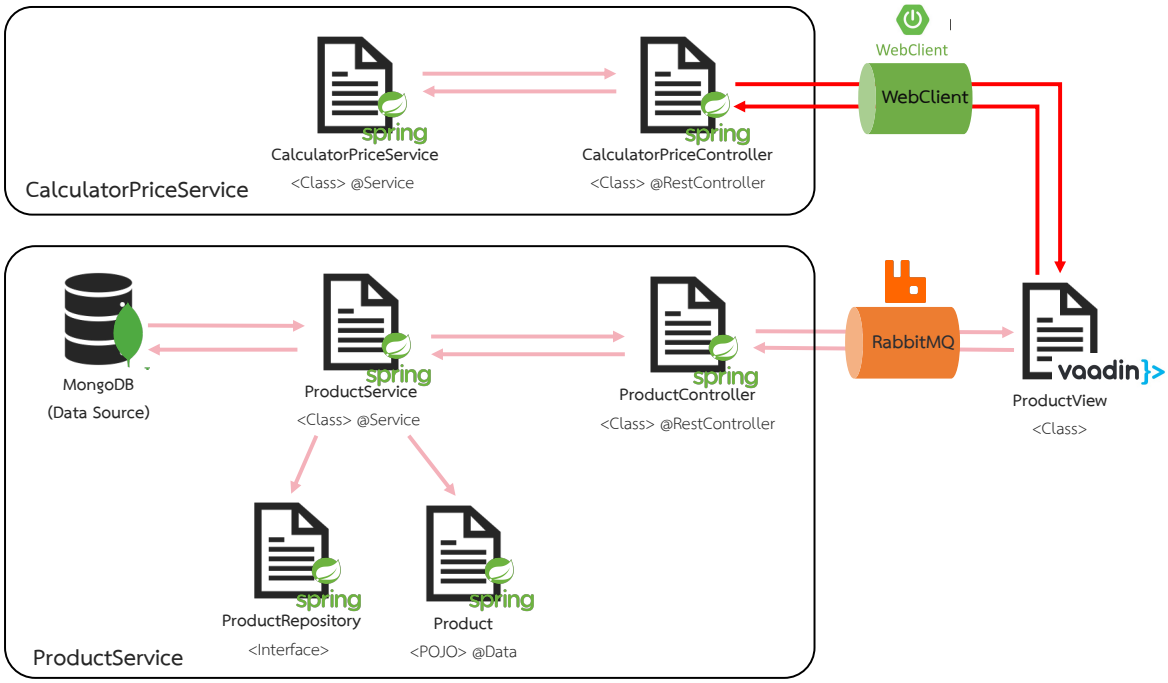
1. แบบฝึกหัดจำนวน 1 ข้อ
2. ให้นักศึกษาสร้างโปรเจคชื่อ ***SOP_Final_ตามด้วยรหัสนักศึกษา*** ด้วยโปรแกรม IntelliJ โดยมีรายละเอียดดังนี้



3. ให้นักศึกษาส่งไฟล์ทั้งโปรเจคในรูปแบบไฟล์ที่ได้รับการบีบอัดแล้ว *.7z, *.zip หรือ *.rar
(สามารถกดที่ File > Export > Project to Zip File ดังภาพ)



คำสั่ง ให้นักศึกษาสร้างโปรแกรมจัดการสินค้าซึ่งประกอบด้วย 7 ไฟล์ดังภาพ ได้แก่ CalculatorPriceService, CalculatorPriceController, ProductService, ProductController, ProductRepository, Product, และ ProductView



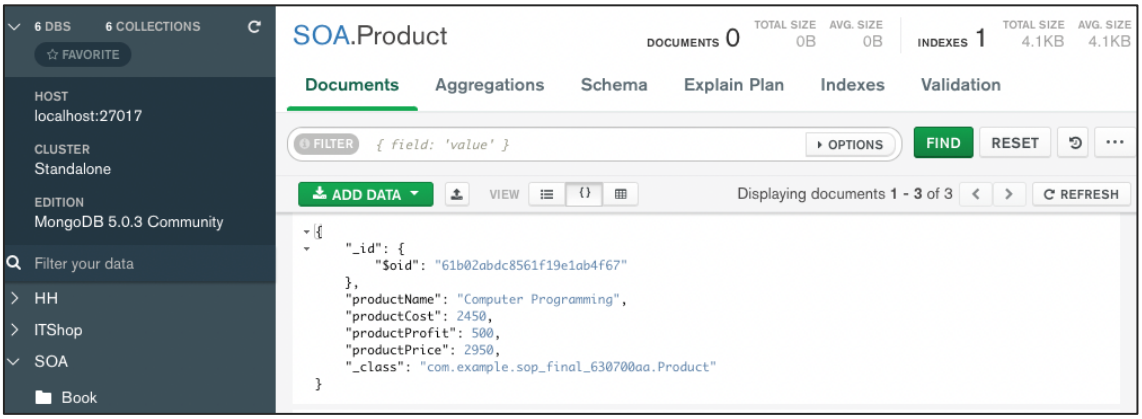
ภาพรวมของโปรแกรมจัดการสินค้า

โดยรายละเอียดแต่ละไฟล์จะปรากฏในตอนต่าง ๆ ดังนี้

- ตอนที่ 1 รายละเอียดการกำหนดค่าของ RabbitMQ และ MongoDB หน้าที่ 3.....(จำนวน 2 คะแนน)
- ตอนที่ 2 รายละเอียดของส่วนประสานผู้ใช้งาน หน้าที่ 5.....(จำนวน 6 คะแนน)
- ตอนที่ 3 รายละเอียดของ Calculator Price Service หน้าที่ 7.....(จำนวน 2 คะแนน)
- ตอนที่ 4 รายละเอียดของ Product Service หน้าที่ 8.....(จำนวน 5 คะแนน)

ตอนที่ 1 รายละเอียดการกำหนดค่าของ RabbitMQ และ MongoDB

ข้อย่อยที่ 1.1 ให้นักศึกษาสร้างฐานข้อมูลชื่อ SOA และ Collection ชื่อ Product ในโปรแกรม MongoDB เพื่อใช้เก็บข้อมูล Product จากนั้น ให้นักศึกษาบันทึกภาพหน้าจอดังตัวอย่างไปนี้ (ตั้งชื่อรูปว่า mongo.png)



โดยกำหนดให้ข้อมูลของ Product จะประกอบด้วย ชื่อสินค้า (Product Name) เป็นข้อความ, ราคาทุนของสินค้า (Product Cost) เป็นเลขทศนิยม, กำไรของสินค้า (Product Profit) เป็นเลขทศนิยม และราคาขาย (Product Price) เป็นเลขทศนิยม โดยตั้งชื่อของแต่ละแอททริบิวในฐานข้อมูลตามตารางต่อไปนี้

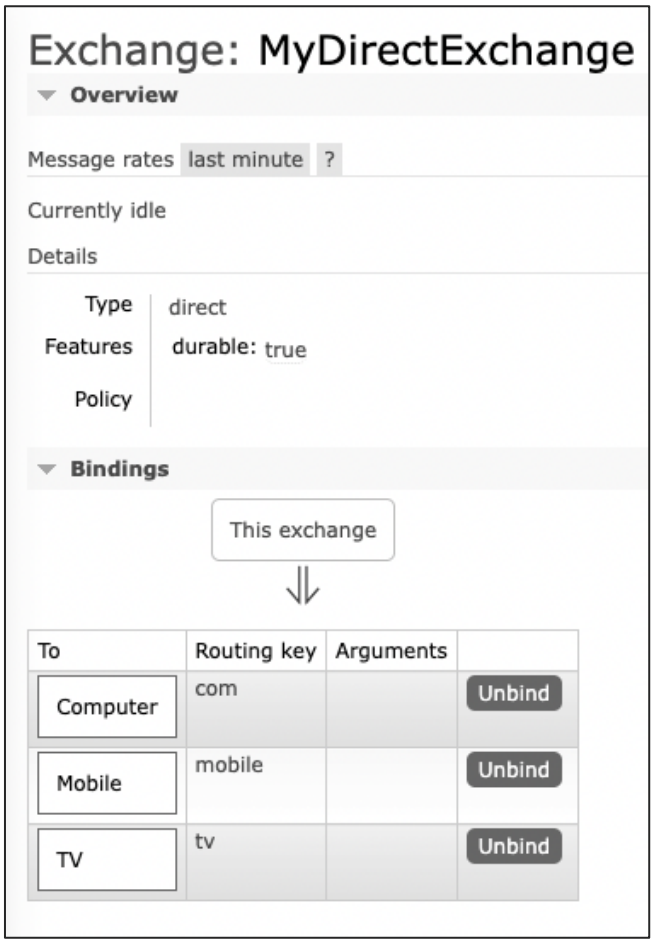
ชื่อแอททริบิว	ชื่อใน Collection ของ Product ของ MongoDB
ชื่อสินค้า (Product Name)	productName
ราคาทุนของสินค้า (Product Cost)	productCost
กำไรของสินค้า (Product Profit)	productProfit
ราคาขาย (Product Price)	productPrice

ข้อย่อยที่ 1.2 ให้นักศึกษาสร้าง Exchange และ Queue ใน RabbitMQ โดยมีรายละเอียดดังต่อไปนี้

- ให้นักศึกษาสร้าง Exchange แบบ Direct ที่มีชื่อว่า ProductExchange
- ให้นักศึกษาสร้างคิวชื่อ AddProductQueue, UpdateProductQueue, DeleteProductQueue, GetNameProductQueue และ GetAllProductQueue
- ให้นักศึกษาเพิ่มคิว AddProductQueue, UpdateProductQueue, DeleteProductQueue, GetNameProductQueue และ GetAllProductQueue ลงใน ProductExchange และกำหนดให้มี Routing Key ดังตารางต่อไปนี้

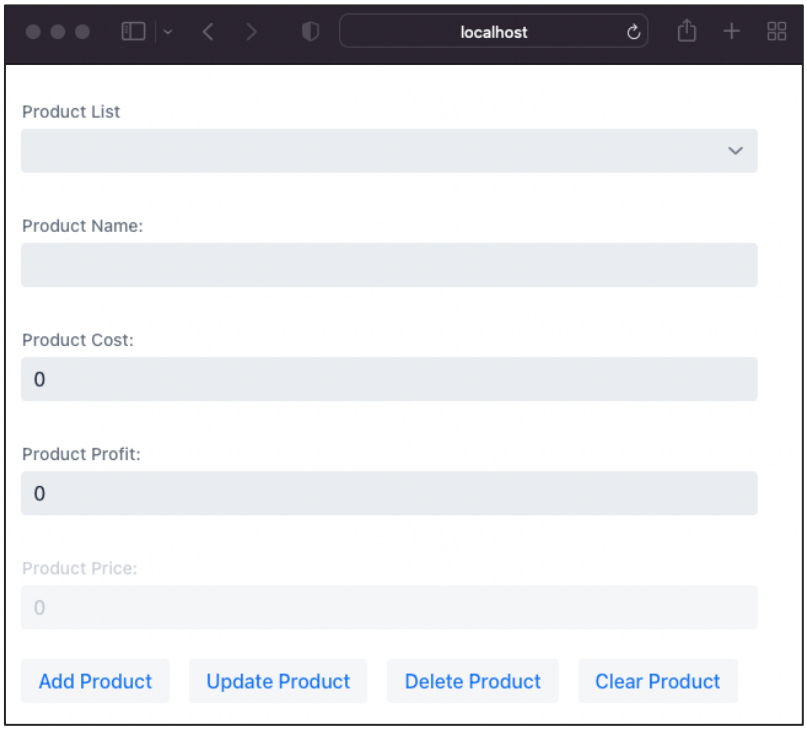
ชื่อคิว	Routing Key
AddProductQueue	add
UpdateProductQueue	update
DeleteProductQueue	delete
GetNameProductQueue	getname
GetAllProductQueue	getall

จากนั้น ให้นักศึกษาบันทึกภาพหน้าจอดังตัวอย่างไปนี้ (ตั้งชื่อรูปว่า rabbit.png)



ตอนที่ 2 รายละเอียดของส่วนประสานผู้ใช้งาน

ข้อย่อยที่ 1.3 ให้นักศึกษาสร้าง ProductView เป็นส่วนติดต่อผู้ใช้งานด้วย Vaadin ดังภาพ



หน้าต่าง ProductView

<u>การออกแบบส่วนติดต่อผู้ใช้งาน</u>

- ส่วนติดต่อผู้ใช้งานประกอบด้วย 9 Component ได้แก่ 1 x ComboBox, 1 x TextField, 3 x NumberField และ 4 x Button และมีการจัด Layout ดังภาพ โดยที่ นักศึกษาสามารถสร้าง Component หรือ Container เพิ่มเติมจากที่โจทย์กำหนดได้ตามสมควร
- กำหนดให้ความกว้างของ ComboBox, TextField และ NumberField เท่ากับ 600 px
- กำหนดให้ TextField มีค่าเริ่มต้นเป็น “” และ NumberField มีค่าเริ่มต้นเป็น 0.0
- กำหนดให้ NumberField ที่ใช้เก็บค่า Product Price จะไม่อนุญาตให้ผู้ใช้กำหนดหรือเปลี่ยนแปลงค่า
- กำหนดให้ ComboBox จะแสดง Product Name ที่เก็บไว้ในฐานข้อมูล

<u>การทำงานต่าง ๆ ของระบบ</u>

- NumberField ของ Product Price จะคำนวณและแสดงผลค่าใหม่เองโดยอัตโนมัติผ่านการเรียกใช้งานเมธอด serviceGetProducts() ของ Service ชื่อ CalculatorPrice ด้วยการสื่อสารผ่าน WebClient เมื่อเกิดเหตุการณ์ต่อไปนี้

- ผู้ใช้กด Enter ที่คีย์บอร์ดตรงบริเวณช่องราคาทุนของสินค้า (Product Cost) หรือกำไรของสินค้า (Product Profit)
- ผู้ใช้กดปุ่ม “Add Product” หรือ “Update Product” ในกรณีนี้โปรแกรมจะคำนวณค่า Product Price ใหม่ก่อนการเพิ่มหรือแก้ไขข้อมูลสินค้า

- เมื่อผู้ใช้กดปุ่ม “Add Product” โปรแกรมจะเพิ่มข้อมูลลงฐานข้อมูลผ่านเมธอด serviceAddProduct() ของ Service ชื่อ Product ด้วยการสื่อสารผ่าน RabbitMQ ***(ก่อนการเพิ่มข้อมูล โปรแกรมจะคำนวณค่า Product Price ใหม่ก่อน)*** ถ้าดำเนินการสำเร็จระบบจะแจ้งเตือนผ่าน Notification Component เป็นเวลา 0.5 วินาที โดยแสดงข้อความดังปรากฏในวีดิโอ
- เมื่อผู้ใช้กดปุ่ม “Update Product” โปรแกรมจะแก้ไขข้อมูลในฐานข้อมูลผ่านเมธอด serviceUpdateProduct() ของ Service ชื่อ Product ด้วยการสื่อสารผ่าน RabbitMQ ***(ก่อนการแก้ไขข้อมูลโปรแกรมจะคำนวณค่า Product Price ใหม่ก่อน)*** ถ้าดำเนินการสำเร็จระบบจะแจ้งเตือนผ่าน Notification Component เป็นเวลา 0.5 วินาที โดยแสดงข้อความดังปรากฏในวีดิโอ
- เมื่อผู้ใช้กดปุ่ม “Delete Product” โปรแกรมจะลบข้อมูลในฐานข้อมูลผ่านเมธอด serviceDeleteProduct() ของ Service ชื่อ Product ด้วยการสื่อสารผ่าน RabbitMQ
- เมื่อผู้ใช้กดปุ่ม “Clear Product” หรือเปิดโปรแกรมขึ้นมาใหม่ โปรแกรมจะล้างค่าเดิมและกำหนดค่าเริ่มต้นดังตารางต่อไปนี้

ชื่อแอททริบิว	ค่าเริ่มต้น
ชื่อสินค้า (Product Name)	“ ”
ราคาทุนของสินค้า (Product Cost)	0.0
กำไรของสินค้า (Product Profit)	0.0
ราคาขาย (Product Price)	0.0

ถ้าดำเนินการสำเร็จระบบจะแจ้งเตือนผ่าน Notification Component เป็นเวลา 0.5 วินาที โดยแสดงข้อความดังปรากฏในวีดิโอ

- เมื่อผู้ใช้กดเลือก ComboBox ที่ชื่อ Product List โปรแกรมจะแสดงชื่อสินค้าทั้งหมดในฐานข้อมูลผ่านเมธอด serviceGetAllProduct() ของ Service ชื่อ Product ด้วยการสื่อสารผ่าน RabbitMQ

คำใบ้ นักศึกษาอาจจะใช้ addFocusListener() เข้ามาช่วยเหลือ

นอกจากนี้ เมื่อผู้ใช้กดปุ่ม “Add Product” หรือ “Update Product” หรือ “Delete Product” ค่าใน ComboBox จะปรับปรุงให้เป็นปัจจุบัน

ตอนที่ 3 รายละเอียดของ Calculator Price Service

ข้อกำหนดสำหรับข้อย่อยที่ 1.4 และ 1.5

ข้อย่อยที่ 1.4 และ 1.5 เป็นคลาสของ Service ชื่อ CalculatorPrice ดังนั้น การเรียกใช้งานภายใน Service ระหว่าง CalculatorPriceService และ CalculatorPriceController ให้เรียกใช้งานแบบ “Inject object” หรือ “Dependency Injection” ด้วย @Autowired เท่านั้น

ข้อย่อยที่ 1.4 ให้นักศึกษาสร้าง CalculatorPriceService เป็น Service ที่มี 1 เมธอด คือ getPrice() ที่รับค่าเป็นราคาทุนของสินค้า (Product Cost) เป็นเลขทศนิยม และกำไรของสินค้า (Product Profit) เป็นเลขทศนิยม และเมธอด getPrice() จะคืนค่าเป็นราคาขาย (Product Price) เป็นเลขทศนิยม โดยคำนวณจากสมการต่อไปนี้

Product Price = Product Cost + Product Profit

ข้อย่อยที่ 1.5 ให้นักศึกษาสร้าง CalculatorPriceController แบบ REST API ที่มี 1 เมธอดโดยมีรายละเอียดดังนี้

- เมธอด serviceGetProducts() มีการรับค่าเป็นราคาทุนของสินค้า (Product Cost) เป็นเลขทศนิยม และกำไรของสินค้า (Product Profit) เป็นเลขทศนิยม
- เมธอด serviceGetProducts() จะคืนค่าเป็นราคาขาย (Product Price) เป็นเลขทศนิยมผ่านการเรียกใช้เมธอด getPrice() ในคลาส CalculatorPriceService
- เมธอด serviceGetProducts() มี URL ในการเรียกใช้งานเป็น `http://localhost:8080//getPrice/{cost}/{profit}` และเรียกผ่านตัวดำเนินการแบบ GET

ตอนที่ 4 รายละเอียดของ Product Service

ข้อกำหนดสำหรับข้อย่อยที่ 1.6 ถึง 1.9 เป็น Service ชื่อ Product

ข้อย่อยที่ 1.6 และ 1.9 เป็นคลาสและอินเตอร์เฟสของ Service ชื่อ Product ดังนั้น การเรียกใช้งานภายใน Service ระหว่าง Product, ProductRepository, ProductService และ ProductController ให้เรียกใช้งานตามหลักการ Repository Pattern เท่านั้น

ข้อย่อยที่ 1.6 ให้นักศึกษาสร้างคลาส Product เป็นคลาสแบบ POJO เพื่อใช้จัดเก็บข้อมูลของ Product ได้แก่ ชื่อสินค้า (Product Name) เป็นข้อความ, ราคาทุนของสินค้า (Product Cost) เป็นเลขทศนิยม, กำไรของสินค้า (Product Profit) เป็นเลขทศนิยม และราคาขาย (Product Price) เป็นเลขทศนิยม ***โดยแต่ละแอททริบิวให้ตั้งชื่อตามที่จะจัดเก็บใน MongoDB***

ข้อย่อยที่ 1.7 ให้นักศึกษาสร้างคลาส ProductRepository เป็นอินเตอร์เฟสสำหรับติดต่อกับฐานข้อมูล MongoDB โดยกำหนดให้นักศึกษา**ประกาศ**เมธอด findByName() ที่รับค่ามาเป็นชื่อสินค้า (Product Name) และคืนค่าเป็น Object ของคลาส Product ตัวที่มีชื่อสินค้าสอดคล้องกับที่กำหนดเข้ามา

ข้อย่อยที่ 1.8 ให้นักศึกษาสร้างคลาส ProductService เป็นคลาสแบบ Service ที่ใช้เป็นตัวกลางระหว่าง ProductRepository และ ProductController ในการแลกเปลี่ยนข้อมูลระหว่าง Service ชื่อ Product กับฐานข้อมูล ที่ประกอบไปด้วย 5 เมธอด ได้แก่

- ให้นักศึกษาสร้างเมธอด addProduct() ที่มีหน้าที่เพิ่มข้อมูลจาก Object ของคลาส Product ที่รับเข้ามาลงฐานข้อมูลและคืนค่า true ถ้าเพิ่มข้อมูลสำเร็จและคืนค่าเป็น false ถ้าเพิ่มข้อมูลไม่สำเร็จ
- ให้นักศึกษาสร้างเมธอด updateProduct() ที่มีหน้าที่แก้ไขข้อมูลจาก Object ของคลาส Product ที่รับเข้ามาลงในฐานข้อมูลที่และคืนค่า true ถ้าแก้ไขข้อมูลสำเร็จและคืนค่าเป็น false ถ้าแก้ไขข้อมูลไม่สำเร็จ
- ให้นักศึกษาสร้างเมธอด deleteProduct() ที่มีหน้าที่ลบข้อมูลจากฐานข้อมูลของตัวที่สอดคล้องกับ Object ของคลาส Product ที่รับเข้ามา และคืนค่า true ถ้าลบข้อมูลสำเร็จและคืนค่าเป็น false ถ้าลบข้อมูลไม่สำเร็จ
- ให้นักศึกษาสร้างเมธอด getAllProduct() ที่มีหน้าที่ดึงข้อมูลทุกรายการจากฐานข้อมูลและคืนค่าในรูปแบบ List ของ Product โดยไม่มีการรับค่า แต่ถ้าการทำงานไม่สำเร็จหรือกรณีอื่น ๆ จะคืนค่าเป็น null
- ให้นักศึกษาสร้างเมธอด getProductByName () ที่มีหน้าที่ดึงข้อมูลจากฐานข้อมูลเฉพาะรายการที่มีชื่อสินค้าตรงกับชื่อสินค้าที่รับเข้ามาและคืนค่าในรูปแบบ Object ของ Product แต่ถ้าการทำงานไม่สำเร็จหรือกรณีอื่น ๆ จะคืนค่าเป็น null

ข้อย่อยที่ 1.9 ให้นักศึกษาสร้างคลาส ProductController เป็น REST API ที่ประกอบไปด้วย 5 เมธอด ได้แก่

- ให้นักศึกษาสร้างเมธอด serviceAddProduct() ที่มีหน้าที่เพิ่มข้อมูลจาก Object ของคลาส Product ที่รับเข้ามาลงฐานข้อมูล และคืนค่า true ถ้าการเพิ่มข้อมูลสำเร็จและคืนค่าเป็น false ถ้าการเพิ่มข้อมูลไม่สำเร็จ ซึ่งการทำงานของเมธอด serviceAddProduct() ต้องเรียกใช้งานเมธอด addProduct() ของคลาส ProductService นอกจากนี้ เมธอด serviceAddProduct() มีการสื่อสารด้วย RabbitMQ และเชื่อมต่อกับคิวที่ชื่อ AddProductQueue
- ให้นักศึกษาสร้างเมธอด serviceUpdateProduct() ที่มีหน้าที่แก้ไขข้อมูลจาก Object ของคลาส Product ที่รับเข้ามาลงฐานข้อมูล และคืนค่า true ถ้าการแก้ไขข้อมูลสำเร็จและคืนค่าเป็น false ถ้าการแก้ไขข้อมูลไม่สำเร็จ ซึ่งการทำงานของเมธอด serviceUpdateProduct() ต้องเรียกใช้งานเมธอด updateProduct() ของคลาส ProductService นอกจากนี้ เมธอด serviceUpdateProduct() มีการสื่อสารด้วย RabbitMQ และเชื่อมต่อกับคิวชื่อ UpdateProductQueue
- ให้นักศึกษาสร้างเมธอด serviceDeleteProduct() ที่มีหน้าที่ลบข้อมูลจาก Object ของคลาส Product ที่รับเข้ามาออกจากฐานข้อมูล และคืนค่า true ถ้าการลบข้อมูลสำเร็จและคืนค่าเป็น false ถ้าการลบข้อมูลไม่สำเร็จ ซึ่งการทำงานของเมธอด serviceDeleteProduct() ต้องเรียกใช้งานเมธอด deleteProduct() ของคลาส ProductService นอกจากนี้ เมธอด serviceDeleteProduct() มีการสื่อสารด้วย RabbitMQ และเชื่อมต่อกับคิวชื่อ DeleteProductQueue
- ให้นักศึกษาสร้างเมธอด serviceGetProductName() ที่มีหน้าที่ดึงข้อมูล Object ของคลาส Product ในฐานข้อมูลที่สอดคล้องกับชื่อสินค้าที่รับเข้ามา และคืนค่าในรูปแบบ Object ของ Product แต่ถ้าการทำงานไม่สำเร็จจะคืนค่าเป็น null ซึ่งการทำงานของเมธอด serviceGetProductName() ต้องเรียกใช้งานเมธอด getProductByName() ของคลาส ProductService นอกจากนี้ เมธอด serviceGetProductName() มีการสื่อสารด้วย RabbitMQ และเชื่อมต่อกับคิวชื่อ GetNameProductQueue
- ให้นักศึกษาสร้างเมธอด serviceGetAllProduct() ที่มีหน้าที่ดึงทุกข้อมูล Product ในฐานข้อมูล และคืนค่าในรูปแบบ List ของ Product โดยไม่มีการรับค่า แต่ถ้าการทำงานไม่สำเร็จจะคืนค่าเป็น null ซึ่งการทำงานของเมธอด serviceGetAllProduct() ต้องเรียกใช้งานเมธอดใดก็ได้จากคลาส ProductService นอกจากนี้ เมธอด serviceGetAllProduct() มีการสื่อสารผ่านด้วย RabbitMQ และเชื่อมต่อกับคิวชื่อ GetAllProductQueue

!!! คำแนะนำ !!!

เมธอดที่ใช้สื่อสารระหว่าง Service ถ้ามีการคืนค่าอาจจะใช้ Wrapper Class หรือชนิดข้อมูลอ้างอิง แทนการใช้งานชนิดข้อมูลพื้นฐาน