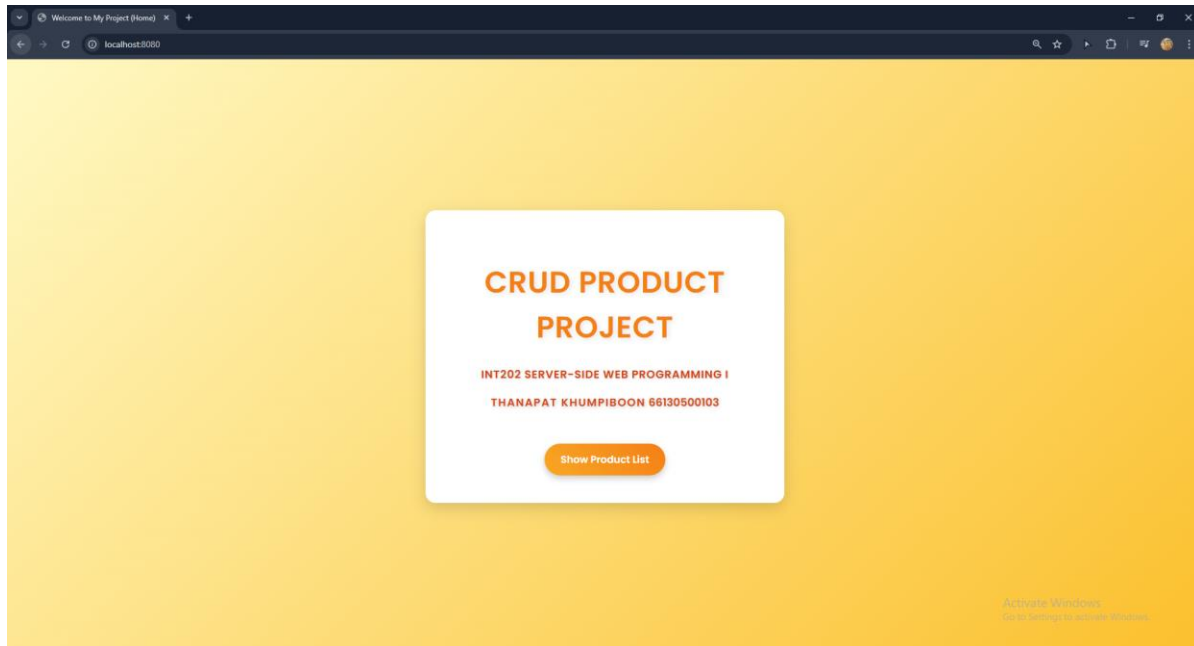


CRUD Product Project

Thanapat Khumpiboon 66130500103

โดย Web Application ที่ผมทำมาก็คือ Web Application ที่สามารถสร้างลบและแก้ไขข้อมูล product ได้ อีกทั้งยังสามารถแสดงข้อมูล product ทั้งหมดที่มีในฐานข้อมูลได้

โดยในส่วนแรกจะเป็นส่วนหน้า Home



เมื่อ กดปุ่ม Show product list ก็จะได้แสดงหน้าข้อมูลสินค้าทั้งหมดที่เรามี (ใน Database ของเรา)

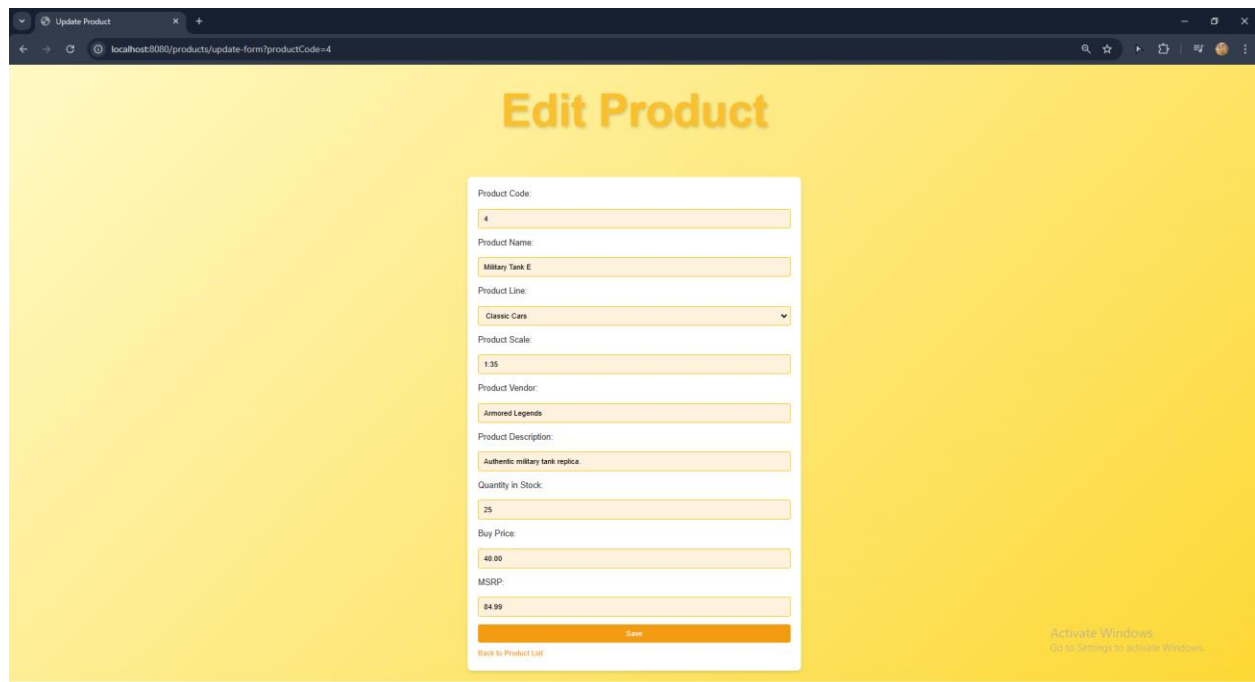
หน้า product list

Product List									
PRODUCT CODE	PRODUCT NAME	PRODUCT LINE	PRODUCT SCALE	PRODUCT VENDOR	PRODUCT DESCRIPTION	QUANTITY IN STOCK	BUY PRICE	MSRP	ACTIONS
1	Classic Car Model A	Classic Cars	1:18	AutoWorld	High-quality die-cast model of a classic car.	50	25.00	49.99	Edit Delete
2	Vintage Truck B	Motorcycles	1:24	Road Legends	Detailed replica of a vintage truck.	30	35.00	69.99	Edit Delete
3	Sports Car C	Vintage Cars	1:18	Speed Masters	Sleek design sports car model.	40	45.00	89.99	Edit Delete
4	Military Tank E	Trucks and Buses	1:35	Armored Legends	Authentic military tank replica.	25	40.00	84.99	Edit Delete
S10_1678	1969 Harley Davidson Ultimate Chopper	Planes	1:10	Min Lin Diecast	This replica features working kickstand, front suspension, gear-shift lever, footbrake lever, drive chain, wheels and steering. All parts are particularly delicate due to their precise scale and require special care and attention.	7933	48.81	95.70	Edit Delete

โดยในหน้า product list จะแสดงข้อมูล product ทั้งหมดที่มีใน database โดยก็จะแสดงออกมาเป็นตารางที่แสดง productcode productname productline product scale productvendor product description quantity instock buyprice msrp และมีปุ่ม Action ก็ปุ่ม Edit และ Delete

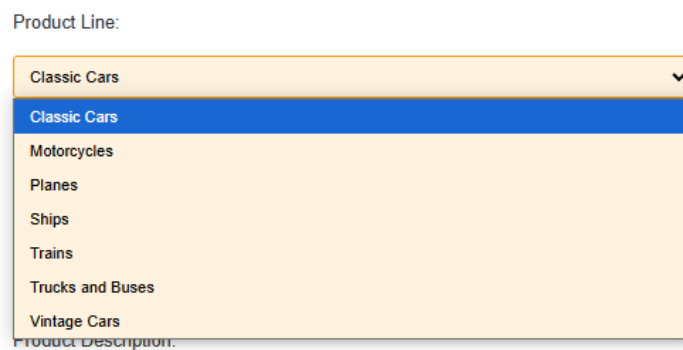
โดยปุ่ม Edit ประโยชน์ของมันก็คือจะสามารถแก้ไขข้อมูลใน Product นั้นๆได้ โดยเมื่อกดปุ่มก็จะทำการแสดงข้อมูล Product นั้นๆ

หน้า Edit Product



โดย Product Code จะไม่สามารถแก้ไขได้เนื่องจากเป็น Primary key หากอนุญาตให้แก้ไขอาจทำให้เกิดข้อผิดพลาดได้

ในส่วน product line จะทำเป็น dropdown ให้เลือก



ส่วน quantity buy price msrp เราจะกำหนดให้ใส่ได้แค่ตัวเลขเท่านั้น (input type = number)

สมมติว่าผมอยากเปลี่ยน productline จาก classic car เป็น planes

Product Code:
4

Product Name:
Military Tank E

Product Line:
Planes

Product Scale:
1:35

Product Vendor:
Armored Legends

Product Description:
Authentic military tank replica.

Quantity in Stock:
25

Buy Price:
40.00

MSRP:
84.99

Save

[Back to Product List](#)

เมื่อกด save จะกลับไปแสดงหน้า product list ใหม่ ซึ่งจะสังเกตเห็นได้ว่าข้อมูลถูกอัปเดตแล้ว

Product updated successfully!!

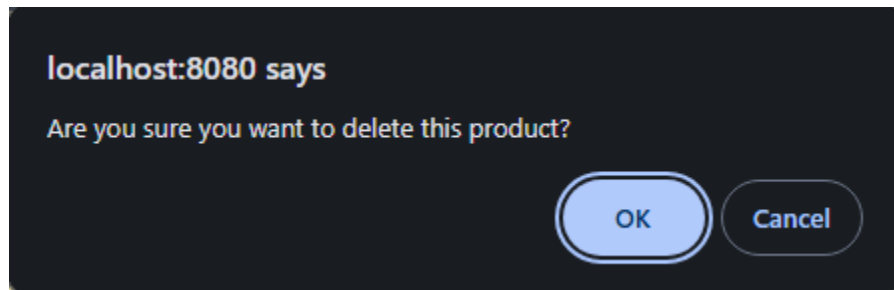
Product List

PRODUCT CODE	PRODUCT NAME	PRODUCT LINE	PRODUCT SCALE	PRODUCT VENDOR	PRODUCT DESCRIPTION	QUANTITY IN STOCK	BUY PRICE	MSRP	ACTIONS
1	Classic Car Model A	Classic Cars	1:18	AutoWorld	High-quality die-cast model of a classic car.	50	25.00	49.99	Edit Delete
2	Vintage Truck B	Motorcycles	1:24	Road Legends	Detailed replica of a vintage truck.	30	35.00	69.99	Edit Delete
3	Sports Car C	Vintage Cars	1:18	Speed Masters	Sleek design sports car model.	40	45.00	89.99	Edit Delete
4	Military Tank E	Planes	1:35	Armored Legends	Authentic military tank replica.	25	40.00	84.99	Edit Delete

แต่หากกด Back to Product List ก็จะได้แสดงหน้า product List เหมือนกันแต่ข้อมูลจะไม่ถูกแก้ไขต้องกด save ก่อน

โดยเมื่อแก้ไขสำเร็จก็จะมี pop ข้อความทางด้านขวามือว่าข้อมูลถูกอัปเดตแล้วเป็นเวลา 3 วิ

ในส่วนปุ่ม **Delete** หากเราคลิกก็จะทำการลบข้อมูล product นั้นไปทันทีแล้วก็ pop notification เตือนเหมือนกัน
แต่ก่อนจะลบจะทำการ alert ถามอีกรอบว่าแน่ใจจะลบข้อมูลมั้ย



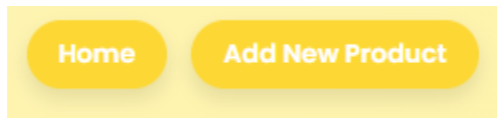
หาก ok ก็ทำการลบข้อมูลทันที

Product deleted successfully!!

Product List									
PRODUCT CODE	PRODUCT NAME	PRODUCT LINE	PRODUCT SCALE	PRODUCT VENDOR	PRODUCT DESCRIPTION	QUANTITY IN STOCK	BUY PRICE	MSRP	ACTIONS
1	Classic Car Model A	Classic Cars	1:18	AutoWorld	High-quality die-cast model of a classic car.	50	25.00	49.99	<button>Edit</button> <button>Delete</button>
2	Vintage Truck B	Motorcycles	1:24	Road Legends	Detailed replica of a vintage truck.	30	35.00	69.99	<button>Edit</button> <button>Delete</button>
3	Sports Car C	Vintage Cars	1:18	Speed Masters	Sleek design sports car model.	40	45.00	89.99	<button>Edit</button> <button>Delete</button>
S10_1878	1969 Harley Davidson Ultimate Chepper	Planes	1:10	Min Lin Diecast	This replica features working kickstand, front suspension, gear-shift lever, footbrake lever, drive chain, wheels and steering. All parts are particularly delicate due to their precise scale and require special care and attention.	7933	48.81	95.70	<button>Edit</button> <button>Delete</button>

จะสังเกตได้ว่าข้อมูลที่มี productcode = 4 ได้หายไปแล้ว (ในที่นี้ผมลบข้อมูลที่มี productcode = 4)

โดยส่วนต่อไปจะเป็นส่วน Home และ Add New Button ในหน้า Product List



Home ก็คือกลับไปหน้าแรกหน้า home นั่นเอง

ในส่วน Add New Button

หน้า product form

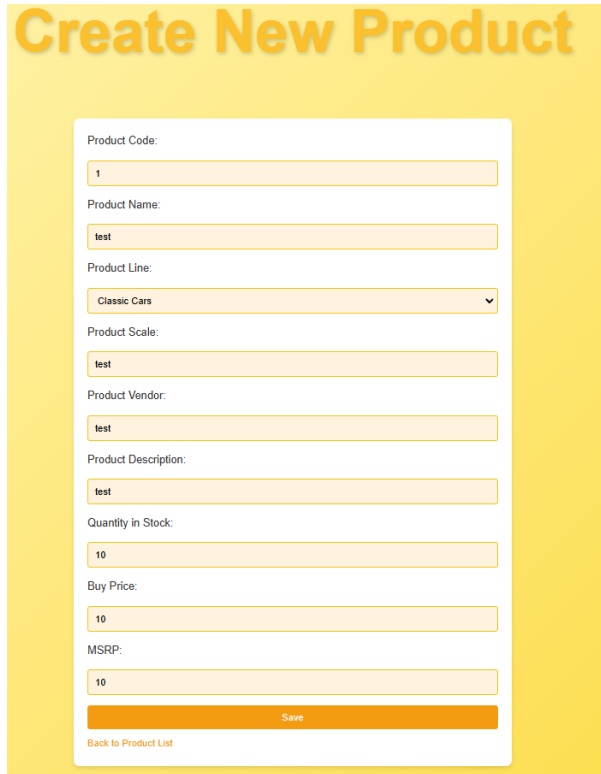
A screenshot of a web browser window. The address bar shows 'localhost:8080/products/product-form'. The page has a yellow background with the title 'Create New Product' in large, bold, orange letters. Below the title is a white form with several input fields: 'Product Code:', 'Product Name:', 'Product Line:' (with a dropdown menu showing 'Classic Cars'), 'Product Scale:', 'Product Vendor:', 'Product Description:', 'Quantity in Stock:', 'Buy Price:', and 'MSRP:'. At the bottom of the form is an orange 'Save' button and a small link 'Back to Product List'. In the bottom right corner of the page, there is a small text 'Activate Windows Go to Settings to activate Windows.'

ก็คือกรอกข้อมูลเพื่อทำการเพิ่มข้อมูล product ใหม่

โดย product code จะห้ามใส่ product code ที่ซ้ำกับที่มีอยู่แล้วหากซ้ำก็จะทำการ pop แจ้งเตือนและให้กรอกข้อมูลใหม่

ดังรูปในหน้าถัดไป

ในที่นี้เราเพิ่มข้อมูลที่มี product code = 1 ซึ่งซ้ำกับที่มีอยู่แล้วใน productlist

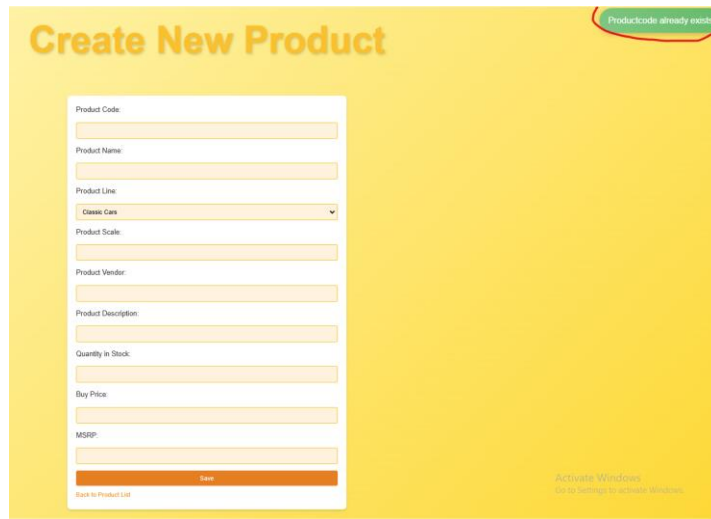


The image shows a 'Create New Product' form on a yellow background. The form is a white box with orange borders. It contains the following fields and values:

- Product Code: 1
- Product Name: test
- Product Line: Classic Cars (dropdown menu)
- Product Scale: test
- Product Vendor: test
- Product Description: test
- Quantity in Stock: 10
- Buy Price: 10
- MSRP: 10

At the bottom of the form, there is a 'Save' button and a 'Back to Product List' link.

เมื่อตอน add ก็แสดงผลดังนี้



The image shows the same 'Create New Product' form, but with a red error message at the top right: 'Product code already exists.' The form fields are empty, and the 'Save' button is still visible at the bottom. There is also a small 'Back to Product List' link at the bottom left of the form.

ก็เดี๋ยวนี product form ให้เพิ่มข้อมูล product ใหม่อีกรอบและส่งแจ้งเตือนด้วยว่า productcode นี้มีอยู่แล้ว (ไม่สามารถเพิ่มได้)

โดยหากเราเพิ่มข้อมูลที่ถูกต้อง ก็จะเพิ่มสินค้าได้ดังรูปต่อไปนี้

Product Code:

Product Name:

Product Line:

Product Scale:

Product Vendor:

Product Description:

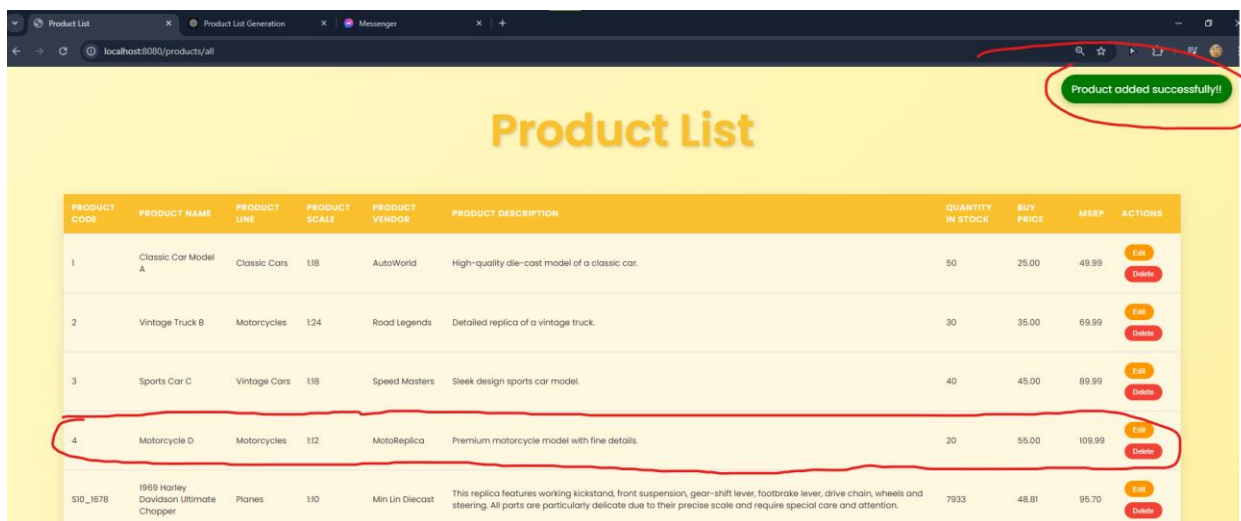
Quantity in Stock:

Buy Price:

MSRP:

[Back to Product List](#)

หลังจากกด save

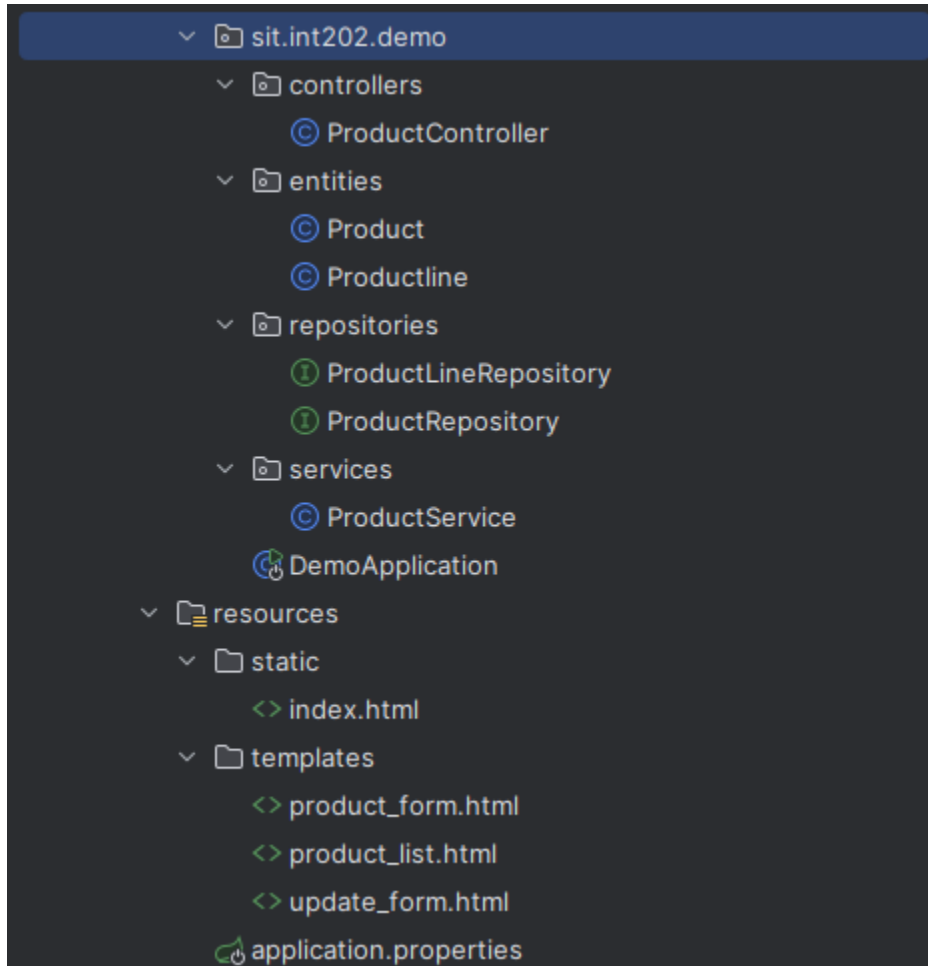


Product List

PRODUCT CODE	PRODUCT NAME	PRODUCT LINE	PRODUCT SCALE	PRODUCT VENDOR	PRODUCT DESCRIPTION	QUANTITY IN STOCK	BUY PRICE	MSRP	ACTIONS
1	Classic Car Model A	Classic Cars	1:18	AutoWorld	High-quality die-cast model of a classic car.	50	25.00	49.99	Edit Delete
2	Vintage Truck B	Motorcycles	1:24	Road Legends	Detailed replica of a vintage truck.	30	35.00	69.99	Edit Delete
3	Sports Car C	Vintage Cars	1:18	Speed Masters	Sleek design sports car model.	40	45.00	89.99	Edit Delete
4	Motorcycle D	Motorcycles	1:12	MotoReplica	Premium motorcycle model with fine details.	20	55.00	109.99	Edit Delete
10_1878	1969 Harley Davidson Ultimate Chopper	Planes	1:10	Min Lin Diecast	This replica features working kickstand, front suspension, gear-shift lever, footbrake lever, drive chain, wheels and steering. All parts are particularly delicate due to their precise scale and require special care and attention.	7933	48.81	95.70	Edit Delete

ก็จะทำการเพิ่มข้อมูลแล้วกลับมาแสดงหน้า product list ใหม่ ซึ่งจะเห็นว่าข้อมูลได้ถูกเพิ่มแล้ว

ส่วน source program



Controller จะรับคำขอจากผู้ใช้ (จะอยู่ในไฟล์ html) โดยส่งคำสั่งไปยัง Service

โดย Service จะจัดการพวก logic ต่างๆ และติดต่อกับ repository เพื่อจัดการกับข้อมูลใน database

Index.html จะเป็นหน้า home

Product_list ใช้สำหรับแสดงสินค้า

Product_from ใช้สำหรับเพิ่มสินค้าใหม่

update_from ใช้สำหรับแก้ไขข้อมูลสินค้า

Read ข้อมูล product

เริ่มจาก index.html หรือหน้า home ส่ง request ด้วย products/all ไปหา productcontroller

```
<div class="wrapper">
  <h1>CRUD Product Project</h1>
  <div class="highlight">
    <span class="course">INT202 SERVER-SIDE WEB PROGRAMMING I</span>
  </div>
  <div class="highlight">
    <span class="name">Thanapat Khumpiboon</span>
    <span class="student-id">66130500103</span>
  </div>
  <a href="products/all">Show Product List</a>
</div>
```

ทำการแสดงข้อมูลใน product list โดย ModelMap ใช้ส่งข้อมูลจาก Controller ไปยัง View(product_list)

แล้วก็ดึงข้อมูลสินค้าใน repositories ผ่าน productservice

```
@GetMapping("/all")
public String allProducts(ModelMap modelMap) {
    modelMap.addAttribute("products", productService.getAllProducts());
    return "product_list";
}
```

ใช้ each เพื่อแสดง product ทั้งหมด

```
<tr th:each="product : ${products}">
  <td th:text="${product.productCode}"></td>
  <td th:text="${product.productName}"></td>
  <td th:text="${product.productLine.productLine}"></td>
  <td th:text="${product.productScale}"></td>
  <td th:text="${product.productVendor}"></td>
  <td th:text="${product.productDescription}"></td>
  <td th:text="${product.quantityInStock}"></td>
  <td th:text="${product.buyPrice}"></td>
  <td th:text="${product.msrp}"></td>
  <td>
```

Create New Product

ส่ง request ไปให้ controller

```
<div class="btn-container">
  <a href="/" class="btn-home">Home</a>
  <a href="/products/product-form" class="btn-add">Add New Product</a>
</div>
```

จากนั้นจะส่ง view product_form มาให้เรากรอกข้อมูล

```
@GetMapping("/product-form")
public String getProductForm() {
    return "product_form";
}
```

โดยใน product_form จะมี action เมื่อกด save หลังจากกรอกข้อมูลก็จะส่ง request ไปที่ add-product ใน controller

```
<div class="container">
  <form action="/products/add-product" method="post">
    <label for="productCode">Product Code:</label>
    <input type="text" name="productCode" id="productCode" required>

    <label for="productName">Product Name:</label>
    <input type="text" name="productName" id="productName" required>

    <label for="productLine">Product Line:</label>
    <select name="productLine.productLine" id="productLine" required>
      <option value="Classic Cars">Classic Cars</option>
      <option value="Motorcycles">Motorcycles</option>
      <option value="Planes">Planes</option>
      <option value="Ships">Ships</option>
      <option value="Trains">Trains</option>
      <option value="Trucks and Buses">Trucks and Buses</option>
      <option value="Vintage Cars">Vintage Cars</option>
    </select>

    <label for="productScale">Product Scale:</label>
    <input type="text" name="productScale" id="productScale" required>

    <label for="productVendor">Product Vendor:</label>
    <input type="text" name="productVendor" id="productVendor" required>

    <label for="productDescription">Product Description:</label>
    <input type="text" name="productDescription" id="productDescription" required>

    <label for="quantityInStock">Quantity in Stock:</label>
    <input type="number" name="quantityInStock" id="quantityInStock" required min="0">

    <label for="buyPrice">Buy Price:</label>
    <input type="number" name="buyPrice" id="buyPrice" step="0.01" required min="0">

    <label for="msrp">MSRP:</label>
    <input type="number" name="msrp" id="msrp" step="0.01" required min="0">

    <input type="submit" value="Save">
  </form>
</div>
```

```

@GetMapping("/{add-product}")
public String addProduct() {
    return "product_form";
}

// add new product
@PostMapping("/{add-product}")
public String addProduct(Product product, ModelMap modelMap, RedirectAttributes redirectAttributes) {
    //if product cannot added,return to product-form page and pop error
    if (productService.isProductExists(product.getProductCode())) {
        redirectAttributes.addFlashAttribute( attributeName: "DuplicateProductCodeMessage", attributeValue: "Productcode already exists");
        return "redirect:/products/product-form";
    }
    //If the product can be added, return to the product list page.
    Product product1 = productService.addProduct(product);
    modelMap.addAttribute( attributeName: "product", product1);
    redirectAttributes.addFlashAttribute( attributeName: "SuccessMessage", attributeValue: "Product added successfully!!");
    return "redirect:/products/all";
}

```

โดย controller ก็จะมีการเช็ค productcode ว่าซ้ำมั้ย ผ่านการเรียกใช้ services method

โดยหากเข้าเงื่อนไขนี้ก็จะทำการส่ง message กลับไปแล้วก็ทำการ return redirect:/products/product-form ซึ่งหมายความว่า จะทำการโหลดหน้า product_form ให้กรอกข้อมูลใหม่นั้นเองโดยการใช้ redirect ก็เพื่อป้องกันการซ้ำกันของข้อมูล

แต่ถ้าหาก productcode นั้นไม่ซ้ำก็จะทำการเพิ่มข้อมูลสินค้านั้นผ่าน product service แล้วก็ส่ง messege ไปให้ด้วยว่าการเพิ่มสำเร็จ จากนั้นจะส่งไปยังหน้าแสดงข้อมูลสินค้าโดยข้อมูลจะถูกเพิ่มแล้ว

Update Product

```
<a th:href="@{'/products/update-form?productCode='+${product.productCode}}">
  <button class="update-btn">Edit</button>
</a>
```

ในหน้า product list เราจะเอาแค่ productcode ที่เราต้องการแก้มันเดี่ยวจากนั้นส่ง request ไปหา controller ที่ update-form

```
@GetMapping("/update-form")
public String getProductForm(@RequestParam String productCode, ModelMap modelMap) {
    Product productform = productService.getProductByCode(productCode);
    modelMap.addAttribute("product", productform);
    return "update_form";
}
```

โดยใน controller ก็จะจัดการดึงข้อมูลผ่าน service จากนั้นก็จะส่งข้อมูลนั้นไปยัง view แล้วก็แสดงหน้า update_form ให้ user แก้ไขข้อมูล โดยมีแค่ productcode ที่เราจะกำหนด readonly ไว้จากกรุปข้างล่าง

```
<form action="/products/update-product" method="post">
  <label for="productCode">Product Code:</label>
  <input type="text" name="productCode" id="productCode" th:value="${product.productCode}" readonly required>

  <label for="productName">Product Name:</label>
  <input type="text" name="productName" id="productName" th:value="${product.productName}" required>

  <label for="productLine">Product Line:</label>
  <select name="productLine.productLine" id="productLine" required>
    <option value="Classic Cars" th:selected="${product.productLine == 'Classic Cars'}">Classic Cars</option>
    <option value="Motorcycles" th:selected="${product.productLine == 'Motorcycles'}">Motorcycles</option>
    <option value="Planes" th:selected="${product.productLine == 'Planes'}">Planes</option>
    <option value="Ships" th:selected="${product.productLine == 'Ships'}">Ships</option>
    <option value="Trains" th:selected="${product.productLine == 'Trains'}">Trains</option>
    <option value="Trucks and Buses" th:selected="${product.productLine == 'Trucks and Buses'}">Trucks and Buses</option>
    <option value="Vintage Cars" th:selected="${product.productLine == 'Vintage Cars'}">Vintage Cars</option>
  </select>

  <label for="productScale">Product Scale:</label>
  <input type="text" name="productScale" id="productScale" th:value="${product.productScale}" required>

  <label for="msrp">MSRP:</label>
  <input type="number" name="msrp" id="msrp" th:value="${product.msrp}" step="0.01" required min="0">

  <input type="submit" value="Save">
</form>
```

โดยเราจะดึงข้อมูลจากที่ controller ส่งมาให้มันเอง หลังจากเราแก้ไขข้อมูลสินค้าเรียบร้อยแล้วเมื่อกด save จะ action ส่ง request

ไปหา controller ที่ update-product

```
@PostMapping("/{update-product}")
public String updateProduct(Product product, ModelMap modelMap, RedirectAttributes redirectAttributes) {
    Product product1 = productService.updateProduct(product);
    redirectAttributes.addFlashAttribute(attributeName: "updateMessage", attributeValue: "Product updated successfully!!");
    modelMap.addAttribute(attributeName: "product", product1);
    return "redirect:/products/all";
}
```

โดย update-product ก็จะทำให้การแก้ไข product นั้นผ่าน productService แล้วก็ส่ง request ไปยัง all เพื่อที่จะแสดงสินค้าทั้งหมด

Delete Product

User ส่ง request ไปยัง delete-product ใน controller

```
<a th:href="@{/products/delete-product?productCode='${product.productCode}'}" onclick="return confirmDelete();" >
    <button class="delete-btn">Delete</button>
</a>
```

จากนั้นก็ทำ controller ก็จะทำให้การเรียกใช้ service เพื่อลบข้อมูลใน repositories แล้วก็ส่ง request ไปยัง all เพื่อแสดงสินค้าทั้งหมด

```
@GetMapping("/{delete-product}")
public String deleteProduct(ModelMap modelMap, @RequestParam String productCode, RedirectAttributes redirectAttributes) {
    Product oldproduct = productService.deleteProduct(productCode);
    redirectAttributes.addFlashAttribute(attributeName: "deleteMessage", attributeValue: "Product deleted successfully!!");
    modelMap.addAttribute(attributeName: "product", oldproduct);
    return "redirect:/products/all";
}
```

CRUD

ใช้ redirect เพื่อให้ไปยัง url นั้นใหม่ ป้องกันการส่ง request ซ้ำ เช่นการลบข้อมูลเดิมซ้ำหากเรากด refresh หน้า

Notification ในส่วน product list

```
<!-- Notifications -->
<div th:if="${SuccessMessage != null}" class="notification show">
    <span th:text="${SuccessMessage}"></span>
</div>
<div th:if="${deleteMessage != null}" class="notification show delete">
    <span th:text="${deleteMessage}"></span>
</div>
<div th:if="${updateMessage != null}" class="notification show update">
    <span th:text="${updateMessage}"></span>
</div>
```

เมื่อข้อมูลถูกเพิ่มลบและแก้ไขสำเร็จ ก็จะทำการ pop ข้อความขึ้นมาว่า process นั้นสำเร็จแล้ว

Notification ในส่วน add product

```
<div th:if="${DuplicateProductCodeMessage}" class="notification show Duplicate" >
  <span th:text="${DuplicateProductCodeMessage}"></span>
</div>
```

ข้อความที่ถูก pop จะ pop แค่ชั่วคราวแล้วหายไป

```
<script>
  function confirmDelete() { Show usages
    return confirm("Are you sure you want to delete this product?");
  }

  window.onload = function() {
    var notification = document.querySelector('.notification.show');
    if (notification) {
      setTimeout(function() {
        notification.classList.remove('show');
      }, 3000); // Remove after 3 seconds
    }
  }
</script>
```