TRƯỜNG ĐẠI HỌC HỌC VĂN LANG

**KHOA CÔNG NGHỆ THÔNG TIN**

A red and black shield with white logo

Description automatically generated

**TÀI LIỆU THỰC HÀNH**

**SPRING BOOT**

**CHƯƠNG 09: RESTful API**

Giảng viên biên soạn: Ths. Nguyễn Minh Tân

2024

**BUỔI 9: RESTful API**

1. **MỤC TIÊU**

Sau khi học xong bài này, Sinh viên có thể nắm được:

- *Nắm vững các bước cơ bản để xây dựng một RESTful API trong Spring Boot*

- *Biết cách sử dụng Spring Data JPA để tương tác với cơ sở dữ liệu trong các API*

- *Hiểu được vai trò của các endpoint API để xử lý các yêu cầu HTTP.*

- *Biết cách tạo giao diện người dùng đơn giản với HTML và AJAX để tương tác với API một cách hiệu quả, không cần tải lại trang.*

1. **NỘI DUNG THỰC HÀNH:**

**6.1 Yêu cầu các chức năng cần triển khai**

**6.1.1 Xây dựng các API để quản lý sản phẩm**

- Thêm sản phẩm mới.

- Cập nhật thông tin sản phẩm.

- Xóa sản phẩm.

- Hiển thị danh sách tất cả sản phẩm.

**6.1.2 Giao diện người dùng**

- Trang web sử dụng AJAX để:

- Hiển thị danh sách sản phẩm.

- *Gửi yêu cầu thêm sản phẩm mới tới server mà không cần tải lại trang.*

- Cập nhật thông tin một sản phẩm.

- Xóa một sản phẩm.

**6.2 Hướng dẫn thực hiện**

Tạo RESTful Controller `ProductApiController`

|  |
| --- |
| package com.vanlang.webbanhang.controller;  import com.vanlang.webbanhang.model.Product; import com.vanlang.webbanhang.service.ProductService; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.http.ResponseEntity; import org.springframework.web.bind.annotation.\*;  import java.util.List;  @CrossOrigin @RestController @RequestMapping("/api/products") public class ProductApiController {  @Autowired  private ProductService productService;   @GetMapping  public List<Product> getAllProducts() {  return productService.getAllProducts();  }   @PostMapping  public Product createProduct(@RequestBody Product product) {  return productService.addProduct(product);  }   @GetMapping("/{id}")  public ResponseEntity<Product> getProductById(@PathVariable Long id) {  Product product = productService.getProductById(id).orElseThrow(() -> new RuntimeException("Product not found on :: " + id));  return ResponseEntity.*ok*().body(product);  }   @PutMapping("/{id}")  public ResponseEntity<Product> updateProduct(@PathVariable Long id, @RequestBody Product productDetails) {  Product product = productService.getProductById(id).orElseThrow(() -> new RuntimeException("Product not found on :: " + id));  product.setName(productDetails.getName());  product.setPrice(productDetails.getPrice());  product.setDescription(productDetails.getDescription());  final Product updatedProduct = productService.addProduct(product);  return ResponseEntity.*ok*(updatedProduct);  }   @DeleteMapping("/{id}")  public ResponseEntity<Void> deleteProduct(@PathVariable Long id) {  Product product = productService.getProductById(id).orElseThrow(() -> new RuntimeException("Product not found on :: "  + id));  productService.deleteProductById(id);  return ResponseEntity.*ok*().build();  } }a |

Tạo dự án front-end để tương tác với API

|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Product Management</title>  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script> </head> <body> <div class="container mt-5"><h1>Product Management</h1>  <button onclick="loadProducts()" class="btn btn-primary mb-3">Refresh Products</button>  <table class="table table-bordered">  <thead>  <tr>  <th>ID</th>  <th>Name</th>  <th>Price</th>  <th>Description</th>  <th>Actions</th>  </tr>  </thead>  <tbody id="productList"></tbody>  </table> <!-- Form to add/update a product -->  <form id="productForm"><input type="hidden" id="productId">  <div class="mb-3"><label for="name" class="form-label">Name:</label> <input type="text" class="form-control"  id="name" required></div>  <div class="mb-3"><label for="price" class="form-label">Price:</label> <input type="number" class="form-control"  id="price" required></div>  <div class="mb-3"><label for="description" class="form-label">Description:</label> <input type="text"  class="form-control"  id="description">  </div>  <button type="submit" class="btn btn-success">Save Product</button>  </form> </div> <script> $(*document*).ready(function () {  loadProducts();  $("#productForm").on('submit', function (e) {  e.preventDefault();  saveProduct();  }); });  function loadProducts() {  $.ajax({  url: '/api/products', type: 'GET', success: function (products) {  let productList = '';  $.each(products, function (index, product) {  productList += `<tr> <td>${product.id}</td> <td>${product.name}</td> <td>${product.price}</td> <td>${product.description}</td> <td> <button onclick="editProduct(${product.id})" class="btn btn-warning">Edit</button> <button onclick="deleteProduct(${product.id})" class="btn btn-danger">Delete</button> </td> </tr>`;  });  $('#productList').html(productList);  }  }); }  function saveProduct() {  const productData = {  id: $('#productId').val(),  name: $('#name').val(),  price: $('#price').val(),  description: $('#description').val()  };  const apiUrl = productData.id ? `/api/products/${productData.id}` : '/api/products';  const apiType = productData.id ? 'PUT' : 'POST';  $.ajax({  url: apiUrl,  type: apiType,  contentType: 'application/json',  data: *JSON*.stringify(productData),  success: function () {  resetForm();  loadProducts();  }  }); }  function editProduct(id) {  $.ajax({  url: `/api/products/${id}`, type: 'GET', success: function (product) {  $('#productId').val(product.id);  $('#name').val(product.name);  $('#price').val(product.price);  $('#description').val(product.description);  }  }); }  function deleteProduct(id) {  if (confirm('Are you sure you want to delete this product?')) {  $.ajax({  url: `/api/products/${id}`, type: 'DELETE', success: function () {  loadProducts();  }  });  } }  function resetForm() {  $('#productId').val('');  $('#name').val('');  $('#price').val('');  $('#description').val(''); } </script> </body> </html> |

**TÀI LIỆU THAM KHẢO**

1. Clean Code: A Handbook of Agile Software Craftsmanship, Robert C. Martin, 2008

2. Eloquent JavaScript: A Modern Introduction to Programming, Marijn Haverbeke, 2018 (3rd edition)

3. You Don't Know JS (series), Kyle Simpson, 2015

4. MDN Web Docs (https://developer.mozilla.org)

5. Stack Overflow (https://stackoverflow.com)

6. GitHub (https://github.com)

7. W3Schools (https://www.w3schools.com)

8. Smashing Magazine (https://www.smashingmagazine.com)