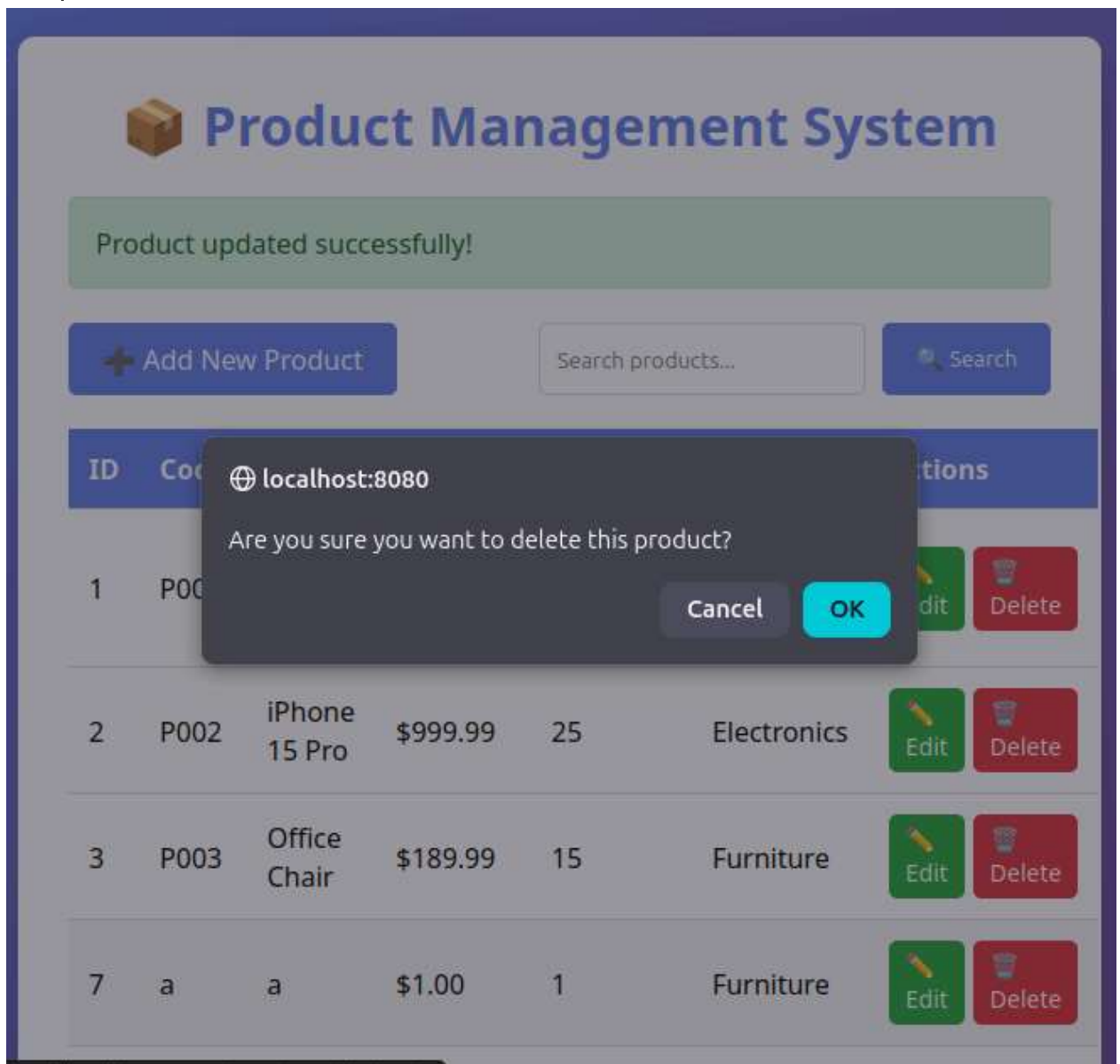


1) Delete flow:



- User click delete button
- Browser sends post request to controller
- Controller maps the post request with endpoint "/delete/id" to method deleteProduct()
- Controller calls productService.deleteProduct(id)
- Service calls productRepository.deleteByld(id)
- JPA queries database and deletes the product
- Controller sends redirect to "/products" with success/error message
- Controller receives request with message and forward to product-list
- Thymeleaf renders HTML
- Response sent to browser

2) Update flow:

- User click edit button
- Browser sends post request to controller
- Controller maps the post request with end point “/edit/id” to method showEditForm()

```
// Show form for editing product
@GetMapping("/edit/{id}")
public String showEditForm(@PathVariable Long id, Model model, RedirectAttributes
redirectAttributes) {
    return productService.getProductById(id)
        .map(product -> {
            model.addAttribute("product", product);
            return "product-form";
        })
        .orElseGet(() -> {
            redirectAttributes.addFlashAttribute("error", "Product not
            found");
            return "redirect:/products";
        });
}
```

- Controller calls productService.getProductById(id)
- Service calls productRepository.findById(id)
- JPA queries database and return 1 product
- Controller set model attribute with the product and forward to product-form
- Thymeleaf renders HTML
- Response sent to browser



Edit Product

Product Code *

Product Name *

Price (\$) *



Quantity *



Category *



Description

- User fills in the form and clicks save product button
- Browser sends post request with endpoint “/save” to Controller
- Controller maps that request to method saveProduct()

```
// Save product (create or update)
@PostMapping("/save")
public String saveProduct(@ModelAttribute("product") Product
product, RedirectAttributes redirectAttributes) {
    try {
        productService.saveProduct(product);
        redirectAttributes.addFlashAttribute("message",
            product.getId() == null ? "Product added
            successfully!" : "Product updated
            successfully!");
    } catch (Exception e) {
        redirectAttributes.addFlashAttribute("error", "Error
        saving product: " + e.getMessage());
    }
    return "redirect:/products";
}
```

- Controller calls productService.saveProduct(product)
- Controller adds message attribute to model, depending on whether or not the product has an id (in the database or not).
- Controller sends redirect to "/products" with success/error message
- Controller receives request with message and forward to product-list
- Thymeleaf renders HTML
- Response sent to browser

7

b

b

\$1.00

1

Furniture



3) Create flow:



- User click add button
- Browser sends post request to controller
- Controller maps the post request with end point "/new" to method showNewForm()

```
// Show form for new product
@GetMapping("/new")
public String showNewForm(Model model) {
    Product product = new Product();
    model.addAttribute("product", product);
    return "product-form";
}
```

- Controller create new empty product and add that as an attribute to model
- Controller forwards to product-form
- User fills in the form and clicks save product button
- Browser sends post request with endpoint “/save” to Controller
- Controller maps that request to method saveProduct()

```
// Save product (create or update)
@PostMapping("/save")
public String saveProduct(@ModelAttribute("product") Product
product, RedirectAttributes redirectAttributes) {
    try {
        productService.saveProduct(product);
        redirectAttributes.addFlashAttribute("message",
            product.getId() == null ? "Product added
            successfully!" : "Product updated
            successfully!");
    } catch (Exception e) {
        redirectAttributes.addFlashAttribute("error", "Error
        saving product: " + e.getMessage());
    }
    return "redirect:/products";
}
```

- Controller calls productService.saveProduct(product)
- Controller adds message attribute to model, depending on whether or not the product has an id (in the database or not).
- Controller sends redirect to “/products” with success/error message
- Controller receives request with message and forward to product-list
- Thymeleaf renders HTML
- Response sent to browser