

Eliminar un Producto con Spring



Eliminación de un producto en Spring Boot

En esta guía, agregaremos la funcionalidad para **eliminar un producto** desde el backend en **Spring Boot**. Explicaremos cómo funciona el código y cómo probarlo en **Postman**.

Paso 1: Modificar `ProductoControlador.java` para agregar el método `eliminarProducto()`

Código actualizado

```
package gm.inventarios.controlador;

import gm.inventarios.excepcion.RecursoNoEncontradoExcepcion;
import gm.inventarios.modelo.Producto;
import gm.inventarios.servicio.ProductoServicio;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
```

```
import java.util.HashMap;
import java.util.List;
import java.util.Map;

@RestController
//http://localhost:8080/inventario-app
@RequestMapping("inventario-app")
@CrossOrigin(value = "http://localhost:4200")
public class ProductoControlador {

    private static final Logger logger =
        LoggerFactory.getLogger(ProductoControlador.class);

    @Autowired
    private ProductoServicio productoServicio;

    //http://localhost:8080/inventario-app/productos
    @GetMapping("/productos")
    public List<Producto> obtenerProductos(){
        List<Producto> productos = this.productoServicio.listarProductos();
        logger.info("Productos obtenidos:");
        productos.forEach((producto -> logger.info(producto.toString())));
        return productos;
    }

    @PostMapping("/productos")
    public Producto agregarProducto(@RequestBody Producto producto){
        logger.info("Producto a agregar: " + producto);
        return this.productoServicio.guardarProducto(producto);
    }

    @GetMapping("/productos/{id}")
    public ResponseEntity<Producto> obtenerProductoPorId(
        @PathVariable int id){
        Producto producto =
            this.productoServicio.buscarProductoPorId(id);
        if(producto != null)
            return ResponseEntity.ok(producto);
        else
            throw new RecursoNoEncontradoExcepcion("No se encontro el id: " + id);
    }

    @PutMapping("/productos/{id}")
    public ResponseEntity<Producto> actualizarProducto(
        @PathVariable int id,
        @RequestBody Producto productoRecibido){
        Producto producto = this.productoServicio.buscarProductoPorId(id);
        if(producto == null)
            throw new RecursoNoEncontradoExcepcion("No se encontro el id: " + id);
        producto.setDescripcion(productoRecibido.getDescripcion());
        producto.setPrecio(productoRecibido.getPrecio());
        producto.setExistencia(productoRecibido.getExistencia());
        this.productoServicio.guardarProducto(producto);
        return ResponseEntity.ok(producto);
    }

    @DeleteMapping("/productos/{id}")
    public ResponseEntity<Map<String, Boolean>>
    eliminarProducto(@PathVariable int id){
        Producto producto = productoServicio.buscarProductoPorId(id);
        if (producto == null)
```

```

        throw new RecursoNoEncontradoExcepcion("No se encontró el id: " + id);
        this.productoServicio.eliminarProductoPorId(producto.getIdProducto());
        Map<String, Boolean> respuesta = new HashMap<>();
        respuesta.put("eliminado", Boolean.TRUE);
        return ResponseEntity.ok(respuesta);
    }
}

```

Paso 2: Explicación del método `eliminarProducto()`

```

@DeleteMapping("/productos/{id}")
public ResponseEntity<Map<String, Boolean>> eliminarProducto(@PathVariable int
id) {
    Producto producto = productoServicio.buscarProductoPorId(id);
    if (producto == null)
        throw new RecursoNoEncontradoExcepcion("No se encontró el ID: " + id);

    this.productoServicio.eliminarProductoPorId(producto.getIdProducto());

    Map<String, Boolean> respuesta = new HashMap<>();
    respuesta.put("eliminado", Boolean.TRUE);
    return ResponseEntity.ok(respuesta);
}

```

Explicación del código

1. Se usa `@DeleteMapping("/productos/{id}")`:
 - Define que este método **manejará solicitudes DELETE** en la URL `/productos/{id}`.
 - El ID del producto a eliminar se envía como parámetro en la URL.
2. Se obtiene el producto con `buscarProductoPorId(id)`:

```

Producto producto = productoServicio.buscarProductoPorId(id);



- Si el producto existe, se almacena en la variable producto.
- Si el producto no existe, se lanza la excepción:

```

```

throw new RecursoNoEncontradoExcepcion("No se encontró el ID:
" + id);

```

- Esto devuelve un **error 404 (Not Found)** al cliente.

3. Se llama al servicio para eliminar el producto:

```

this.productoServicio.eliminarProductoPorId(producto.getIdProducto());

```

- Borra el producto de la base de datos.

4. Se crea una respuesta indicando que el producto fue eliminado:

```

Map<String, Boolean> respuesta = new HashMap<>();
respuesta.put("eliminado", Boolean.TRUE);

```

```
return ResponseEntity.ok(respuesta);
```

- Devuelve una respuesta JSON:

```
{  
    "eliminado": true  
}
```

Paso 3: Reiniciar la aplicación Spring Boot

Para aplicar los cambios, debemos **detener y volver a ejecutar** la aplicación.

Pasos:

1. **En IntelliJ IDEA:**
 - Clic **derecho** sobre la clase `InventariosApplication.java`.
 - Seleccionar "**Run InventariosApplication**".
 2. Esperar a que la aplicación se **inicie correctamente**.
-

Paso 4: Prueba en Postman

Ahora probaremos la funcionalidad de eliminación con **Postman**.

Prueba 1: Eliminar un producto existente

1. **Abrir Postman.**
2. **Seleccionar `DELETE` en el tipo de solicitud.**
3. **Ingresar la URL:**

```
http://localhost:8080/inventario-app/productos/4
```

4. **Presionar "Send".**

Respuesta esperada (200 OK)

```
{  
    "eliminado": true  
}
```

- **Si ves esta respuesta, el producto se eliminó correctamente.**
- **Verifica en la base de datos o con un `GET` en Postman** para confirmar que el producto ya no existe.

Prueba 2: Intentar eliminar un producto que no existe

1. Cambiar la URL a un ID inexistente, por ejemplo:

`http://localhost:8080/inventario-app/productos/99`

2. Presionar "Send".

Respuesta esperada (404 Not Found)

```
{  
    "timestamp": "01:12:39.114+00:00",  
    "status": 404,  
    "error": "Not Found",  
    "path": "/inventario-app/productos/99"  
}
```

- Esto confirma que la API maneja correctamente los errores cuando se intenta eliminar un producto inexistente.
-

Conclusión

- Se agregó el método `eliminarProducto()` en `ProductoControlador.java` para eliminar productos.
- Se implementó una validación para manejar errores cuando el producto no existe.
- Se probó la funcionalidad en Postman con solicitudes `DELETE`, confirmando que los productos se eliminan correctamente.
- Si se intenta eliminar un producto inexistente, se devuelve un error `404 Not Found`.

Con estos cambios, la API ahora permite **eliminar productos** de manera eficiente y segura.

Saludos!

Ing. Ubaldo Acosta

Fundador de [GlobalMentoring.com.mx](https://www.globalmentoring.com.mx)