Actividad integradora 3EV
Desarrollo de una aplicación Web con Spring Boot
11
Área de administración
Alta / baja / modificación de categorías
Alta / baja / modificación de productos

CONTENIDO

1.- Objetivos

- Objetivos de desarrollo web
  - Practicar la creación de controladores, vistas, mapeos, etc.
  - Practicar la creación de formularios y binding a modelos.
  - Practicar la validación de modelos con bindingResults.
  - Practicar redirecciones avanzadas y mensajes informativos.
- Objetivos de acceso a datos
  - Practicar las operaciones básicas con repositorios.

2.- Requisitos

Haber completado todas las partes anteriores de la actividad, en la que se crearon los listados de categorías y de productos.

3.- Tareas a realizar – Gestión de categorías

3.1.- Formulario de creación de categoría

Crear un formulario para la creación de nueva categoría. El formulario:

- Responderá a la misma URL que el listado, añadiendo "/new". Por ejemplo, si el listado de categorías es "/admin/categories", la URL del formulario será "/admin/categories/new".
- Usará un DTO específico como modelo, que se asociará al formulario con th:object y th:field.
- El DTO / formulario tendrá atributos / campos para los atributos de la categoría que no se generen automáticamente. Los campos autogenerados como el identificador o fechas de actualización o modificación no deben formar parte del DTO.
- El DTO usará, para validación, anotaciones Jakarta Bean Validation.
- Tendrá un botón para enviar el formulario con POST, a la misma dirección.
- Tendrá un botón/enlace para volver al listado de categorías

El sistema procesará la petición POST, y:

- Verificará posibles errores, tanto los provocados por los atributos de validación, como posibles errores de negocio (ejemplo: si no puede haber dos categorías con el mismo nombre). Si hay errores de validación de negocio, se deben añadir a bindingResults como errores globales,
- Si hay errores, se mostrará de nuevo el formulario y:
  - Si hay errores globales, se mostrará una lista al principio del formulario, antes de los campos.
  - Si hay errores en campos, se destacarán los campos con estilos de error, y se mostrará un mensaje de error junto a cada campo con error.
- Si no hay errores, se procederá a crear la categoría y:
  - Si cuando se cree la categoría se produce un error (excepción) esto se considerará error global, se añadirá a los errores globales de bindingResult y se volverá a mostrar el formulario.
  - Si no hay errores, se redirigirá (redirect) al listado de categorías, pasando un mensaje usando "flash attributes" (ejercicio de investigación: qué son y como se usan los flash attributes)

Para acceder a este formulario, se debe añadir un enlace en el listado de categorías.

3.2.- Formulario de modificación de categoría

Crear un formulario para la modificación de categoría. El formulario:

- Responderá a la misma URL que el listado, añadiendo "/edit/{id de categoría}". Por ejemplo, si el listado de categorías es "/admin/categories", la URL del formulario para editar la categoría con id 10 será "/admin/categories/edit/10".
- Usará un DTO específico como modelo, que se asociará al formulario con th:object y th:field.
- El DTO / formulario tendrá atributos / campos para todos los atributos de la categoría que no se generen automáticamente. Los campos autogenerados como el identificador o fechas de actualización o modificación no deben formar parte del DTO. Se puede valorar usar el mismo DTO que para crear categoría si se considera viable.
- El DTO usará, para validación, anotaciones Jakarta Bean Validation.
- Tendrá un botón para enviar el formulario con POST, a la misma dirección.
- Tendrá un botón/enlace para volver al listado de categorías

El sistema procesará la petición POST, y:

- Verificará posibles errores, tanto los provocados por los atributos de validación, como posibles errores de negocio (ejemplo: si se está cambiando el nombre de categoría y no puede haber dos categorías con el mismo nombre). Si hay errores de validación de negocio, se deben añadir a bindingResults como errores globales,
- Si hay errores, se mostrará de nuevo el formulario y:
    - Si hay errores globales, se mostrará una lista al principio del formulario, antes de los campos.
    - Si hay errores en campos, se destacarán los campos con estilos de error, y se mostrará un mensaje de error junto a cada campo con error.
- Si no hay errores, se procederá a modificar la categoría y:
    - Si cuando se modifique la categoría se produce un error (excepción) esto se considerará error global, se añadirá a los errores globales de bindingResult y se volverá a mostrar el formulario.
    - Si no hay errores, se redirigirá (redirect) al listado de categorías, pasando un mensaje usando "flash attributes" (de nuevo, investigar qué son y como se usan los flash attributes)

Para acceder a este formulario, se debe añadir un enlace en el listado de categorías. Un enlace por cada categoría. Se deben colocar en la columna "Acciones" que se dejó vacía en la actividad anterior.

3.3.- Formulario de borrado de categoría

Crear un formulario para el borrado de categoría. El formulario:

- Responderá a la misma URL que el listado, añadiendo "/delete/{id de categoría}". Por ejemplo, si el listado de categorías es "/admin/categories", la URL del formulario para eliminar la categoría con id 10 será "/admin/categories/delete/10".
- Cuando se cargue el formulario, mostrará los datos de la categoría, y pedirá confirmación al usuario.
- Si el usuario decide eliminar la categoría, se hará un POST a la misma dirección y:
    - Si cuando se elimine la categoría se produce un error (excepción) esto se considerará error global, se añadirá de alguna manera al modelo, y se volverá a mostrar el formulario.
    - Si no hay errores, se redirigirá (redirect) al listado de categorías, pasando un mensaje usando "flash attributes" (de nuevo, investigar qué son y como se usan
- Siempre habrá un enlace / botón para poder volver al listado de categorías sin guardar cambios.

Para acceder a este formulario, se debe añadir un enlace en el listado de categorías. Un enlace por cada categoría. Se deben colocar en la columna "Acciones" que se dejó vacía en la actividad anterior.

4.- Tareas a realizar – Gestión de productos

4.1.- Formulario de creación de producto

Crear un formulario para la creación de nuevo producto. El formulario:

- Responderá a la misma URL que el listado, añadiendo "/new". Por ejemplo, si el listado de productos es "/admin/products", la URL del formulario será "/admin/products/new".
- Usará un DTO específico como modelo, que se asociará al formulario con th:object y th:field.
- El DTO / formulario tendrá atributos / campos para los atributos del producto que no se generen automáticamente. Los campos autogenerados como el identificador o fechas de actualización o modificación no deben formar parte del DTO.
- El DTO usará, para validación, anotaciones Jakarta Bean Validation.
- Tendrá un botón para enviar el formulario con POST, a la misma dirección.
- Tendrá un botón/enlace para volver al listado de productos

El sistema procesará la petición POST, y:

- Verificará posibles errores, tanto los provocados por los atributos de validación, como posibles errores de negocio (ejemplo: si no puede haber dos productos con el mismo nombre). Si hay errores de validación de negocio, se deben añadir a bindingResults como errores globales,
- Si hay errores, se mostrará de nuevo el formulario y:
    - Si hay errores globales, se mostrará una lista al principio del formulario, antes de los campos.
    - Si hay errores en campos, se destacarán los campos con estilos de error, y se mostrará un mensaje de error junto a cada campo con error.
- Si no hay errores, se procederá a crear el producto y:
    - Si cuando se cree el producto se produce un error (excepción) esto se considerará error global, se añadirá a los errores globales de bindingResult y se volverá a mostrar el formulario.
    - Si no hay errores, se redirigirá (redirect) al listado de productos, pasando un mensaje usando "flash attributes" (ejercicio de investigación: qué son y como se usan los flash attributes)

Para acceder a este formulario, se debe añadir un enlace en el listado de productos.

4.2.- Formulario de modificación de producto

Crear un formulario para la modificación de producto. El formulario:

- Responderá a la misma URL que el listado, añadiendo "/edit/{id de producto}". Por ejemplo, si el listado de productos es "/admin/products", la URL del formulario para editar el producto con id 10 será "/admin/products/edit/10".
- Usará un DTO específico como modelo, que se asociará al formulario con th:object y th:field.
- El DTO / formulario tendrá atributos / campos para todos los atributos del producto que no se generen automáticamente. Los campos autogenerados como el identificador o fechas de actualización o modificación no deben formar parte del DTO. Se puede valorar usar el mismo DTO que para crear producto si se considera viable.
- El DTO usará, para validación, anotaciones Jakarta Bean Validation.
- Tendrá un botón para enviar el formulario con POST, a la misma dirección.
- Tendrá un botón/enlace para volver al listado de productos

El sistema procesará la petición POST, y:

- Verificará posibles errores, tanto los provocados por los atributos de validación, como posibles errores de negocio (ejemplo: si se está cambiando el nombre de producto y no puede haber dos productos con el mismo nombre). Si hay errores de validación de negocio, se deben añadir a bindingResults como errores globales,
- Si hay errores, se mostrará de nuevo el formulario y:
    - Si hay errores globales, se mostrará una lista al principio del formulario, antes de los campos.
    - Si hay errores en campos, se destacarán los campos con estilos de error, y se mostrará un mensaje de error junto a cada campo con error.
- Si no hay errores, se procederá a modificar el producto y:
    - Si cuando se modifique el producto se produce un error (excepción) esto se considerará error global, se añadirá a los errores globales de bindingResult y se volverá a mostrar el formulario.
    - Si no hay errores, se redirigirá (redirect) al listado de productos, pasando un mensaje usando "flash attributes" (de nuevo, investigar qué son y como se usan los flash attributes)

Para acceder a este formulario, se debe añadir un enlace en el listado de productos. Un enlace por cada producto. Se deben colocar en la columna "Acciones" que se dejó vacía en la actividad anterior.

4.3.- Formulario de borrado de producto

Crear un formulario para el borrado de producto. El formulario:

- Responderá a la misma URL que el listado, añadiendo "/delete/{id de producto}". Por ejemplo, si el listado de productos es "/admin/products", la URL del formulario para eliminar el producto con id 10 será "/admin/products/delete/10".
- Cuando se cargue el formulario, mostrará los datos del producto, y pedirá confirmación al usuario.
- Si el usuario decide eliminar el producto, se hará un POST a la misma dirección y:
    - Si cuando se elimine el producto se produce un error (excepción) esto se considerará error global, se añadirá de alguna manera al modelo, y se volverá a mostrar el formulario.
    - Si no hay errores, se redirigirá (redirect) al listado de productos, pasando un mensaje usando "flash attributes" (de nuevo, investigar qué son y como se usan
- Siempre habrá un enlace / botón para poder volver al listado de productos sin guardar cambios.

Para acceder a este formulario, se debe añadir un enlace en el listado de productos. Un enlace por cada producto. Se deben colocar en la columna "Acciones" que se dejó vacía en la actividad anterior.

5.- Requisitos específicos respecto a la validación

Es obligatorio implementar validaciones de tres tipos:

- Validaciones automáticas usando atributos de Jakarta Bean Validation. @NotNull, @NotBlank, @Min, etc.
- Comprobando en la capa de servicios valores, y lanzando excepciones que se capturen en el controlador, y que se añadirán a los bindingResults, rechazando el valor erróneo. Por ejemplo, comprobar que no se duplica un nombre de producto o categoría.
- Comprobando en la capa de servicios condiciones que no tienen que ver con un valor concreto, y lanzando excepciones que se capturan en el controlador, y que se añadirán a los bindingResults, como errores globales, sin asociarlas a un atributo / campo concreto. Por ejemplo, no permitir el borrado de una categoría si tiene productos asociados.

En la rúbrica se incluirán criterios relacionados con los tres tipos de validación y con la presentación al usuario de los tres tipos de error.

Si no se dispone de un caso de uso para validar condiciones “globales”, como por ejemplo si se realizan borrados en cascada, se debe añadir código en el controlador para provocar aleatoriamente errores globales. Se sugiere un código similar a este:

```
if (new Random().nextDouble() < 0.3) throw new RuntimeException("Error");
```

Este código lanza una excepción aproximadamente un 30% de las veces que se ejecuta.