Control práctico de DP1 (Grupo 3)

Enunciado

En este ejercicio, añadiremos la funcionalidad de gestión de habitaciones de recuperación que se asociarán a las visitas de mascotas realizadas en nuestra clínica. Para ello realizaremos una serie de ejercicios basados en funcionalidades que implementaremos en el sistema, y validaremos mediante pruebas unitarias. Si desea ver el resultado que arrojarían las pruebas, puedes ejecutarlas (bien mediante tu entorno de desarrollo favorito, bien con el comando "*.\mvnw test*" en la carpeta raíz del proyecto). Cada prueba correctamente pasada valdrá un punto.

Para comenzar el control debe aceptar la tarea de este control práctico a través del siguiente enlace:

https://classroom.github.com/a/wQSwu9iw

Al aceptar dicha tarea, se creará un repositorio único individual para usted, debe usar dicho repositorio para realizar el control práctico. Debe entregar la actividad en EV asociada al control check proporcionando como texto la dirección url de su repositorio personal. Recuerde que además debe entregar su solución del control.

La entrega de su solución al control se realizará mediante un único comando "*git push*" a su repositorio individual. Recuerde que debe hacer commit de todos sus cambios y push antes de cerrar sesión en la computadora y abandonar el aula, de lo contrario, su intento se evaluará como no presentado.

Su primera tarea en este control será clonar su repositorio. A continuación, deberá importar el proyecto en su entorno de desarrollo favorito y comenzar los ejercicios abajo listados. Al importar el proyecto, observará que el mismo presenta errores de compilación. No se preocupe, dichos errores irán despareciendo conforme usted vaya implementando los distintos ejercicios del control.

Nota importante 1: No modifique los nombres de las clases ni la signatura (nombre, tipo de respuesta y parámetros) de los métodos proporcionados como material de base para el control. Las pruebas que se usan para la evaluación dependen de que las clases y los métodos tengan la estructura y nombres proporcionados. Si los modifica probablemente no pueda hacer que pasen las pruebas, y obtendrá una mala calificación.

Nota importante 2: No modifique las pruebas unitarias proporcionadas como parte del proyecto bajo ningún concepto. Aunque modifique las pruebas en su copia local del proyecto, éstas serán restituidas mediante un comando git previamente a la ejecución de las pruebas para la emisión de la nota final, por lo que sus modificaciones en las pruebas no serán tenidas en cuenta en ningún momento.

Nota importante 3: Mientras haya ejercicios no resueltos habrá tests que no funcionan y, por tanto, el comando "*mvnw install*" finalizará con error. Esto es normal debido a la forma en la que está planteado el control y no hay que preocuparse por ello. Si se quiere probar la aplicación se puede ejecutar de la forma habitual pese a que "mvnw install" finalice con error.

Test 1 – Creación de la Entidad habitación de recuperación y su repositorio asociado

Se propone modificar la clase "RecoveryRoom" para que sea una entidad. Esta entidad estará alojada en el paquete "org.springframework.samples.petclinic.recoveryroom", y debe tener los siguientes atributos y restricciones:

- Un atributo de tipo Integer llamado id que actúe como clave primaria en la tabla de la base de datos relacional asociada a la entidad.
- Un atributo "name" de tipo cadena obligatorio, y cuya longitud mínima son 3 carácteres y la máxima 50.
- Un atributo "size" de tipo numérico de coma flotante (double) obligatorio, que solo podrá tomar valores positivos (cero incluido) que indicará el tamaño de la habitación de recuperación en metros cuadrados.
- Un atributo "secure" de tipo booleano obligatorio, que indicará si la habitación está protegida o no.

Se propone modificar el interfaz "RecoveryRoomRepository" alojado en el mismo paquete, para que extienda a CrudRepository. No elimine por ahora la anotación @Transient.

Test 2 – Creación de la Entidad tipo de habitación de recuperación

Modificar la clase denominada "RecoveryRoomType" alojada en el paquete "org.springframework.samples.petclinic.recoveryroom" para que sea una Entidad. Esta entidad debe tener los siguientes atributos y restricciones:

- Un atributo de tipo Integer llamado "Id" que actúe como clave primaria en la tabla de la base de datos relacional asociada a la entidad.
- Un atributo "name" de tipo cadena obligatorio, y cuya longitud minima son 5 carácteres y la máxima 50.

Se debe descomentar y anotar el método findAllRecoveryRoomTypes, del repositorio de habitaciones de recuperación, para que ejecute una consulta que permita obtener todos los tipos de habitaciones de recuperación existentes.

Test 3 – Modificación del script de inicialización de la base de datos para incluir dos habitaciones de recuperación (y los tipos de habitaciones de recuperación asociados)

Modificar el script de inicialización de la base de datos, para que se creen las siguientes habitaciones de recuperación y tipos de habitaciones de recuperación:

Habitación de recuperación 1:

- Id: 1
- Name: "Big room for dangerous animals"
- Size: 6.50
- Secure: true

Habitación de recuperación 2:

- Id: 2
- Name: "Medium box"

- Size: 1.50
- Secure: false

Tipo de habitación de recuperación 1:

- Id: 1
- Name: “room”

Tipo de habitación de recuperación 2:

- Id: 2
- Name: “box”

Test 4 – Creación de un Servicio de gestión de habitaciones de recuperación

Modificar la clase RecoveryRoomService, para que sea un servicio Spring de lógica de negocio, que permita obtener todas las habitaciones de recuperación (como una lista) usando el repositorio. No modifique por ahora la implementación de los demás métodos del servicio.

Test 5 – Modificar el repositorio para obtener un tipo de habitación de recuperación dado su nombre.

Crear una consulta personalizada que pueda invocarse a través del repositorio de habitaciones de recuperación que obtenga un tipo de habitación de recuperación por su nombre. Exponerlo a través del servicio de gestión de habitaciones de recuperación mediante el método “getRecoveryRoomType(String name)”.

Test 6 – Creación de un Formatter de tipos de habitaciones de recuperación

Implementar los métodos del formatter para la clase RecoveryRoomType (usando la clase llamada RecoveryRoomTypeFormatter que debe estar ya alojada en el mismo paquete con el que estamos trabajando). El formatter debe mostrar las habitaciones de recuperación usando la cadena de su nombre, y debe obtener un tipo de habitación de recuperación dado su nombre buscándolo en la BD. Recuerde que, si el formatter no puede parsear un valor apropiado a partir del texto proporcionado, debe lanzar una excepción de tipo ParseException.

Test 7 – Creación del método controlador para el acceso al formulario de creación/edición de habitaciones de recuperación

Crear el método controlador asociado para acceder al formulario para crear una nueva habitación de recuperación, el cual se encuentra en la carpeta webapp/WEB-INF/jsp/recoveryroom con el nombre “createOrUpdateRecoveryRoomForm.jsp”. Este formulario permite especificar el nombre, el tamaño, si es segura o no, y el tipo de habitación de recuperación asociado. El formulario debe quedar disponible a través de la url:

http://localhost:8080/recoveryroom/create

El formulario debe aparecer por defecto vacío. Los datos del formulario deben enviarse a la misma dirección usando el método post. Quizás necesite añadir algún método al servicio de gestión de habitaciones de recuperación y tipos de habitaciones de recuperación para que funcione correctamente.

El método controlador asociado debe crearse en la clase “RecoveryRoomController” del mismo paquete, y pasar al formulario un objeto de tipo RecoveryRoom con el nombre “recoveryRoom” a través del modelo.

Recuerde que como parte de la implementación debe modificar la configuración de seguridad de la aplicación para permitir que solo los usuarios administradores (authority “admin”) accedan al formulario.

Test 8 – Creación del Controlador para la creación de nuevas habitaciones de recuperación.

Se propone crear un método de controlador en la clase anterior que responda a peticiones tipo post en la misma url mencionada anteriormente en el Test 7 y que se encargue de validar los datos de la nueva habitación de recuperación, mostrar los errores encontrados si existieran a través del formulario, y si no existen errores, almacenar la habitación de recuperación a través del servicio de gestión de habitaciones de recuperación. Tras grabar la habitación de recuperación, en caso de éxito, se mostrará la página de inicio de la aplicación (welcome), sin usar redirección. Por tanto, deberá implementar el método “save” del servicio de gestión de habitaciones de recuperación.

Test 9 – Crear una relación entre Visit y RecoveryRoom y otra entre RecoveryRoom y RecoveryRoomType

Se debe crear una relación N a 1 opcional unidireccional desde Visit (alojada en el paquete “org.springframework.samples.petclinic.pet”) hacia RecoveryRoom. Cada Visit tiene asociada una o ninguna RecoveryRoom, la misma RecoveryRoom puede tener asociadas N visitas. Nótese que en la entidad Visit se debe dar una implementación a los métodos getRecoveryRoom y setRecoveryRoom existentes. Por otro lado, se debe crear otra relación N a 1 unidireccional obligatoria desde RecoveryRoom hacia RecoveryRoomType.

Además, hay que modificar el script de inicialización de la base de datos para que alguna visita tenga una habitación de recuperación asociada y que cada habitación de recuperación quede asociada al tipo de habitación de recuperación correspondiente.

Test 10 – Comprobación de unicidad en el nombre de habitaciones del mismo tipo

Modificar el método “save” del servicio de gestión de habitaciones para que si la habitación especificada es de un tipo determinado se impida que exista ya otra habitación del mismo tipo que la especificada con el mismo nombre de habitación. Si se da este caso se debe lanzar una excepción de tipo “*DuplicatedRoomNameException*” (esta clase está ya creada en el paquete *recoveryroom*). Además, en caso de lanzarse la excepción, la transacción asociada a la operación de guardado de la habitación de recuperación debe echarse atrás (hacer rollback).