Grupo: C1.015

Repositorio: https://github.com/jormunrod/Acme-SF-24.1.1

Analisys Report

Isaac Solís Padilla (isasolpad@alum.us.es)

Sevilla, 8-3-2024

Contenido

1. Resumen ejecutivo

En esta entrega se han realizado ajustes críticos en el sistema de gestión para mejorar la integridad y la validación de los datos. Estos cambios incluyen la implementación de restricciones como la verificación de la nulidad, el mapeo adecuado de fechas y horas, la validación de formatos de URL y correos electrónicos, y la restricción de longitudes de texto.

Además, se han establecido relaciones entre diferentes entidades del sistema con validaciones para asegurar conexiones correctas y datos coherentes, abordando aspectos fundamentales como la relación entre módulos de entrenamiento, sesiones y desarrolladores, y ajustando propiedades específicas para cumplir con los requisitos del sistema.

2. Tabla de revisión

Nº Revisión	Fecha	Descripción
1	06-03-2024	Versión inicial
2	08-03-2024	Versión final

3. Introducción

En el desarrollo de sistemas de información, es fundamental asegurar la precisión, coherencia y validez de los datos ingresados y almacenados en la base de datos. Este principio es especialmente crítico en sistemas complejos que gestionan una variedad de entidades interconectadas, como módulos de entrenamiento, sesiones de entrenamiento y perfiles de desarrolladores. Para garantizar la integridad de los datos y el funcionamiento óptimo del sistema, se implementan diversas restricciones y validaciones. Estas medidas abarcan desde la verificación de la nulidad y el formato de los datos hasta la imposición de relaciones lógicas entre las entidades. El objetivo de estas prácticas es prevenir errores, facilitar el mantenimiento del sistema y mejorar la experiencia de los usuarios finales al interactuar con la plataforma.

4. Contenidos

4.1. Training Module

- creationMoment: A parte de la restricción de que esa fecha esté en pasado, he tenido que añadir dos restricciones más:
    - @NotNull: La cual he tenido que añadirla ya que no es una propiedad opcional.
    - @Temporal (TemporalType.TIMESTAMP): La cual he tenido que añadir para poder mapear la propiedad a un tipo fecha y hora en la base de datos.
- difficultyLevel: A parte de crear el enum que nos piden, hemos tenido que añadirle una restricción más:
    - @NotNull: La cual he tenido que añadir ya que no es una propiedad opcional.
- updateMoment: A parte de la restricción de que esa fecha esté en pasado, he tenido que añadir una restricción más:
    - @Temporal (TemporalType.TIMESTAMP): La cual he tenido que añadir para poder mapear la propiedad a un tipo fecha y hora en la base de datos.

  Para esta propiedad tenemos una restricción que nos dice que esta fecha tiene que ser posterior a la fecha de creación, pero esta restricción se creará en el service.

- link: He tenido que añadir dos restricciones:
    - @URL: La cual hemos añadido para verificar que la cadena que metemos en la base de datos tenga el formato de una URL
    - @Length (max = 255): La cual hemos añadido para evitar que se produzca un error de longitud a nivel de base de datos.
- Aunque en el enunciado nos lo dice muy sutilmente, he tenido que añadir una relacion ManyToOne con Project a los que he tenido que añadir las siguientes restricciones:
    - @NotNull: Con esta anotación verificamos que siempre haya una referencia válida a un proyecto asociado, en otras palabras, evitar la excepción *NullPointerException*
    - @Valid: La utilizamos para asegurarnos de que el proyecto al que hacemos referencia sea válido, es decir, que cumpla con todas sus restricciones.
- Tenemos otra relación, con Developer, que aunque esta entidad sea opcional, la relación no, y a la que he tenido que añadir las mismas restricciones que a la relación de proyecto.
    - @NotNull: Con esta anotación verificamos que siempre haya una referencia válida a un developer asociado, en otras palabras, evitar la excepción *NullPointerException*
    - @Valid: La utilizamos para asegurarnos de que el developer al que hacemos referencia sea válido, es decir, que cumpla con todas sus restricciones.

4.2. Training Sesión

- period: Esta propiedad he tenido que dividirla en dos fechas para poder tener un periodo de tiempo, debido a que si solo tenemos una propiedad que se llame *period* no podemos tener un periodo de tiempo. Las dos fechas son las siguiente:
    - startDate: Indica el comienzo del periodo y tiene como restricciones:

        - @NotNull: La cual he tenido que añadir porque no es un atributo opcional.
        - @Temporal (TemporalType.TIMESTAMP): La cual he tenido que añadir para poder mapear la propiedad a un tipo fecha y hora en la base de datos.
    - finishDate: Indica el fin del periodo y tiene como restricciones:
        - @NotNull: La cual he tenido que añadir porque no es un atributo opcional.
        - @Temporal (TemporalType.TIMESTAMP): La cual he tenido que añadir para poder mapear la propiedad a un tipo fecha y hora en la base de datos.

  Esta propiedad no indica una restricción más que se tendría que hacer en el service, lo cual también me ha ayudado a decidir la opción de dividir la propiedad en dos.

- contactEmail: No nos dice ninguna restricción específica, pero he tenido que añadir las siguiente:
    - @NotBlank: La cual hemos tenido que añadir debido a que es obligatorio el email y además no queremos que se permita almacenar una cadena vacía.
    - @Email: La cual he tenido que añadir para verificar que la cadena que se introduce en la base de datos tenga una estructura de email.
    - @Length (max = 255): La cual hemos añadido para evitar que se produzca un error de longitud a nivel de base de datos.
- link: He tenido que añadir dos restricciones:
    - @URL: La cual hemos añadido para verificar que la cadena que metemos en la base de datos tenga el formato de una URL
    - @Length (max = 255): La cual hemos añadido para evitar que se produzca un error de longitud a nivel de base de datos.
- Aunque el enunciado no nos lo dice explícitamente, he tenido que añadir una relacion ManyToOne con Training Module a la que he tenido que añadir las siguientes restricciones:
    - @NotNull: Con esta anotación verificamos que siempre haya una referencia válida a un developer asociado, en otras palabras, evitar la excepción *NullPointerException*
    - @Valid: La utilizamos para asegurarnos de que el developer al que hacemos referencia sea válido, es decir, que cumpla con todas sus restricciones.

4.3. Developer

- email: No nos dice ninguna restricción específica, pero he tenido que añadir las siguiente:
    - @NotBlank: La cual hemos tenido que añadir debido a que es obligatorio el email y además no queremos que se permita almacenar una cadena vacía.
    - @Email: La cual he tenido que añadir para verificar que la cadena que se introduce en la base de datos tenga una estructura de email.
    - @Length (max = 255): La cual hemos añadido para evitar que se produzca un error de longitud a nivel de base de datos.
- link: He tenido que añadir dos restricciones:
    - @URL: La cual hemos añadido para verificar que la cadena que metemos en la base de datos tenga el formato de una URL

- @Length (max = 255): La cual hemos añadido para evitar que se produzca un error de longitud a nivel de base de datos.

-

5. Conclusiones

Este documento detalla las implementaciones adicionales que se han realizado en cada entidad del sistema, como TrainingModule, TrainingSesion y Developer, para garantizar su funcionalidad óptima y cumplir con los requisitos no especificados en el documento Requirement Student.

Se ha seguido un enfoque estructurado para cada entidad, donde se especifican las propiedades adicionales agregadas, junto con las razones detrás de cada adición. Se han aplicado anotaciones como *@NotNull*, *@Temporal*, *@URL*, *@Length*, *@NotBlank*, *@Email*, *@Valid* para garantizar la integridad de los datos y la coherencia del sistema.

En resumen, el documento proporciona una visión detallada de las modificaciones realizadas en cada entidad, evidenciando un enfoque meticuloso para cumplir con los requisitos no explícitos y mejorar la funcionalidad del sistema en su conjunto. La división de las entidades y la descripción de las modificaciones facilitan la comprensión y el seguimiento del proceso de implementación.

6. Bibliografía

Intencionalmente en blanco.