Grupo: C1.015

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

Analysis Report

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

Sevilla, 26-4-2024

Contenido

1. Resumen ejecutivo

En esta entrega se han realizado ajustes críticos en el sistema de gestión para la integridad y validación de datos. Estos cambios incluyen validaciones customizadas, autorizaciones especificas para cada rol y mucho mas para cada entidad y función.

2. Tabla de revisión

Nº Revisión	Fecha	Descripción
1	26-04-2024	Versión inicial.
2	26-04-2024	Verisón final.

3. Introducción

En la reciente actualización del sistema de gestión, hemos implementado ajustes críticos diseñados para fortalecer la integridad y la validación de datos. Estos cambios son esenciales para garantizar que la información manejada sea precisa y confiable. Entre las mejoras más destacadas se incluyen validaciones personalizadas que aseguran la precisión de los datos introducidos en el sistema, adaptándose específicamente a las necesidades de cada tipo de información. Además, se han establecido autorizaciones específicas para cada rol dentro de la organización, lo que permite un control más riguroso sobre quién puede acceder y modificar los datos. Este conjunto de medidas se ha extendido a cada entidad y función dentro del sistema, proporcionando un marco más robusto para la operación segura y eficiente del mismo. Estos ajustes no solo mejoran la seguridad y la funcionalidad del sistema, sino que también reflejan nuestro compromiso continuo con la mejora y la adaptación a las necesidades cambiantes de nuestra organización y sus stakeholders.

4. Contenidos

4.1. Training Module

4.1.1. CreateService

En esta clase, he tenido que verificar en el método “authorised()” que si el desarrollador no tiene ningún training module, pues que puede entrar, y que si tiene algún training module pues hay que comprobar para que tenga acceso que no sea nulo el trainig module (por posible get hacking) y que el rol sea el que corresponde, en este caso “developer”.

En el método “load()”, he cargado el atributo “draftMode”, el desarrollador con el que estamos logueado y además la fecha del momento de creación para que se añada automaticamente.

En el método “bind()”, no he hecho nada significativo.

En el método “validate()”, he hecho dos restricciones, que el códe no sea igual a alguno que ya haya en la base de datos, y que el tiempo total sea mayor que cero.

En el método “perform()”, he puesto “object.setId(0)”, y esto lo he hecho para que no se pueda actualizar un training module cambiando la Id (post hacking).

En el método “unbind()” he hecho un dropdown con el enum “DifficultyLevel()”, y con la entidad “Project”.

4.1.2. DeleteService

En esta clase, he tenido que verificar en el método “authorised()” que si el desarrollador no tiene ningún training module, pues que puede entrar, y que si tiene algún training module pues hay que comprobar para que tenga acceso que no sea nulo el trainig module (por posible get hacking) y que el rol sea el que corresponde, en este caso “developer”.

El método “load()” no he hecho nada significativo.

El método “bind()” no he hecho nada significativo.

El método “validate()” no he hecho nada significativo.

El método “perform()”, hemos tenido que eliminar todos los trainig sessions asociados al trainig module que queremos borrar con “.deleteAll()”.

En el método “unbind()” he hecho un dropdown con el enum “DifficultyLevel()”, y con la entidad “Project”.

4.1.3. ListService

En esta clase, he tenido que verificar en el método “authorised()” que si el desarrollador no tiene ningún training module, pues que puede entrar, y que si tiene algún training module pues hay que comprobar para que tenga acceso que no sea nulo el trainig module (por posible get hacking) y que el rol sea el que corresponde, en este caso “developer”.

El método “load()” no he hecho nada significativo.

El método “unbind()”, he internacionalizado el atributo draftMode, de modo que no aparezca “true”, o “false”, si no que aparezca un tick para decir “true”; y una cross, para el “false”.

4.1.4. PublishService

En esta clase, he tenido que verificar en el método “authorised()” que si el desarrollador no tiene ningún training module, pues que puede entrar, y que si tiene algún training module pues hay que comprobar para que tenga acceso que no sea nulo el trainig module (por posible get hacking) y que el rol sea el que corresponde, en este caso “developer”.

El método “load()” no he hecho nada significativo.

El método “bind()” no he hecho nada significativo.

El método “validate()” he hecho que solo se pueda publicar un training module si tienen trainign sessions y esas training sesions están publicadas.

En el método “perform()”, he puesto el “draftMode” a false, para decir que ya esta publicado ese training module.

En el método “unbind()” he hecho un dropdown con el enum “DifficultyLevel()”, y con la entidad “Project”.

4.1.5. ShowService

En esta clase, he tenido que verificar en el método “authorised()” que si el desarrollador no tiene ningún training module, pues que puede entrar, y que si tiene algún training module pues hay que comprobar para que tenga acceso que no sea nulo el trainig module (por posible get hacking) y que el rol sea el que corresponde, en este caso “developer”.

El método “load()” no he hecho nada significativo.

En el método “unbind()” he hecho un dropdown con el enum “DifficultyLevel()”, y con la entidad “Project”.

4.1.6. UpdateService

En esta clase, he tenido que verificar en el método “authorised()” que si el desarrollador no tiene ningún training module, pues que puede entrar, y que si tiene algún training module pues hay que comprobar para que tenga acceso que no sea nulo el trainig module (por posible get hacking) y que el rol sea el que corresponde, en este caso “developer”.

En el método “load()”, he cargado el atributo “draftMode”, el desarrollador con el que estamos logueado y además la fecha del momento de creación para que se añada automaticamente.

En el método “bind()”, no he hecho nada significativo.

En el método “validate()”, he hecho dos restricciones, que el códe no sea igual a alguno que ya haya en la base de datos, pero que si pueda ser el mismo que tenía, y que el tiempo total sea mayor que cero.

En el método “perform()”, he puesto la fecha de actualizacion automática, es decir, que cuando se actualice, pues se añada la fecha de actualización.

En el método “unbind()” he hecho un dropdown con el enum “DifficultyLevel()”, y con la entidad “Project”.

4.2. Training Session

4.2.1. CreateService

En la función authorise(), se realiza una verificación para determinar si un usuario puede acceder a un módulo de entrenamiento. Se verifica si el usuario asociado con la solicitud tiene el rol de desarrollador asociado con el módulo y si el módulo no es nulo, para evitar posibles intentos de acceso no autorizado o intentos de manipulación mediante solicitudes maliciosas. Si se cumplen estas condiciones, se establece el estado de autorización en verdadero; de lo contrario, se establece en falso, lo que determina si el acceso está permitido o no

En el método “load()”, he cargado el atributo “draftMode”, el desarrollador con el que estamos logueado y además la fecha del momento de creación para que se añada automaticamente.

En el método “bind()”, no he hecho nada significativo.

En el método “validate()”, hemos tenido que comprobar que el “code” no sea igual a ninguno que haya en la base de datos, y además, que el “startDate”, sea una semana posterior al “creationMoment” del training module y por ultimo que la duración entre el “finishDate” y el “startDate” sea al menos de una semana.

En el método “perform()”, he puesto “object.setId(0)”, y esto lo he hecho para que no se pueda actualizar un training module cambiando la Id (post hacking).

El método “unbind()”, no he hecho nada significativo.

4.2.2. DeleteService

En la función authorise(), se realiza una verificación para determinar si un usuario puede acceder a un módulo de entrenamiento. Se verifica si el usuario asociado con la solicitud tiene el rol de desarrollador asociado con el módulo y si el módulo no es nulo, para evitar posibles intentos de acceso no autorizado o intentos de manipulación mediante solicitudes maliciosas. Si se cumplen estas condiciones, se establece el estado de autorización en verdadero; de lo contrario, se establece en falso, lo que determina si el acceso está permitido o no

En el método “load()”, no he hecho nada significativo.

En el método “bind()” no he hecho nada significativo.

En el método “validate()” no he hecho nada significativo.

En el método “perform()”, no he hecho nada significativo.

En el método “unbind()”, no he hecho nada significativo.

4.2.3. ListService

En la función authorise(), se realiza una verificación para determinar si un usuario puede acceder a un módulo de entrenamiento. Se verifica si el usuario asociado con la solicitud

tiene el rol de desarrollador asociado con el módulo y si el módulo no es nulo, para evitar posibles intentos de acceso no autorizado o intentos de manipulación mediante solicitudes maliciosas. Si se cumplen estas condiciones, se establece el estado de autorización en verdadero; de lo contrario, se establece en falso, lo que determina si el acceso está permitido o no

En el método “load()”, no he hecho nada significativo.

El método “unbind()”, he internacionalizado el atributo draftMode, de modo que no aparezca “true”, o “false”, si no que aparezca un tick para decir “true”; y una cross, para el “false”.

Hay un segundo método “unbind()”, lo he hecho para poder globalizar la id del training module para poder crear un training session en el training modulo que estemos.

4.2.4. PublishService

En la función authorise(), se realiza una verificación para determinar si un usuario puede acceder a un módulo de entrenamiento. Se verifica si el usuario asociado con la solicitud tiene el rol de desarrollador asociado con el módulo y si el módulo no es nulo, para evitar posibles intentos de acceso no autorizado o intentos de manipulación mediante solicitudes maliciosas. Si se cumplen estas condiciones, se establece el estado de autorización en verdadero; de lo contrario, se establece en falso, lo que determina si el acceso está permitido o no

En el método “load()”, he cargado el atributo “draftMode”, el desarrollador con el que estamos logueado y además la fecha del momento de creación para que se añada automaticamente.

En el método “bind()”, no he hecho nada significativo.

En el método “validate()”, hemos tenido que comprobar que el “code” no sea igual a ninguno que haya en la base de datos, y además, que el “startDate”, sea una semana posterior al “creationMoment” del training module y por ultimo que la duración entre el “finishDate” y el “startDate” sea al menos de una semana.

En el método “perform()”, he puesto “object.setId(0)”, y esto lo he hecho para que no se pueda actualizar un training module cambiando la Id (post hacking).

El método “unbind()”, no he hecho nada significativo.

4.2.5. ShowService

En la función authorise(), se realiza una verificación para determinar si un usuario puede acceder a un módulo de entrenamiento. Se verifica si el usuario asociado con la solicitud tiene el rol de desarrollador asociado con el módulo y si el módulo no es nulo, para evitar posibles intentos de acceso no autorizado o intentos de manipulación mediante solicitudes maliciosas. Si se cumplen estas condiciones, se establece el estado de autorización en verdadero; de lo contrario, se establece en falso, lo que determina si el acceso está permitido o no.

En el método “load()”, no he hecho nada significativo.

En el método “unbind()”, no he hecho nada significativo.

4.2.6. UpdateService

En la función authorise(), se realiza una verificación para determinar si un usuario puede acceder a un módulo de entrenamiento. Se verifica si el usuario asociado con la solicitud tiene el rol de desarrollador asociado con el módulo y si el módulo no es nulo, para evitar posibles intentos de acceso no autorizado o intentos de manipulación mediante solicitudes maliciosas. Si se cumplen estas condiciones, se establece el estado de autorización en verdadero; de lo contrario, se establece en falso, lo que determina si el acceso está permitido o no

En el método “load()”, no he hecho nada significativo.

En el método “bind()”, no he hecho nada significativo.

En el método “validate()”, hemos tenido que comprobar que el “code” no sea igual a ninguno que haya en la base de datos, y además, que el “startDate”, sea una semana posterior al “creationMoment” del training module y por ultimo que la duración entre el “finishDate” y el “startDate” sea al menos de una semana.

En el método “perform()”, no he hecho nada significativo.

El método “unbind()”, no he hecho nada significativo.

5. Conclusiones

La documentación detallada de las distintas clases y métodos del sistema de gestión de módulos de entrenamiento refleja un enfoque cuidadoso en la seguridad y la integridad de los datos. Se han implementado medidas específicas para verificar autorizaciones y evitar accesos no autorizados, como la verificación del rol de desarrollador y la validación contra nulidad de los módulos en los métodos “authorised()”. Además, se han establecido controles para garantizar la unicidad de los códigos de módulos y la lógica de fechas en los métodos “validate()”. Aunque algunas funciones no registran cambios significativos, se destacan las precauciones contra la manipulación de identificadores y la internacionalización de ciertos atributos. Estas prácticas subrayan el compromiso del sistema para mantener un entorno seguro y eficiente en la gestión de módulos de entrenamiento.

6. Bibliografía

Intencionalmente en blanco.