INFORME DE ANÁLISIS D02

Grupo: C2.026

- Ignacio Blanquero Blanco (ignblabla@alum.us.es)
- Adrián Cabello Martín (adrcabmar@alum.us.es)
- María de la Salud Carrera Talaverón (marcartal1@alum.us.es)
- Joaquín González Ganfornina (joagongan@alum.us.es)
- Natalia Olmo Villegas (natolmvil@alum.us.es)

Repositorio: https://github.com/maryycarrera/Acme-SF-D04

Fecha: 22 de junio de 2024

Tabla de contenido

https://github.com/maryycarrera/Acme-SF-D04

https://github.com/maryycarrera/Acme-SF-D04

Resumen ejecutivo

El propósito de este informe es llevar a cabo un análisis exhaustivo de todos los requisitos obligatorios especificados en la segunda entrega del proyecto Acme-SF-D02 de la asignatura Diseño y Pruebas II. Este proyecto tiene como objetivo mejorar las competencias en el desarrollo web, enfocándose en la implementación de un sistema de información de tamaño moderado. El informe detalla las decisiones adoptadas para la implementación de estos requisitos y su correlación con las validaciones y observaciones realizadas por el profesor a cargo de la asignatura.

Historial de versiones

Versión	Fecha	Descripción
v1.0	01/03/2024	Estructura inicial Informe de Análisis D02.
v1.1	05/03/2024	Actualización Informe de Análisis D02.
V1.2	07/03/2024	Actualización Informe de Análisis D02.
v1.3	08/03/2024	Revisión y finalización del Informe de Análisis D02.
v2.0	21/06/2024	Realización de mejoras en el análisis correspondiente a las tareas obligatorias II, III, IV y V para la segunda convocatoria.
v2.1	22/06/2024	Realización de mejoras en el análisis correspondiente a las tareas suplementarias XIII y XIV para la segunda convocatoria. Realización de mejoras en la conclusión del documento para la segunda convocatoria. Finalización del documento para la segunda convocatoria.

Introducción

El presente documento, titulado “Informe de Análisis D02”, constituye un análisis exhaustivo de los requisitos individuales asignados en el proyecto del grupo C2.026 al estudiante Ignacio Blanquero Blanco, en el marco del proyecto Acme-SF-D02. Este proyecto forma parte de la asignatura de Diseño y Pruebas II, de naturaleza educativa, y está orientado a que los estudiantes desarrollemos las competencias en el ámbito del desarrollo web.

El propósito principal de este informe es examinar y resolver los siguientes requisitos obligatorios del proyecto. En primer lugar, un módulo de entrenamiento, que consiste en una o varias actividades de corta duración destinadas a ampliar o actualizar conocimientos y habilidades relacionadas con el tema de un proyecto, debe almacenar datos específicos, tales como un código único, el momento de la creación, detalles descriptivos, nivel de dificultad, momento de actualización, enlace y tiempo estimado. Además, cada módulo de entrenamiento se compone de sesiones de entrenamiento que requieren el almacenamiento de datos como un código único, período de tiempo, ubicación, instructor, correo electrónico y enlace. Asimismo, el sistema debe gestionar tableros de control para desarrolladores con datos sobre módulos y sesiones de entrenamiento, y generar datos de muestra, incluyendo dos cuentas de desarrollador con credenciales específicas. Este informe detalla los pasos necesarios para identificar, implementar y validar estas modificaciones, los cuales se abordarán en las siguientes secciones.

El contenido del documento se estructura de la siguiente manera: **Resumen ejecutivo**, el cual proporciona una visión general del proyecto y sus objetivos; **Historial de versiones**, que incluye un registro detallado de los cambios y versiones del documento; Introducción, que ofrece una descripción general del contenido y la estructura del documento; **Informe de Análisis – Acme SF-D02**, que detalla el análisis realizado para cumplir con los requisitos obligatorios, así como sus soluciones, los problemas encontrados y la validación por parte del profesor; **Conclusiones**, que presenta reflexiones finales sobre el entregable y los aprendizajes obtenidos; y **Bibliografía**, que enumera las referencias utilizadas para el desarrollo del proyecto.

Informe de análisis – Acme-SF-D02

Requisito obligatorio II:

A **training module** consists of one or several short-term training activities aimed at extending or updating knowledge and skills related to the topic of a **project**. The system must store the following data about them: a **code** (pattern "[A-Z]{1,3}-[0-9]{3}", not blank, unique), a **creation moment** (in the past), some **details** describing the training module (not blank, shorter than 101 characters), a **difficulty level** ("Basic", "Intermediate", or "Advanced"), an optional **update moment** (in the past, after the creation moment), an **optional link** with further information, and an estimated **total time**.

Análisis y decisiones

Para cumplir con el requisito, se han considerado las restricciones relacionadas con las anotaciones vistas en las clases de teoría. Este análisis resultó fundamental para entender las limitaciones y los conceptos generales necesarios para realizar una implementación correcta y para posteriormente completar la tarea extra individual de UML.

La entidad **"TrainingModule"** se añadió en la ruta "src/main/java/acme/entities/trainingmodules", diferenciándose de otras entidades, las cuales se encuentran en diferentes rutas. Tras un análisis detallado de dicha entidad, se decidió aplicar las siguientes restricciones: el atributo **"code"**, de tipo String, se anotó con @NotBlank y @Pattern(regexp = "[A-Z]{1,3}-[0-9]{3}") para asegurar un formato específico que no sea nulo ni esté vacío, y @Column(unique = true) para garantizar su unicidad. El atributo **"creationMoment"**, de tipo Date, se anotó con @Temporal(TemporalType.TIMESTAMP), @Past y @NotNull para asegurar que el momento de creación sea una fecha pasada y no nula. El atributo **"details"**, de tipo String, se limitó a un máximo de 100 caracteres utilizando @Lenght(max = 100) y se anotó con @NotBlank para que no esté vacío. El nivel de dificultad **("difficultyLevel")** se implementó como un enumerado "DifficultyLevel" cuyos valores pueden ser BASIC, INTERMEDIATE, ADVANCED y no nulos (@NotNull). El atributo **"updateMoment"** también se anotó con @Temporal(TemporalType.TIMESTAMP) Y @Past para asegurar que sea una fecha pasada. El atributo **"link"** se anotó con @URL para garantizar que el enlace sea válido. Para el tiempo total estimado **("estimatedTotalTime")**, se utilizó un tipo primitivo "int" con las anotaciones @NotNull y @Min(1) para asegurar que sea un valor positivo y no nulo. Por último, el atributo **"draftMode"**, de tipo booleano, se anotó con @NotNull para asegurar que no sea nulo.

En cuanto a las relaciones, se implementaron dos relaciones ManyToOne unidireccionales desde "TrainingModule" hacia "Project" y hacia "Developer", ambos anotados con @NotNull, @Valid y @ManyToOne(optional = false) para garantizar que dichas relaciones sean válidas y obligatorias.

Solución

Para abordar este requisito, se ha implementado todo lo mencionado anteriormente en la clase Java denominada "TrainingModule", a la cual se puede acceder mediante el siguiente enlace: https://github.com/maryycarrera/Acme-SF-D04/tree/master/src/main/java/acme/entities/trainingmodules

Una vez implementadas estas modificaciones, el requisito puede considerarse cumplido de acuerdo con las especificaciones detalladas en el enunciado.

Problemas encontrados

Inicialmente, no se consideró la asociación de la entidad “TrainingModule” con el rol de “Developer”, por lo que fue necesario agregarla posteriormente. Además, se introdujo el atributo “draftMode” para distinguir claramente entre contenido en modo borrador y contenido publicado, dado que nuestro sistema maneja información que requiere aprobación antes de su publicación definitiva.

Además, al definir el atributo “estimatedTotalTime”, se optó inicialmente por utilizar el tipo Integer. Sin embargo, tras discutirlo con los compañeros del grupo, se decidió cambiarlo a un tipo primitivo int. Esta elección se basa en que los tipos primitivos no admiten valores nulos, garantizando que el atributo “estimatedTotalTime” siempre contendrá un valor válido.

Validación del profesor

En la primera sesión correspondiente al segundo entregable, el profesor señaló que el requisito estaba incompleto debido a la ausencia de la relación entre la entidad “TrainingModule” y el rol “Developer”. Para la segunda sesión, esta deficiencia fue corregida, lo que permitió al profesor validar adecuadamente el funcionamiento del requisito durante la clase de laboratorio.

Requisito obligatorio III:

Each **training module** is made up of **training sessions**. The system must store the following data about them: a **code** (pattern "TS-[A-Z]{1,3}-[0-9]{3}", not blank, unique), a time **period** (at least one week ahead the training module creation moment, at least one week long), a **location** (not blank, shorter than 76 characters), an **instructor** (not blank, shorter than 76 characters), a mandatory **contact email**, and an **optional link** with further information.

Análisis y decisiones

Para cumplir con el requisito, se han considerado las restricciones relacionadas con las anotaciones vistas en las clases de teoría. Este análisis resultó fundamental para entender las limitaciones y los conceptos generales necesarios para realizar una implementación correcta y para posteriormente completar la tarea extra individual de UML.

La entidad **"TrainingSession"** se ha integrado en la ruta "src/main/java/acme/entities/trainingsessions", distinguiéndose por su ubicación en comparación con otras entidades que se encuentran en diferentes rutas del proyecto. Después de un análisis exhaustivo de esta entidad, se han aplicado las siguientes restricciones: el atributo **"code"**, de tipo String, ha sido anotado con @NotBlank, @Column(unique = true) y @Pattern(regexp = "TS-[A-Z]{1,3}-[0-9]{3}, asegurando así que siga un formato específico sin ser nulo ni vacío, y garantizando su unicidad en la base de datos. Los atributos **"startPeriodDate"** y **"finishPeriodDate"**, ambos de tipo Date, han sido anotados con @NotNull y @Temporal(TemporalType.TIMESTAMP), asegurando que las fechas de inicio y fin del período sean obligatorias y del tipo adecuado de dato temporal. La localización **("location")** y el instructor **("instructor")**, ambos de tipo String, se han limitado a un máximo de 75 caracteres utilizando @NotBlank y @Lenght(max = 75), garantizando que no estén vacío y que cumplan con la longitud máxima permitida. El atributo **"contactEmail"**, de tipo String, se ha validado como dirección de correo electrónico utilizando @NotBlank y @Email, asegurando que el formato de correo electrónico sea válido y no esté vacío. El atributo **"link"**, también de tipo String, se ha anotado con @URL para garantizar que el enlace proporcionado sea válido y cumpla con los estándares de URL. Por último, el atributo **"draftMode"**, de tipo booleano, se anotó con @NotNull para asegurar que no sea nulo.

En cuanto a las relaciones, se ha implementado una relación ManyToOne unidireccional desde "TrainingSession" hacia "TrainingModule", anotada con @NotNull, @Valid y @ManyToOne(optional = false), asegurando que la relación sea válida y obligatoria.

Solución

Para abordar este requisito, se ha implementado todo lo mencionado anteriormente en la clase Java denominada "TrainingSession", a la cual se puede acceder mediante el siguiente enlace: https://github.com/maryycarrera/Acme-SF-D04/blob/master/src/main/java/acme/entities/trainingsessions/TrainingSession.java

Una vez implementadas estas modificaciones, el requisito puede considerarse cumplido de acuerdo con las especificaciones detalladas en el enunciado.

Problemas encontrados

Durante la fase de desarrollo de este requisito, encontré dificultades en cuanto a la implementación del atributo "period". Inicialmente consideré utilizar el tipo Period debido a su

nombre sugestivo. Sin embargo, al revisar las diapositivas de teoría y no encontrar ninguna referencia a este tipo, decidí consultar el foro de la asignatura en el que otro estudiante ya había planteado la misma duda. Finalmente, logré entender que lo correcto para la implementación de este atributo era crear dos atributos separados, uno para el inicio y otro para el fin de la sesión de entrenamiento.

Validación del profesor

El profesor validó correctamente el funcionamiento del requisito en clase de laboratorio.

https://github.com/maryycarrera/Acme-SF-D04

Requisito obligatorio IV:

The system must handle **developer** dashboards with the following data: total number of **training modules** with an **update moment**; total number of **training sessions** with a **link**; average, deviation, minimum, and maximum **time** of the **training modules**.

Análisis y decisiones

Para cumplir con el requisito, se han considerado las restricciones relacionadas con las anotaciones vistas en las clases de teoría. Este análisis resultó fundamental para entender las limitaciones y los conceptos generales necesarios para realizar una implementación correcta y para posteriormente completar la tarea extra individual de UML.

La clase "DeveloperDashboard" se ha incorporado en la ruta "src/main/java/acme/forms/DeveloperDashboard", lo cual la diferencia de otras clases dentro del proyecto. Tras un análisis exhaustivo, se han establecido las siguientes especificaciones con el objetivo de cumplir con los requisitos detallados. Los atributos "totalNumberTrainingModulesWithUpdateMoment", encargado de contabilizar el número total de módulos de entrenamiento con momento de actualización; "totalNumberTrainingSessionsWithLink", que registra el total de sesiones de entrenamiento con enlace disponible; "minimumTimeTrainingModules", que indica el tiempo mínimo de los módulos de entrenamiento; y "maximumTimeTrainingModules", que indica el tiempo máximo de los módulos de entrenamiento, han sido implementados como tipos primitivos int. En contraste, los atributos "averageTimeTrainingModules", que calcula el tiempo promedio de los módulos de entrenamiento, y "deviationTimeTrainingModules", que determina la desviación de los tiempos de los módulos de entrenamiento, han sido definidos como tipos primitivos double.

Solución

Para abordar este requisito, se ha implementado todo lo mencionado anteriormente en la clase denominada "DeveloperDashboard", a la cual se puede acceder mediante el siguiente enlace: https://github.com/maryycarrera/Acme-SF-D04/blob/master/src/main/java/acme/forms/DeveloperDashboard.java

Con estas modificaciones implementadas, el requisito se considera como completado conforme a las especificaciones detalladas en el enunciado.

Problemas encontrados

Inicialmente, se optó por definir los atributos utilizando los tipos Double e Integer. Sin embargo, después de recibir una revisión por parte del profesor de laboratorio, se indicó que esta elección no era la más adecuada. En consecuencia, varios miembros del grupo que enfrentaban el mismo problema decidieron que lo más adecuado sería implementar estos atributos con los tipos primitivos int y double. Esta decisión se tomó con el fin de evitar posibles complicaciones en el Dashboard en caso de ausencia de datos, dado que los tipos primitivos no admiten valores nulos. Por lo tanto, estos tipos de datos siempre contienen un valor por defecto (por ejemplo, 0 para int y 0.0 para double).

Validación del profesor

Basándose en las observaciones realizadas en el apartado anterior y las correcciones realizadas, el profesor validó correctamente el requisito.

https://github.com/maryycarrera/Acme-SF-D04

Requisito obligatorio V:

Produce assorted sample data to test your application informally. The data must include two **developer** accounts with credentials "**developer1/developer1**" and "**developer2/developer2**".

Análisis y decisiones

Para cumplir con el requisito, se han considerado las restricciones relacionadas con las anotaciones vistas en las clases de teoría. Este análisis resultó fundamental para entender las limitaciones y los conceptos generales necesarios para realizar una implementación correcta y para posteriormente completar la tarea extra individual de UML.

Los archivos CSV utilizados, nombrados como training-module.csv, training-session.csv y developer.csv, están ubicados en el directorio "src/main/webapp/WEB-INF/resources/sample-data".

Para los atributos de tipo String, la base de datos se ha poblado de la siguiente manera: se ha probado con cadenas de un solo carácter, de dos caracteres, un número intermedio de caracteres en función de la longitud de esta, el número máximo de caracteres menos uno y el número máximo de caracteres permitidos, en función de la restricción de esta.

Para el atributo de momento de creación, se han evaluado diversos valores, poniendo especial énfasis en los momentos cercanos al momento base del proyecto, que es el 30 de julio de 2022 a las 00:00. Se ha probado el valor mencionado, así como otros momentos cercanos a este (por ejemplo, el 29 de julio de 2022 a las 23:59), además de otras fechas adicionales.

Para los atributos que hacen referencia a URL, se han considerado diversos valores basados en la longitud de los enlaces, garantizando siempre que cumplan con la estructura adecuada. Se han probado enlaces de longitud corta (siete y ocho caracteres), así como enlaces de longitud intermedia y aquellos que alcanzan los límites máximos proporcionados en los datos de muestra dados por los profesores de la asignatura, es decir, 254 y 255 caracteres.

Para los atributos de correo electrónico, se han considerado direcciones con un formato válido y una longitud que oscila entre 6 y 254 caracteres, valores establecidos en los datos de muestra mencionados anteriormente, proporcionados por los profesores de la asignatura.

Respecto a los atributos de la fecha de inicio y fin del período, se han considerado valores que cumplan con las restricciones establecidas en el Requisito obligatorio III (trainingSessions). Es decir, la fecha de inicio debe ser una semana posterior a la creación del módulo de entrenamiento, y la duración del período debe ser de al menos una semana. Se han probado valores específicos como que la fecha de inicio sea exactamente siete días después de la creación del módulo y que la fecha de fin del período sea también siete días posterior a la fecha de inicio. Además, se han evaluado valores cercanos a estos casos y otros valores adicionales.

Por último, en cuanto a los valores de tipo entero, se han realizado una serie de pruebas utilizando el valor mínimo aceptable, que es uno, así como valores cercanos a este. También se han considerado valores intermedios de dos y tres cifras, y valores numéricos elevados, de cuatro y cinco cifras.

A la hora de incluir las dos cuentas del rol "Developer" con las credenciales "developer1/developer1" y "developer2/developer2", se han utilizado datos de ejemplo que son coherentes con el contexto del proyecto.

https://github.com/maryycarrera/Acme-SF-D04

Solución

Para cumplir con este requisito, se han completado los archivos CSV correspondientes a las entidades “TrainingModule”, “TrainingSession” y “Developer”, a los cuales se puede acceder mediante el siguiente enlace: https://github.com/maryycarrera/Acme-SF-D04/tree/master/src/main/webapp/WEB-INF/resources/sample-data

Con estas modificaciones implementadas, el requisito se considera como completado conforme a las especificaciones detalladas en el enunciado.

Problemas encontrados

Inicialmente, se pobló la base de datos sin considerar las indicaciones proporcionadas en las transparencias de clase, es decir, no se tuvieron en cuenta los límites de los atributos al realizar las pruebas. Tras consultar con varios compañeros del grupo, se decidió repoblar la base de datos siguiendo las directrices de las transparencias de clase y utilizando los datos de prueba proporcionados por los profesores de la asignatura en el archivo “Sample-Data”, disponible en la carpeta Scrapbook del Workspace.

Validación del profesor

El requisito mencionado no pudo ser revisado durante la clase de laboratorio debido a la falta de tiempo.

https://github.com/maryycarrera/Acme-SF-D04

Requisito suplementario XIII:

There is a new project-specific role called **developer**, which has the following profile data: **degree** (not blank, shorter than 76 characters), a **specialisation** (not blank, shorter than 101 characters), list of **skills** (not blank, shorter than 101 characters), an **email**, and an **optional link** with further information.

Análisis y decisiones

Para cumplir con el requisito, se han considerado las restricciones relacionadas con las anotaciones vistas en las clases de teoría. Este análisis resultó fundamental para entender las limitaciones y los conceptos generales necesarios para realizar una implementación correcta y para posteriormente completar la tarea extra individual de UML.

La entidad “Developer” se ha integrado en la ruta “src/main/java/acme/roles/developer”, distinguiéndose por su ubicación en comparación con otras entidades que se encuentran en diferentes rutas del proyecto. Después de un análisis exhaustivo de esta entidad, se han aplicado las siguientes restricciones: el atributo “degree”, de tipo String, ha sido anotado con @NotBlank y @Lenght(max = 75), asegurando así que no esté vacío y que cumpla con la longitud máxima permitida. Los atributos “specialisation” y “skills”, también de tipo String, han sido anotados con @NotBlank y @Lenght(max = 100), garantizando que no estén vacíos y que cumplan con la longitud máxima establecida. El atributo “email”, de tipo String, se ha validado como dirección de correo electrónico utilizando @NotBlank y @Email, asegurando que el formato de correo electrónico sea válido y que no esté vacío. El atributo “link”, también de tipo String, se ha anotado con @URL para garantizar que el enlace proporcionado sea válido y cumpla con los estándares de URL.

En lo que respecta a las relaciones, esta clase no mantiene ninguna conexión con otras clases.

Solución

Para abordar este requisito, se ha implementado todo lo mencionado anteriormente en la clase denominada “Developer”, a la cual se puede acceder mediante el siguiente enlace: https://github.com/maryycarrera/Acme-SF-D04/blob/master/src/main/java/acme/roles/Developer.java

Con estas modificaciones implementadas, el requisito se considera como completado conforme a las especificaciones detalladas en el enunciado.

Problemas encontrados

Intencionalmente en blanco.

Validación del profesor

El requisito mencionado no pudo ser revisado durante la clase de laboratorio debido a la falta de tiempo.

https://github.com/maryycarrera/Acme-SF-D04

Requisito suplementario XIV:

Produce a UML domain model.

Análisis y decisiones

Después de cumplir con los requisitos previos, se procedió a la creación del diagrama UML, el cual representa las relaciones entre las entidades “Project”, “TrainingModule”, “TrainingSession” y “Developer”, de acuerdo con las estructuras y cardinalidades descritas a continuación.

Se ha establecido una relación “OneToMany” opcional entre la entidad “Project” y la entidad “TrainingModule”. Esta relación indica que un proyecto puede estar asociado opcionalmente con uno o varios módulos de entrenamiento, es decir, un proyecto puede tener cero o múltiples módulos de entrenamiento asociados, mientras que cada “TrainingModule” está relacionado exactamente con un solo proyecto.

En cuanto a la relación entre la entidad “TrainingModule” y la entidad “TrainingSession”, se ha decidió que sea una composición. Esto se debe a que, al eliminar un módulo de entrenamiento, también deben eliminarse las sesiones de entrenamiento que contiene; no tiene sentido que dichas sesiones (componentes) sigan existiendo si el módulo de entrenamiento (contenedor) no existe. En términos de cardinalidad, un módulo de entrenamiento puede tener una o múltiples sesiones de entrenamiento asociadas, mientras que cada “TrainingSession” está relacionada con exactamente un “TrainingModule”.

Para relacionar la entidad “Developer” con la entidad “TrainingModule”, se ha optado por una relación “OneToMany” opcional, que índice que un desarrollador puede crear ninguno, uno o múltiples módulos de entrenamiento, mientras que un módulo de entrenamiento puede ser creado únicamente por un único desarrollador.

En el diagrama también se puede apreciar el formulario “DeveloperDashboard”, el cual incluye operaciones relacionadas con los módulos y sesiones de entrenamiento. Además, se encuentra el tipo enumerado “DifficultyLevel”, el cual representa un atributo de la entidad “TrainingModule” con valores posibles de BASIC, INTERMEDIATE y ADVANCED. Asimismo, se han definido el tipo de datos y diversas restricciones para cada atributo de las entidades. Para abordar las restricciones más complejas de los requisitos, se han utilizado notas descriptivas con el fin de proporcionar una explicación detallada sobre estas.

Solución

Para abordar este requisito, se ha producido el diagrama UML, al cual se puede acceder mediante el siguiente enlace: https://github.com/maryycarrera/Acme-SF-D04/blob/master/reports/Student%203/D02/UML-Diagram-Student3-(ignblabla).png

Con estas modificaciones implementadas, el requisito se considera como completado conforme a las especificaciones detalladas en el enunciado.

Problemas encontrados

Cuando establecí la relación entre la entidad “TrainingModule” y “Developer”, tuve dudas sobre si debería ser una relación “ManyToOne” opcional u obligatoria. Después de pensarlo detenidamente, decidí implementarla como opcional, puesto que no había ningún requisito del sistema que especificara que un desarrollador debía crear al menos un módulo de entrenamiento.

https://github.com/maryycarrera/Acme-SF-D04

Para la relación de “TrainingModule” con “Project” enfrenté una situación similar y decidí adoptar la misma solución que antes: implementarla como una relación “ManyToOne” opcional.

Validación del profesor

El requisito mencionado no pudo ser revisado durante la clase de laboratorio debido a la falta de tiempo.

https://github.com/maryycarrera/Acme-SF-D04

Conclusiones

En el análisis del proyecto Acme-SF-D02, se ha logrado abordar y resolver diversos requisitos obligatorios y suplementarios mediante la implementación de varias entidades y relaciones en el sistema de información. En primer lugar, se cumplió con la correcta implementación de los módulos y las sesiones de entrenamiento, incluyendo sus relaciones y atributos específicos. Este avance asegura que el sistema pueda gestionar de manera eficiente las actividades de formación y los datos asociados. Además, se poblaron las distintas entidades de la base de datos y se calcularon y mostraron métricas clave como el número total de módulos de entrenamiento con momento de actualización, el número de total de sesiones de entrenamiento con enlace, así como estadísticas de tiempo relacionadas con los módulos.

Durante el desarrollo de los requisitos, se identificaron y resolvieron varios problemas, como la correcta implementación de atributos y la configuración de relaciones entre entidades. Las consultas en foros y la revisión de materiales teóricos fueron cruciales para superar estos obstáculos. Esto ha permitido desarrollar habilidades en implementación de entidades y relaciones, así como manejo de anotaciones y restricciones. Además, la realización del diagrama UML y su correspondiente análisis ha ayudado a fortalecer la comprensión del modelado del sistema y la relación entre las distintas entidades.

Las validaciones realizadas por el profesor en las sesiones de laboratorio confirmaron el correcto funcionamiento de las implementaciones, aunque algunos requisitos no pudieron ser revisados debido a limitaciones de tiempo.

En conclusión, el informe demuestra un enfoque meticuloso y efectivo en la implementación de los requisitos del proyecto Acme-SF-D02, proporcionando una base sólida para futuros desarrollos y mejoras en el sistema. La experiencia adquirida y las soluciones implementadas contribuyen significativamente al objetivo educativo de la asignatura.

Bibliografía

- Documentación de la asignatura (08 Annexes).
- Excel proporcionado por los profesores de la asignatura, titulado “Sample-Data”.
- Foro de la asignatura: https://ev.us.es/ultra/courses/_85092_1/cl/outline.