Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas 2

Curso 2023 – 2024

Reporte de análisis estudiante 4

Josué Rodríguez López

josrodlop19@alum.us.es

Repositorio: https://github.com/DP2-C1-013/Acme-SF-D01-24.1.0

Fecha	Versión
7/3/2024	v2.0

1. Resumen

Este reporte tiene como objetivo realizar un listado del análisis de los distintos requisitos individuales asignados al estudiante #4 para el proyecto de la signatura. Para cada requisito se dará información sobre las conclusiones a las que se ha llegado tras el análisis y de las decisiones tomadas.

2. Control de versiones

Nº de revisión	Fecha	Descripción
1.0	14/02/2024	Desarrollo inicial del reporte de análisis
2.0	07/03/2024	

3. Introducción

El reporte de análisis está centrado en el análisis de los requisitos asignados al estudiante #4. En esta versión del documento corresponde al entregable 2 del proyecto, pero siguen apareciendo los análisis de entregables anteriores. Se analizarán los requisitos más complejos, centrándose principalmente en aquellos elementos que hayan requerido la valoración de distintas opciones. Los requisitos triviales no serán tratados o se hará de forma muy superficial si hicieran falta para el desarrollo del análisis de otro más complejo. Todos los requisitos han sido revisados en las sesiones de Follow Up.

El contenido del reporte de análisis está estructurado en los siguientes puntos:

1. Resumen ejecutivo: resumen muy general del contenido principal del reporte.
2. Control de versiones del documento.
3. Introducción: a diferencia del resumen ejecutivo, se realiza un resumen más detallado del contenido principal al mismo y se explica la estructura del documento.
4. Contenidos: punto principal del reporte en el que se entra con todo detalle en el análisis de los distintos requisitos.
5. Conclusión.
6. Bibliografía.

4. Contenidos

4.1 Entregable 1

En el entregable 1 no ha sido necesario el análisis de ninguno de los requisitos debido a la trivialidad de los mismos.

4.2. Entregable 2

4.2.1 Entidad Sponsorship

The sponsorships are related to a project with the aim of achieving greater visibility in the market. The system must store the following data about them: a code (pattern "[A-Z]{1,3}-[0-9]{3}", not blank, unique), a moment (in the past), a duration (after the moment, at least one month long), an amount (positive), a type of sponsorship ("Financial", "In kind"), an optional contact email, and an optional link with further information.

Este requisito conlleva la creación de una entidad Sponsorship que guarda información sobre el patrocinio de los proyectos. Por ello, esta entidad debe estar relacionada con la entidad Project. Sin embargo, no queda claro el tipo de relación que se debe implementar. A continuación se presentan las opciones más viables barajadas:

1. **Relación simple Many to One con multiplicidad 0..*:** esta opción conlleva que la relación admite que un proyecto no tenga ninguna entidad Sponsorship asociada.
2. **Agregación con multiplicidad 0..*:** caso similar al 1 pero se entiende la entidad Sponsorship como una parte de Project.
3. **Relación simple Many to One con multiplicidad 1..*:** esta opción conlleva que la relación admite que un proyecto tenga por lo menos una entidad Sponsorship asociada.
4. **Agregación con multiplicidad 1..*:** caso similar al 3 pero se entiende la entidad Sponsorship como una parte obligatoria de Project.

Se ha optado por la opción 1. Los motivos son que se asume que un proyecto puede no tener asociado ninguna entidad Sponsorship si el proyecto no se patrocina, por eso la multiplicidad 0..* opcional. Además, se entiende el patrocinio como un elemento externo que se relaciona con el proyecto y no como una parte del mismo, por lo que se utiliza una relación simple.

A continuación se muestra el análisis de las restricciones de los atributos de la entidad. Como nota, se considerará que un atributo es obligatorio a menos que se indique explícitamente lo contrario:

1. **code:** código que identifica una instancia de la entidad Sponsorship, de tipo String. Este atributo no puede ser una cadena vacía (NotBlank), debe ser único (Unique) y debe seguir el formato especificado por la expresión regular *[A-Z]{1,3}-[0-9]{3}*. Un ejemplo de una cadena que cumple el patrón de la expresión es ABC-123.
2. **moment** y **duration:** de tipo Date, este par de atributos representa el inicio (moment) y final (duration) del patrocinio. Ambos deben ser obligatorios (NotNull) y en el caso de moment, este debe ser un momento pasado (Past). En cuanto a que

datos guardar en la base de datos para manejar estos atributos se han barajado dos opciones:

a. **Fecha y hora (Temporal(TIMESTAMP)**: permite una definición del periodo más detallada, pudiendo especificar el momento exacto de inicio y fin. Ocupa más espacio en memoria y es ligeramente más complejo de manejar.
b. **Fecha Temporal(Date))**: es ligeramente más simple de manejar y ocupa menos espacio en memoria. Sin embargo, limita más el nivel de definición que se puede alcanzar.

La opción elegida ha sido la a, fecha y hora. El motivo es que se valora más la definición del intervalo de duración del patrocinio que la eficiencia en memoria o una insignificante mejora del manejo.

Estos dos momentos deben estar separados por al menos un mes, siendo moment el primero de los dos cronológicamente. Puesto que esta restricción relaciona dos atributos, será implementada sobre el servicio.

3. **type**: indica el tipo del patrocinio. Este puede tomar dos valores: *In_kind* y *Financial*. Debido a que hay una lista de valores posibles se implementa como un enumerado.
4. **amount**: representa el presupuesto asignado al patrocinio. Atributo obligatorio (NotNull). Se han barajado tres posibles tipos para este atributo:
    a. **Integer**: simple de implementar pero imposibilita poner valores decimales.
    b. **Double**: simple de implementar y permite valores decimales.
    c. **Money**: tipo de dato personalizado que incluye el framework que permite poner cantidades decimales y especificar el tipo de moneda usada.

    Se ha optado por la opción 3 puesto que en el contexto del proyecto es necesario poder diferenciar los distintos tipos de divisa. Además, también se cree necesario poder manejar cantidades decimales de dinero.

    También es necesario aplicar restricción para que la cantidad monetaria sea estrictamente positiva, por lo que debe ser mayor que 0. Sin embargo, se considera que es posible realizar un patrocinio de forma altruista. Este argumento es respaldado por el atributo anterior, type, en el que uno de sus posibles valores es *In_kind*, que indica que el patrocinio se realiza a coste 0. Por ello, el patrocinio solo puede ser gratuito si type toma el valor *In_kind.* Estas dos restricciones se aplicarán directamente sobre el servicio debido a los siguientes motivos:

    - Min(0) no se puede implementar con la notación @Min(0) directamente sobre la entidad Sponsorship puesto que el atributo amount es del tipo Money implementado por el framework.
    - La relación entre type y el valor de amount solo se puede realizar en el servicio pues relaciona dos atributos.
5. **email**: atributo opcional de tipo String que representa la dirección de correo electrónico del encargado del patrocinio. Tiene que cumplir la restricción que valida que el formato es correcto (Email).
6. **link**: atributo opcional de tipo String que representa una dirección web de información sobre el patrocinio. Debe cumplir la restricción que valida que el formato sea el correcto (URL).
7. **draftMode**: atributo de tipo Boolean que representa, como veremos en el siguiente requisito, si la composición que asocia las entidades SponsorShip e Invoice está completa.

4.2.2 Entidad Invoice

Each sponsorship is billed through the use of invoices. The system must store the following data about them: a code (pattern "IN-[0-9]{4}-[0-9]{4}", not blank, unique), a registration time (in the past), a due date (at least one month ahead the registration time), a quantity (positive not nought), the tax that it is applied (positive or nought), the total amount (calculated by adding together the quantity and the tax applied), and an optional link with further information.

Este requisito conlleva la creación de una entidad Invoice que guarda información sobre las facturas emitidas a partir de un patrocinio. Por ello, esta entidad debe estar relacionada con Sponsorship. No obstante, nuevamente es necesario analizar qué tipo de relación es la más adecuada:

1. **Relación simple Many to One con multiplicidad 0..*:** esta opción conlleva que la relación admite que una entidad Sponsorship no tenga ninguna entidad Invoice asociada.
2. **Agregación con multiplicidad 0..*:** caso similar al 1 pero se entiende la entidad Invoice como una parte de Sponsorship.
3. **Relación simple Many to One con multiplicidad 1..*:**esta opción conlleva que la relación admite que una entidad Sponsorship tenga al menos entidad Invoice asociada.
4. **Agregación con multiplicidad 1..*:** caso similar al 3 pero se entiende la entidad Invoice como una parte obligatoria de Sponsorship.

En este requisito se ha seleccionado la opción 4. El motivo es que todo patrocinio debe tener obligatoriamente una factura asociada al estar tratando con pagos. Además, una factura se considera una parte de un patrocinio, por lo que no tendría sentido tratar a Invoice y Sponsorship como entidades diferentes que se relacionan mediante una relación simple.

A continuación se muestra el análisis de las restricciones de los atributos de la entidad. Como nota, se considerará que un atributo es obligatorio a menos que se indique explícitamente lo contrario:

1. **code**: código que identifica una instancia de la entidad Sponsorship, de tipo String. Este atributo no puede ser una cadena vacía (NotBlank), debe ser único (Unique) y debe seguir el formato especificado por la expresión regular *IN-[0-9]{4}-[0-9]{4}*. Un ejemplo de una cadena que cumple el patrón de la expresión es IN-1234-1234.
2. **registrationTime** y **dueDate:** de tipo Date, este par de atributos representa la fecha de registro de la factura (registrationTime) y la fecha límite en la que se debe pagar (dueDate). Ambos deben ser obligatorios (NotNull) y en el caso de registrationTime, este debe ser un momento pasado (Past). En cuanto a qué datos guardar en la base de datos para manejar estos atributos se han barajado dos opciones:
    a. **Fecha y hora (Temporal(TIMESTAMP):** permite una definición del periodo más detallada, pudiendo especificar el momento exacto de inicio y fin. Ocupa más espacio en memoria y es ligeramente más complejo de manejar.
    b. **Fecha Temporal(Date)):** es ligeramente más simple de manejar y ocupa menos espacio en memoria. Sin embargo, limita más el nivel de definición que se puede alcanzar.

La opción elegida ha sido la a, fecha y hora. El motivo es que es necesario saber el momento exacto de emisión y la fecha exacta hasta la que se puede pagar.

Estos dos momentos deben estar separados por al menos un mes, siendo registrationTime el primero de los dos cronológicamente. Puesto que esta restricción relaciona dos atributos, será implementada sobre el servicio.

3. **quantity**: representa la cantidad a pagar en la factura sin contar impuestos. Atributo obligatorio (NotNull) Se han barajado tres posibles tipos para este atributo:
    a. **Integer**: simple de implementar pero imposibilita poner valores decimales.
    b. **Double**: simple de implementar y permite valores decimales.
    c. **Money**: tipo de dato personalizado que incluye el framework que permite poner cantidades decimales y especificar el tipo de moneda usada.

    Se ha optado por la opción 3 puesto que en el contexto del proyecto es necesario poder diferenciar los distintos tipos de divisa. Además, también se necesita poder manejar cantidades decimales de dinero.

    También es necesario aplicar restricción para que la cantidad monetaria sea estrictamente positiva, por lo que debe ser mayor que 0. No obstante, por el mismo motivo que con el atributo amount del requisito anterior, esta cantidad sí que deberá poder ser 0. Esta restricción se implementará directamente en el servicio al no poderse implementar con anotaciones sobre la entidad por ser un tipo de dato personalizado ofrecido por el framework.

4. **tax**: de tipo double, representa el impuesto que se aplica sobre quantity. Al ser un porcentaje, su valor debe estar entre 0 (Min(0)) y 1 (Max(1)).
5. **totalAmount**: atributo derivado (Transient) de tipo Money, se calcula como la suma del atributo quantity más el atributo tax por quantity. Se debe devolver la cantidad en la misma moneda que en el atributo quantity.
6. **link:** atributo opcional de tipo String que representa una dirección web de información sobre la factura. Debe cumplir la restricción que valida que el formato sea el correcto (URL).

4.2.3 SponsorDashboard

The system must handle sponsor dashboards with the following data: total number of invoices with a tax less than or equal to 21.00%; total number of sponsorships with a link; average, deviation, minimum, and maximum amount of the sponsorships; average, deviation, minimum, and maximum quantity of the invoices.

Este requisito expresa que el sistema debe manejar datos derivados de de las entidades Sponsorship y de invoice. Estos datos que se piden serán calculados mediante consultas relativamente simples a la base de datos, por lo que no será necesario persistir los datos calculados. Estas consultas se implementarán sobre el repositorio. Por ello, estos datos se implementarán como un Form con los siguientes atributos, que al ser derivados no es necesario validar puesto que se suponen correctos de primera instancia:

- **numInvoicesLessTax**: de tipo Integer, representa el número de facturas con impuestos menores o iguales a 21%.

- **numLinkedSponsorship**: de tipo Integer, representa el número de patrocinios que tienen asociados un enlace web.
- **averageAmountSponsorship**: de tipo Money, representa la media del atributo amount de las entidades Sponsorship.
- **deviationAmountSponsorship**: de tipo Money, representa la desviación típica del atributo amount de las entidades Sponsorship.
- **minAmountSponsorship**: de tipo Money, representa el valor mínimo del atributo amount de las entidades Sponsorship.
- **maxAmountSponsorship**: de tipo Money, representa el valor máximo del atributo amount de las entidades Sponsorship.
- **averageAmountInvoice**: de tipo Money, representa la media del atributo quantity de las entidades Invoice.
- **deviationAmountInvoice**: de tipo Money, representa la desviación típica del atributo quantity de las entidades Invoice.
- **minAmountInvoice**: de tipo Money, representa el valor mínimo del atributo quantity de las entidades Invoice.
- **maxAmountInvoice**: de tipo Money, representa el valor máximo del atributo quantity de las entidades Invoice.

4.2.4 Rol Sponsor

There is a new project-specific role called sponsor, which has the following profile data: name (not blank, shorter than 76 characters), a list of expected benefits (not blank, shorter than 101 characters), an optional web page with further information, and an optional email contact.

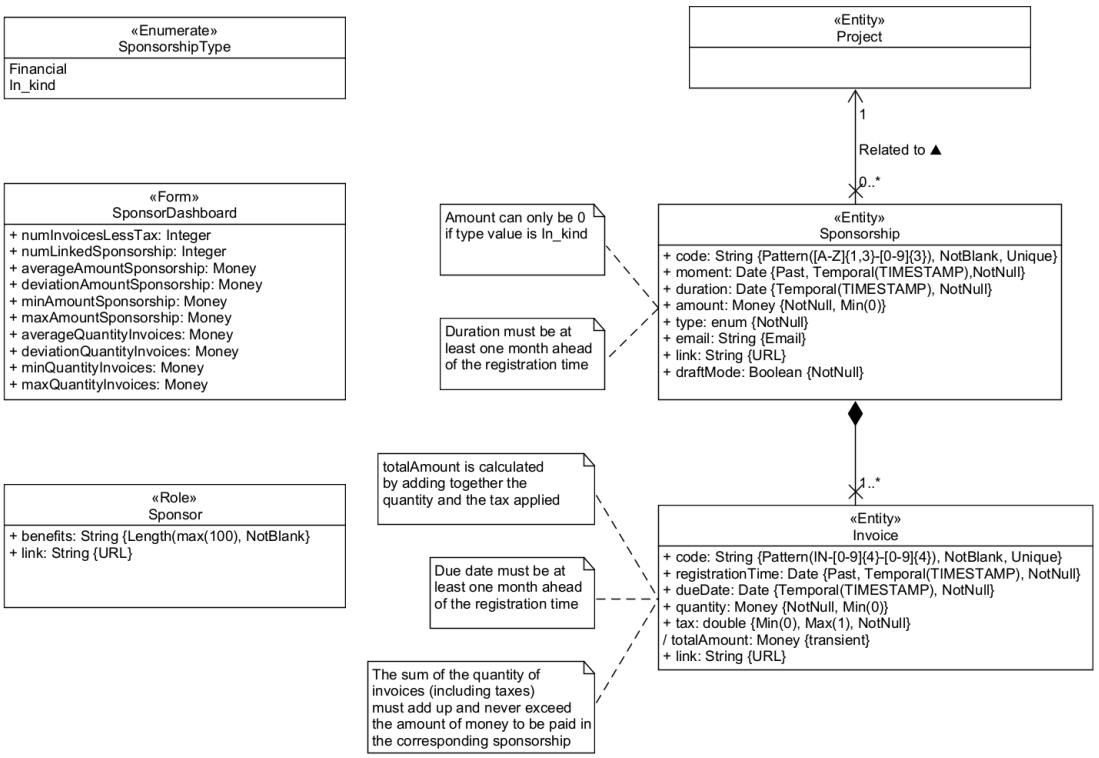
Este requisito implica implementar un nuevo rol en el sistema que represente a patrocinadores. Por ello, se implementará usando una entidad que extienda a AbstractRole. El requisito pide que se guarde información que se ha modelado con los siguientes atributos (se considera obligatorio un atributo a menos que se indique explícitamente lo contrario):

- **name**: nombre de la persona asociada al rol, de tipo String. Debe ser menor de 76 caracteres (Length(max(75)) y no puede ser una cadena vacía (NotBlank).
- **benefits**: representa la lista de beneficios que obtiene el patrocinador. Como en los requisitos se indica que no puede ser una cadena vacía (NotBlank) y que tiene que tener una longitud menor que 76 caracteres (Length(max(75)) se asume que este atributo se modela usando el tipo String.
- **link**: de tipo String, representa un enlace a una página web que ofrece más información sobre el patrocinador. Es un atributo opcional. que debe cumplir una restricción que valida que el texto esté en el formato correcto (Email).
- **email**: de tipo String, representa el correo electrónico del patrocinador. Es una tributo opcional que debe cumplir una restricción que valida que el texto esté en el formato correcto (URL).

Los la información a guardar de un patrocinador queda clara, pero presenta un problema. En el sistema, un rol se modela extendiendo la entidad abstracta AbstractRole. Esta entidad abstracta tiene presenta una relación ManytoOne (obligatoria) con la entidad UserAccount, que a su vez, tiene un atributo de tipo DefaultUserEntity. Esta última obliga a guardar en sus

atributos el nombre completo del usuario y su correo electrónico, por lo que no sería necesario modelar los atributos name y email al estar ya implementados por el framework.

4.2.5 Diagrama UML resultante del análisis



5. Conclusión

Este documento está actualmente en su segunda versión, correspondiente al segundo entregable. Los requisitos recogidos en esta versión han sido más complejos que los de la anterior por lo que han requerido una análisis en profundidad. Gracias a este análisis se han encontrado requisitos que estaban mal elicitados y que han sido corregidos.

6. Bibliografía

Intencionadamente en blanco.