Informe de Análisis D02

Integrantes del grupo C1.026:

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

Fecha: Sevilla, 08 de Marzo 2024

Tabla de contenido

[Resumen del Informe 4](#_Toc160821582)

[Historial de Versiones 5](#_Toc160821583)

[Introducción 6](#_Toc160821584)

[Informe de análisis – Acme-SF-D02 7](#_Toc160821585)

[Registro de análisis 1 7](#_Toc160821586)

[Requisito 7](#_Toc160821587)

[Análisis y decisiones 7](#_Toc160821588)

[Validación del profesor 7](#_Toc160821589)

[Registro de análisis 2 7](#_Toc160821590)

[Requisito 7](#_Toc160821591)

[Análisis y decisiones 8](#_Toc160821592)

[Validación del profesor 8](#_Toc160821593)

[Registro de análisis 3 8](#_Toc160821594)

[Requisito 8](#_Toc160821595)

[Análisis y decisiones 8](#_Toc160821596)

[Validación del profesor 8](#_Toc160821597)

[Registro de Análisis 4 9](#_Toc160821598)

[Requisito 9](#_Toc160821599)

[Análisis y decisiones 9](#_Toc160821600)

[Validación del profesor 9](#_Toc160821601)

[Registro de análisis 5 9](#_Toc160821602)

[Requisito 9](#_Toc160821603)

[Análisis y decisiones 9](#_Toc160821604)

[Validación del profesor 9](#_Toc160821605)

[Registro de Análisis 6 10](#_Toc160821606)

[Requisito 10](#_Toc160821607)

[Análisis y decisiones 10](#_Toc160821608)

[Validación del profesor 10](#_Toc160821609)

[Registro de Análisis 7 10](#_Toc160821610)

[Requisito 10](#_Toc160821611)

[Análisis y decisiones 10](#_Toc160821612)

[Validación del profesor 10](#_Toc160821613)

[Registro de Análisis 8 10](#_Toc160821614)

[Requisito 10](#_Toc160821615)

[Análisis y decisiones 10](#_Toc160821616)

[Validación del profesor 11](#_Toc160821617)

[Conclusiones 12](#_Toc160821618)

[Bibliografía 13](#_Toc160821619)

# Resumen del Informe

La finalidad de este informe es ofrecer información acerca del análisis de los requisitos específicos que se debían entregar en la segunda entrega del proyecto. Además, se describen las decisiones que se han tomado durante el transcurso de este y, en caso necesario, se establecerán conexiones con las validaciones y observaciones proporcionadas por el profesor a cargo de la asignatura.

# Historial de Versiones

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Contenidos** | **Fecha** | **Contribuyente** |
| V1.0 | Reporte de análisis individual. | 08/03/2024 | Joaquín González Ganfornina |

# Introducción

Para esta entrega, se nos ha encomendado obligatoriamente resolver tres requisitos de información de manera individual. En mi caso, he tenido que crear varias entidades en el proyecto, para satisfacer con lo que pedían estos requisitos.

Además de eso, hay un requisito obligatorio en el que debemos generar datos para testear la aplicación de manera informal. Para ello, se han creado y modificado archivos “csv” con datos de las entidades que creamos para los requisitos anteriores.

Finalmente, y de forma voluntaria, se pedía generar un diagrama UML, el presente informe de análisis, un informe de la planificación y el progreso y crear un nuevo rol llamado “auditor”.

En los siguientes apartados se describe las acciones tomadas para cumplir con estos requisitos con más detalle.

# Informe de análisis – Acme-SF-D02

## Registro de análisis 1

### Requisito

El sistema debe almacenar la siguiente información sobre “code audits”. Un código (con patrón “[A-Z]{1,3}-[0-9]{3}”, sin estar en blanco, único), una fecha de ejecución (en pasado), un tipo (“Static”, “Dynamic”), una lista de acciones correctivas propuesta (sin estar en blanco, con menos de 101 caracteres), una nota (calculada como la moda de las marcas en los correspondientes “audit record”; los empates pueden romperse arbitrariamente), y un enlace opcional con más información.

### Análisis y decisiones

Este requisito era bastante simple. Siguiendo la descripción del requisito, se creó una entidad “CodeAudit.java” con los atributos definidos, aplicando las restricciones correspondientes. Esta entidad se guardó dentro del paquete “codeaudits”.

Los atributos "code", "correctiveActions", “markMode” y "link" fueron definidos como tipo String, restringiendo sus valores según la descripción del requisito. Por otro lado, el atributo "executionDate" se designó como tipo Date y se aplicó una restricción para que sea una fecha en pasado. Para el atributo “type”, se creó un enum llamado “CodeType.java” con los valores correspondientes, que se almacena también en el paquete “codeaudits”.

Esta entidad tiene relaciones con las entidades “auditor” y “project”, ambas son ManyToOne.

### Validación del profesor

Tras la revisión del profesor, se decidió cambiar el tipo de “markMode”, que en principio se implementó como un enum, para que fuera del tipo String con un patrón. Además, se añadió la relación con “auditor”, ya que en principio esta relación no se contempló.

## Registro de análisis 2

### Requisito

El resultado de cada auditoría de código se basa en el análisis de sus “audit record”. El sistema debe almacenar la siguiente información sobre ellos: un código (con patrón “AU-[0-9]{4}-[0-9]{3}”, sin estar en blanco, único), el periodo durante el cual fue dictaminado (en el pasado, como mínimo de una hora), una nota (“A+”, “A”, “B”, “C”, “F”, or “F-”), y un enlace opcional con más información.

### Análisis y decisiones

Para satisfacer el requisito, se creó una entidad “AuditRecord.java” con los atributos definidos, aplicando las restricciones correspondientes. Esta entidad se guardó dentro del paquete “auditrecords”.

Los atributos "code", "mark" y "link" fueron definidos como tipo String, restringiendo sus valores según la descripción del requisito. Para la duración, se crearon dos atributos: “startDate” y “finishDate”. Estos dos atributos tienen la anotación @Past para que las fechas sean en pasado. La restricción de que la duración debe ser de una hora todavía no la hemos podido aplicar.

Esta entidad tiene una relación ManyToOne con la entidad “codeAudit”.

### Validación del profesor

Tras la revisión del profesor, se decidió cambiar el tipo de “mark”, al igual que el “markMode” de la entidad “codeAudit”.

## Registro de análisis 3

### Requisito

El sistema debe administrar “dashboards” de los auditores con la siguiente información: el número total de “code audits” para los tipos “Static” y “Dynamic”; la media, desviación, mínimo y máximo número de “audit records”; media, desviación, mínimo y máximo tiempo de la longitud del periodo. Todos estos datos para los “audit records” de ese auditor.

### Análisis y decisiones

Para satisfacer el requisito, se creó una clase “AuditorDashboard.java” con los atributos que se definen. Esta entidad se guardó dentro del paquete “forms”.

Los atributos “totalStaticCodeAudits” y “totalDynamicCodeAudits”; fueron definidos con el tipo primitivo int. En cambio, para los atributos “minumumNumberOfAuditRecords” y “maximumNumberOfAuditRecords” se usó el tipo Integer ya que es posible que exista un conjunto vacío, y este no tiene valores mínimos o máximos por lo que puede tener el valor null. restringiendo sus valores según la descripción del requisito. Finalmente, para los atributos “averageNumberOfAuditRecords”, “deviationNumberOfAuditRecords”, “averagePeriodLengthOfAuditRecords”, “deviationPeriodLengthOfAuditRecords”, “minumumPeriodLengthOfAuditRecords” y “maximumPeriodLengthOfAuditRecords” se usó el tipo Double por la misma razón que se usó anteriormente el tipo Integer, es decir, parar que no se dé ningún error al existir un conjunto vacío.

### Validación del profesor

Tras la revisión, cambiamos algunos datos de los tipos primitivos “int” y “double” a los tipos “Integer” y “Double” dado que, en primera instancia, no contemplamos los conjuntos vacíos.

## Registro de Análisis 4

### Requisito

Genera datos para testear tu aplicación de manera informal. Debe incluir dos cuentas de auditor con las credenciales “auditor1/auditor1” y “auditor2/auditor2”.

### Análisis y decisiones

Para incluir las cuentas con credenciales “auditor1/auditor1” y “auditor2/auditor2”, se modificó el archivo “user-account.csv” del paquete “src\main\webapp\WEB-INF\resources\sample-data”, añadiendo dos líneas para crear estas cuentas. Además, se añadieron otras 8 cuentas de auditor en este mismo archivo.

Finalmente, se generaron datos para probar todas las entidades “CodeAudit” y “AuditRecord” y para probar el rol “Auditor”. Esto se hizo creando archivos csv para cada clase java en el paquete “src\main\webapp\WEB-INF\resources\sample-data”. Para cada entidad, se generaron 10 objetos de prueba. Para los atributos de todas las entidades, se prueban los valores máximos, valores cercanos al máximo, valores mínimos, valores cercanos al mínimo y varios valores intermedios. Además, para los atributos que pueden ser nulos, se comprueba lo que ocurre cuando tienen algún valor y cuando tienen valor null.

### Validación del profesor

Este requisito no fue revisado por el profesor durante la última clase práctica.

## Registro de análisis 5

### Requisito

Hay un nuevo rol específico en el proyecto llamado auditor, que tiene los siguientes datos de perfil: firma (no en blanco, con menos de 76 caracteres), un id profesional (no en blanco, con menos de 26 caracteres), una lista de certificados (no en blanco, con menos de 101 caracteres) y un enlace opcional con más información.

### Análisis y decisiones

Para satisfacer el requisito, se creó una clase “Auditor.java” con los atributos que se definen. Esta entidad se guardó dentro del paquete “roles”.

Los atributos "firm", "profesionalId", “certifications” y "link" fueron definidos como tipo String, restringiendo sus valores según la descripción del requisito.

### Validación del profesor

Tras la revisión del profesor, no fue necesario cambiar nada de esta clase.

## Registro de Análisis 6

### Requisito

Crea un diagrama UML.

### Análisis y decisiones

En este requisito opcional, se pedía crear un diagrama UML con las entidades creadas en los requisitos anteriores. Para ello he usado la herramienta “Umlet” que se encontraba en el Workspace. Esta tarea se llevó a cabo sin mayores dificultades.

### Validación del profesor

Este requisito no fue revisado por el profesor durante la última clase práctica.

## Registro de Análisis 7

### Requisito

Produce un informe de análisis.

### Análisis y decisiones

En este requisito opcional, se pedía crear el presente informe de análisis. Para realizar este informe, se siguió con la misma estructura de la entrega anterior, pero se analizó el trabajo correspondiente a esta entrega.

### Validación del profesor

Este requisito no fue revisado por el profesor durante la última clase práctica.

## Registro de Análisis 8

### Requisito

Produce un informe de progreso y planificación.

### Análisis y decisiones

Al igual que en este documento, para el informe de progreso y planificación se utilizó la misma estructura que para la anterior entrega, adaptándola al contenido de la presente entrega.

### Validación del profesor

Este requisito no fue revisado por el profesor durante la última clase práctica.

# Conclusiones

En esta entrega, se debíamos satisfacer más requisitos que en la primera entrega, por lo que su implementación ha resultado más compleja.

Por otro lado, los requisitos eran muy similares entre ellos y se podían satisfacer de manera sencilla gracias a las trasparencias de clase y tomando como referencia los proyectos “Starters”

En conclusión, a pesar de que esta entrega ha tenido más carga de trabajo que la anterior, su implementación no ha sido excesivamente compleja, y aunque nos hemos encontrado con algunos errores puntuales, los hemos resuelto de forma eficiente.

# Bibliografía

En blanco a propósito.