

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Individual Analysis Report




Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas 2


Curso 2022 – 2023

Grupo de prácticas: C1-03-09	
Autores por orden alfabético	Rol
León Madroñal, Juan Carlos	Tester, Desarrollador

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)


Índice de contenido

1. Resumen ejecutivo	2
2. Tabla de revisiones	4
3. Introducción	5
4. Análisis	6
5. Conclusiones	7
7. Bibliografía	8

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)


1. Resumen ejecutivo

En este documento se detallarán los problemas que se han encontrado en los requisitos junto a las respectivas decisiones tomadas para resolverlos y las ventajas y desventajas de elegir una u otra alternativa.

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)


2. Tabla de revisiones

Fecha	Versión	Descripción
17/03/2023	1.0	Primera versión del documento

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)

3. Introducción

A continuación se deja constancia de los análisis que se han tenido que hacer sobre algunos requisitos, aunque debo decir que no ha habido gran problema con el entendimiento de la mayoría de estos.

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)

4. Análisis

1) **[MANDATORY]** *The system must store the following data about the sessions: a title (not blank, shorter than 76 characters), an abstract (not blank, shorter than 101 characters), **an indication on whether it can be considered a theory session or a hands-on session**, a time period (at least one day ahead, from one up to five hour long), and an optional link with further information.*

En este requisito se nos pide almacenar una variable que indique dos posibles valores. Como primera opción pensamos hacerlo con un valor tipo booleano ya que en términos de performance es mucho más óptimo. Sin embargo, tras hablar con el cliente se entendió que era probable que posteriormente hubiera más tipos de sesiones, así que un tipo booleano ya no nos servía. Finalmente, nos vimos en la obligación de hacerlo con un enumerado común (ya que involucra a requisitos de otras partes del proyecto) al que llamamos *ActivityType*.


2) *There is a new project-specific role called assistant, which has the following profile data: supervisor (not blank, shorter than 76 characters), **list of expertise fields** (not blank, shorter than 101 characters), résumé (not blank, shorter than 101 characters), and an optional link with further information.*

3) **[MANDATORY]** *A tutorial provides additional support to a course by means of one or more sessions. The system must store the following data about them: a code (pattern “[A-Z]{1,3}[0-9][0-9]{3}”, not blank, unique), a title (not blank, shorter than 76 characters), an abstract (not blank, shorter than 101 characters), **some goals** (not blank, shorter than 101 characters), and an estimated total time.*

En cuanto a estos dos requisitos, en concreto a las variables que aparecen en negrita, hubo también una discusión a la hora de cómo modelarlo, ya que a simple vista se tratan las dos de una “lista” de valores.


En una primera versión, decidimos extraer las dos propiedades en entidades a las cuales se le establecían unas relaciones con la clase “padre”, de modo que quedaba una simple entidad con una única propiedad, algo que es muy poco eficiente, pero que tiene una gran ventaja si en algún momento se decide escalar estos *expertise fields* y *goals* para así añadirle más propiedades.

En conclusión y tras la revisión semanal con el cliente, decidimos que lo más eficiente sería almacenar las “listas” como una propiedad tipo String (ya que no se tenía previsto escalar las propiedades como describimos anteriormente) y que, a la hora de persistir los datos, cada elemento de estas listas estuviese separado por comas, de modo que solo tendríamos que parsear la cadena para acceder a los datos.

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)

5. Conclusiones

En conclusión, tras el análisis realizado de los requisitos, podemos afirmar que en este sprint se han empezado a presentar los primeros conflictos y ambigüedades con los requisitos descritos por el cliente, con el que ha habido una comunicación fluida para resolverlos sin ningún tipo de problema.

	Diseño y Pruebas II Acme-Life-Long-Learning-Inc
	Analysis Report (jualeomad)

7. Bibliografía

Intencionalmente en blanco.