Student #1, Sprint 2: Analisis report

Group: C1.04.14

Repository: https://github.com/marizqlav/Acme-L3-D01

Student #1

Name: Domínguez-Adame, Alberto
email: albdomrui@alum.us.es

Student #2

Name: Herrera Ramírez, Ismael
email: ismherram@alum.us.es

Student #3

Name: Olmedo Marín, Marcos
email: marolmmar1@alum.us.es

Student #4

Name: Izquierdo Lavado, Mario
email: marizqlav @alum.us.es

Student #5

Name: Merino Palma, Alejandro
email: alemerpal@alum.us.es

Table of contents

Summary

Acme Life-Long Learning, Inc. (Acme L3, Inc. for short) is a company that specializes in helping learners get started on a variety of matters with the help of renowned lecturers. The goal of this project is to develop a WIS to help this organization manage their business.

Revision table

Number	Date	Description
1	15/03/2023	Full redaction of the document.

Introduction

This document lists analysis records, each one including the following data: a verbatim copy of the requirement to which the record refers; detailed conclusions from the analysis and decisions made to mend the requirement; a link to the validation performed by a lecturer. .

This document has the following structure:

- Analysis report

Contents

Analysis records

Requirement nº4 Lecturer: There is a new project-specific role called lecturer, which has the following profile data: alma mater (not blank, shorter than 76 characters), a résumé (not blank, shorter than 101 characters), list of qualifications (not blank, shorter than 101 characters), and an optional link with further information.

Decisions:

- One decision to be made is how to implement the list of qualifications, which we could implement as a list of string or a long single string.

- Another one is the relation between Lecturer and both Lectures and Courses.

Conclusions: Through a discussion in class with the teacher and the other groups we have determined that:

- Given the restrictions (not blank, shorter than 101 characters) the easiest method was to implement it with a single String.

- The best way to implement the relations will be a ManyToOne from both Course and Lecture to Lecturer.

Requirement nº5 Course: A course aggregates several lectures by the same lecturer. The system must store the following data about them: a code (pattern "[A-Z]{1,3} [0-9]{3}", not blank, unique), 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 course or a hands-on course (depending on the lectures that it aggregates), a retail price (positive or nought), and an optional link with further information. Purely theoretical courses must be rejected by the system.

Decisions:

- To indicate if a course is hands-on or theoretical, i will implement a function in the service that will calculate the number of each type of lecture. If theory>hands-on, the course will be a theory course, and if hands-on>theory, then it will be hands-on. There is also the third possibility that theory=hands-on.

- The type of course is a derived attribute as it depends on the number of theory and hands-on lectures. It can be persistent or not.

Conclusions:

- As discussed with our teacher in class, I have decided that in case there is the exact number of theory and hands-on courses, a third choice will be added in the enumerated LectureType in case the third option occurs. This option will be "balanced".

- Again, as discussed with our teacher the easiest method to implement derived attributes in most cases will be to make them non persistent.

Requirement nº6 Lecture: A lecture is a document that a lecturer uses to get some knowledge across. The system must store the following data about them: a title (not blank, shorter than 76 characters), an abstract (not blank, shorter than 101 characters), an estimated learning time (in hours, positive, not nought), a body (not blank, shorter than 101 characters), an indication on whether it can be considered theoretical or hands-on, and an optional link with further information.

Decisions:

- To implement Lecture I have made many decisions. Firstly for the attribute lectureType, I have decided to implement it as an Enumerated with three values: theory, handson and balanced. The balanced option is not specified in the requirements, but it will be used when a lecture has the same number of theoretical and hands-on lectures. This decision was approved by the teacher in class.

- The estimated learning time, as it is an estimate and does not need to be calculated between two periods of time, I will implement it as a Double to make it easier to work on. Again this decision has been reviewed by our teacher in a follow up session.

Relation between Requirement nº6 & nº5: In order to implement the bidirectional relation between Course and Lecture i have decided to make a third class called CourseLecture, wich will have a ManyToOne from itself to both Course and Lecture, keeping the id's of both. This decision was reviewed by our teacher in laboratory sessions.

Requirement nº7 LecturerDashboard: The system must handle lecturer dashboards with the following data: total number of theory and hands-on lectures; average, deviation, minimum, and maximum learning time of the lectures; average, deviation, minimum, and maximum learning time of the courses.

Decisions: For the statistics of both lectures and courses (min, max, deviation, average), we have various options. Make each statistic an individual propertia of the class LecturerDashboards or, as suggested by our teacher, make two Maps<String, Date> one for lectures and the other for courses, where the string will contain the type of statistic (max, min, deviation), and the Date the value of the learning time of each statistic.

Decision: I will implement the second option, as it is more elegant and was suggested by our teacher in class.

Requirement nº8: It’s just to produce data to test the application. No analysis is required.

Conclusion

This deliverable had many requirements that required analysis. Through discussions with our teacher and other classmates we have achieved a satisfying solution to many of the problems encountered. However many of this solutions may not be final, as it may happen that in the next deliverable when we implement the functionalities, we may encounter new issues that force us to change our initial approach.

Bibliography

Intentionally blank.