CoursesManagementApp

Product Backlog Specification

VERSIONS HISTORY

Date	Version	Description	Author
2/14/22	<1.0>	1st version of the requirements definition document	A. Zarras

1 Introduction

The objective of this project is to develop a Web application that allows an instructor to manage the grading of the courses that he teaches.

The rest of this document is structured as follows. In Section 2 we focus on the development process that shall be followed and other scoring and organizational issues. Sections 3 and 4 provide the Product Backlog, i.e., the "raw" functional and non-functional requirements that should be further analyzed to drive the design, implementation and testing of the application.

2 Development process and organization issues

To realize the project we shall rely on a Scrum approach. Each team shall organize a number of sprints during which the team shall implement **user stories** from the project backlog and their **tests**. The deadline for the project is: 18/5/2022.

2.1 Deliverables

Definition of "done" story: A user story is done if it is **implemented correctly** and validated with one or more appropriate **tests**.

At the end of the project the Scrum team shall **deliver (via turnin)**

- **The project implementation.**
- **A Sprint report,** according to the given **Sprint report template (SprintReport-v0.doc), describing the sprints that they performed and the "done" user stories that have been developed during each sprint**. The report shall also comprise the specification of **detailed use cases, derived from the given user stories,** the **detailed design** of the application and **CRC cards** that document the responsibilities and the collaborations between the different part of the application.
- **A DEMO video**, about 15' minutes, using a screen capture tool like ActivePresenter for example. In the demo you should illustrate that the user stories of the application are working. The demo further demonstrate the acceptance tests that have been developed for the application.
    - Put the demo video on GitHub, Google Drive or youtube. In the project folder that you turnin include a txt file named DEMO-LINK.txt that contains the link that points to the folder that contains the video.
- Turn in the **project** and the other deliverables using **turnin deliverables@myy803 <your-project>.zip**, where your-project is a zip file of your Eclipse project.

2.2 Scoring

1. Working implementations of the **user stories** is **40%** of the total score.
2. **Acceptance, integration and unit tests** is **20%** of the total score.
3. **Design quality**, usage of recommended **patterns** and **best practices** to satisfy the extensibility and maintainability requirements is **30%** of the total score.
4. Quality of **reporting** is **10%** of the total score.

3 Functional Requirements / User Stories

ID	AS A <User Type>	I WANT <An Action>	SO THAT <A Benefit/Value>
US1	Instructor	To login to the application with my user name and password.	To safely manage the courses that I teach.
US2	Instructor	To browse the list of my courses.	To manage the courses' descriptions and the students' grades.
US3	Instructor	To add a course in the list, by giving information like the course id, name, syllabus, year, semester, etc.	To populate the list of the courses with new ones.
US4	Instructor	To remove a course from the list.	Clean up the list of courses.
US5	Instructor	To update the description of a course.	Correct possible mistakes and keep the information up to date with the current situation.
US6	Instructor	To browse the list of students that enrolled to a particular course.	To manage the students that enrolled in the course.
US7	Instructor	To add a student to the list of a particular course, by giving information like the student id, name, year of registration, semester, etc.	To populate the list with the students that enrolled in the course.
US8	Instructor	To remove a student from the list of a particular course.	To deal with students that resigned from the course.
US9	Instructor	To update students' information (id, name, year of studies, etc.).	Correct possible mistakes and keep the information up to date with the current situation.
US10	Instructor	To register the grades of the student in the final exam and the project of the course.	To manage the grading of the students that enrolled

			in the course.
US11	Instructor	To calculate the overall grades of the students that enrolled in a particular course with respected to a weighted average.	To manage the grading of the students that enrolled in the course.
US12	Instructor	To calculate descriptive statistics about the students grades in a particular course, including min, max, mean, standard deviation, variance, percentiles, skewness, kurtosis, median.	Study the distribution of the students grades in the course.

4 Non Functional Requirements

[NF1] **Maintainability:** In software engineering, maintainability is the degree of effectiveness and efficiency with which a product or system can be modified by the maintainers. In the case of this project we specifically focus on the following concerns:

- Keep the application independent from the software library that is going to be used for the calculation of descriptive statistics.
- Be able to easily extend the list of descriptive statistics with new ones in the future.
- Keep the application independent from the database management system that is going to be used for the data.
- Be able to easily change the database management system for another without significant modifications to the application logic.
- Be able to easily extend the application with new functionalities in the future.
- Be able to change or add different views for the interaction with the provided services (e.g. add a restful api layer for using the application programmatically).

To achieve these maintainability concerns the application should be designed according to the well known principles and exploit best practices like the GoF design patterns (Strategy, Template Method, Adapter, etc. [1]) and Folwler's enterprise application architecture patterns [2].

[NF2] **Usability**: In software engineering usability concerns the ease of use and learnability. In the context of this project the application should provide a simple and user-intuitive interface. The application should also provide help, in the form of user guidelines, concerning its main functionalities of the application.

5 IDE, Java and External API Requirements/Constraints/Recommendations

The application should be implemented in **Java.** The application should be compatible at least with **Java 8**. You can further use the **Spring framework**. Spring and lately **Spring Boot** are very popular technologies that facilitate the development of **Web-based applications**. As a base database management system you can use **MySQL**. Moreover, a popular library for descriptive statistics is **Apache Math 3**. To write automated tests for the application you can use **JUnit and Mockito.** The preferred IDE is **Eclipse**.

6 References

[1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley.

[2] Martin Fowler. Catalog of Patterns of Enterprise Application Architecture. Addison-Wesley. https://martinfowler.com/eaaCatalog/