Laboratory Assignment AND Assessment Requirements Specification

Version 1.0

March, 2022

Developed by:

Foidaș Andrei Ștefan

Ghilea Ramona Claudia

933

Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Description of Change | Author | Date |
| V01 | Initial configuration of the project | Foidaș Andrei  Ghilea Ramona | 01.03.2022 |
|  |  |  |  |

**Contents**

[Laboratory Assignment AND Assessment Requirements Specification 1](#_Toc97582888)

[Version 1.0 1](#_Toc97582889)

[March, 2022 1](#_Toc97582890)

[1 Functional Requirements 3](#_Toc97582891)

[2 Actors 3](#_Toc97582892)

[3 Use cases – diagram 4](#_Toc97582893)

[3.1 Use case number 1 (Description of the use case) 4](#_Toc97582894)

[3.2 Use case number 2 (Description of the use case) 4](#_Toc97582895)

[3.3 Use case number 3 (Description of the use case) 5](#_Toc97582896)

[3.4 Use case number 4 (Description of the use case) 5](#_Toc97582897)

[3.5 Use case number 5 (Description of the use case) 5](#_Toc97582898)

[3.6 Use case number 6 (Description of the use case) 6](#_Toc97582899)

[3.7 Use case number 7 (Description of the use case) 6](#_Toc97582900)

[3.8 Use case number 7 (Description of the use case) 6](#_Toc97582901)

[3.9 Use case number 7 (Description of the use case) 7](#_Toc97582902)

[4 Analysis 7](#_Toc97582903)

[4.1 Entities 7](#_Toc97582904)

[4.2 Relations between entities 7](#_Toc97582905)

[4.3 Attributes 7](#_Toc97582906)

[4.4 System behavior 7](#_Toc97582907)

[4.4.1 Use case 1-2-3 7](#_Toc97582908)

[4.5 System events 8](#_Toc97582909)

[5 Design 8](#_Toc97582910)

[5.1 Class diagram 8](#_Toc97582911)

[5.2 Sequence diagrams (for each use case) 9](#_Toc97582912)

[5.3 GRASP 13](#_Toc97582913)

**Analysis and design Document**

# Functional Requirements

List the functional requirements (FR) of the system.

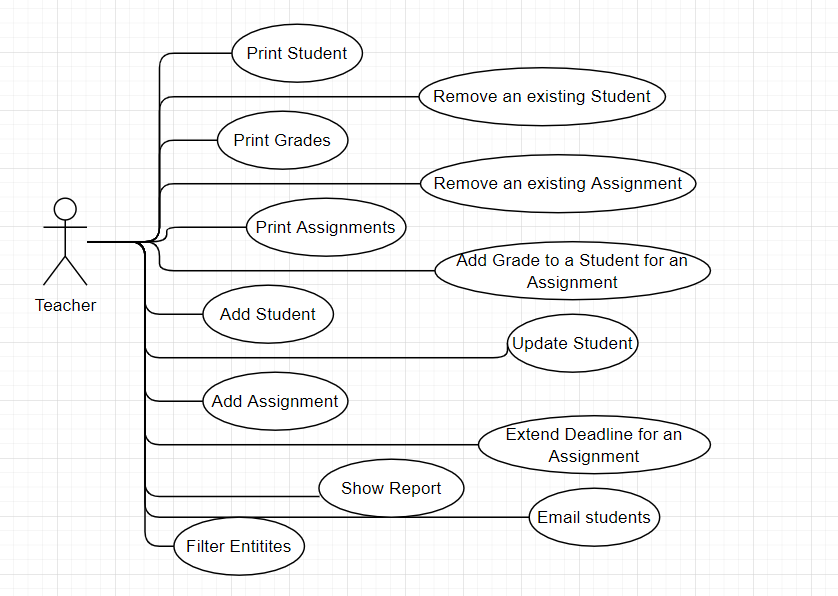
|  |  |
| --- | --- |
| Section/ Requirement ID | Requirement Definition |
| FR 1.1 | Add a new student |
| FR 1.2 | Remove a student |
| FR 1.3 | Update student |
| FR 1.4 | List all students |
| FR 2 | Add an assignment/laboratory theme |
| FR 3 | Extend the deadline for an assignment |
| FR 4 | Grade a student for an assignment |
| FR 5 | Remove an assignment |
| FR 6 | Show Report |
| FR 7 | Filter Entries |
| FR 8 | Email Students |

# Actors

Teacher

The only actors that interact with the application are teachers.

# Use cases – diagram



## Use case number 1 (Description of the use case)

Actors: teacher

Description: add a new student

Precondition: all fields are specified and valid

Postcondition: a new student was added in the list and repository size will increase by 1

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System Response |
| 1 Completes the necessary fields for adding |  |
|  | 2 Checks if everything is alright, adds a new element in the list if so |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 2 (Description of the use case)

Actors: teacher

Description: delete a student

Precondition: valid id belonging to an existing student is specified

Postcondition: the student with the specified id is removed from the list and the repository

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System response |
| 1 Give an id as input |  |
|  | 2 Checks if it is a valid id and there is a student with that id and deletes the student |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 3 (Description of the use case)

Actors: teacher

Description: update a student

Precondition: valid id belonging to an existing student and all other fields are specified and valid

Postcondition: the student with the specified id has the data updated

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System response |
| 1 Give an id and all other fields for the Student entity as input |  |
|  | 2 Checks if it is a valid id and there is a student with that id, than checks if the rest of the input is valid, and updates the data for that student |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 4 (Description of the use case)

Actors: teacher

Description: add a new assignment

Precondition: all fields are specified and valid

Postcondition: a new assignment was added in the list and repository size will increase by 1

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System Response |
| 1 Completes the necessary fields for adding |  |
|  | 2 Checks if everything is alright, adds a new element in the list if so |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 5 (Description of the use case)

Actors: teacher

Description: delete an assignment

Precondition: valid id belonging to an existing assignment is specified

Postcondition: the assignment with the specified id is removed from the list and the repository

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System response |
| 1 Give an id as input |  |
|  | 2 Checks if it is a valid id and there is an assignment with that id and deletes the assignment |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 6 (Description of the use case)

Actors: teacher

Description: update an assignment’s deadline

Precondition: valid id belonging to an existing assignment, valid number of weeks for the extension and the current week number is less than or equal to the number of weeks with the assignment deadline

Postcondition: the assignment with the specified id has the data updated

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System response |
| 1 Give an id and number of weeks for extension |  |
|  | 2 Checks if it is a valid id and there is an assignment with that id, than checks if the rest of the input is valid and that the conditions are met, and updates the data for that assignment |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 7 (Description of the use case)

Actors: teacher

Description: add grade to a student for an assignment

Precondition: valid ids belonging to existing student and assignment, valid fields for grade entity

Postcondition: a new grade was added in the list and repository size will increase by 1

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System response |
| 1 Completes the necessary fields for adding |  |
|  | 2 Checks if ids are valid and if input is ok, adds a new element in the list if so |
| 3 - | 3. If the input is invalid, throws an exception |

## Use case number 7 (Description of the use case)

Actors: teacher

Description: show report

Precondition: there is data in the repository so reports can be created

Postcondition: presents reports related to student’s grades for assignments

Exceptions: When the repository is empty.

|  |  |
| --- | --- |
| Action | System response |
| 1 Select report function |  |
|  | 2 Shows reports |
| 3 - | 3. If repository is empty, throws an exception |

## Use case number 7 (Description of the use case)

Actors: teacher

Description: filter entries

Precondition: valid input related to entries specific fields

Postcondition: a list of entries filtered by the given input

Exceptions: When the fields aren’t filled or valid.

|  |  |
| --- | --- |
| Action | System response |
| 1 Completes the necessary fields for filtering |  |
|  | 2 Checks if the input is ok, print filtered entries |
| 3 - | 3. If the input is invalid, throws an exception |

# Analysis

## Entities

Student, Assignment, Grade

## Relations between entities

One student can have multiple assignments and one assignment can be assigned to many students. It is a many-to-many relationship between the two classes. Class Grade has as id, a pair consisting of studentId and assignmentId and it is the association class between the Student and Assignment classes.

## Attributes

Student: id, name, group, email professorName

Assignment: id, description, deadline, assignationDate

Grade: id(studentId, assignmentId), value, deliverDate, feedback

## System behavior

## Use case 1-2-3

The system will act as a subsystem to a larger environment, in order to speed up a certain process in the company’s workflow.

## System events

After each operation a message is shown to the user either if the command terminated successfully or with an error message.

# Design

## Class diagram

****

## Sequence diagrams (for each use case)

* **Add Student Sequence Diagram**

****

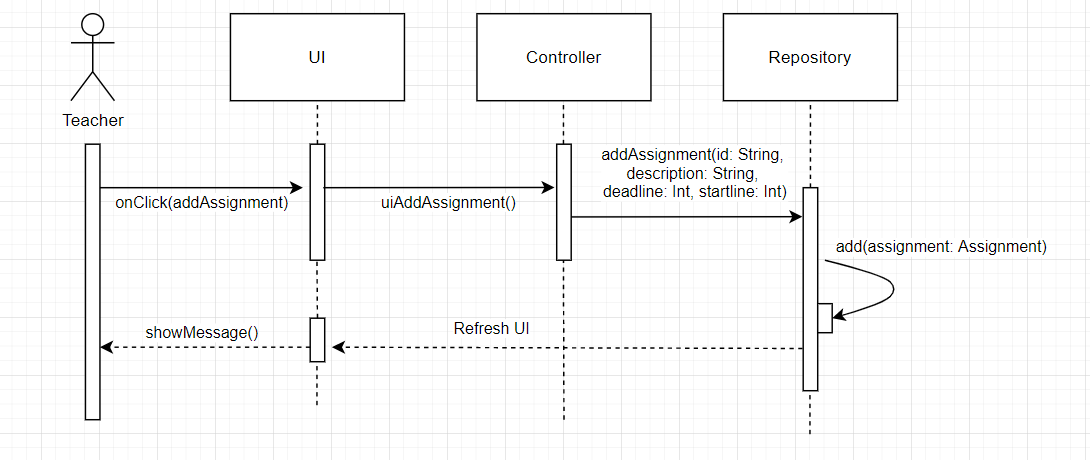
* **Delete Student Sequence Diagram**

****

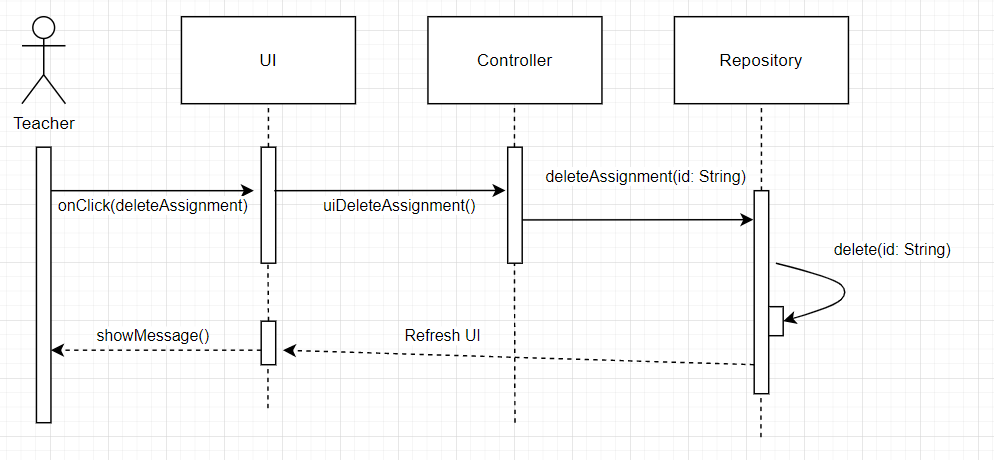
* **Update Student Sequence Diagram**

****

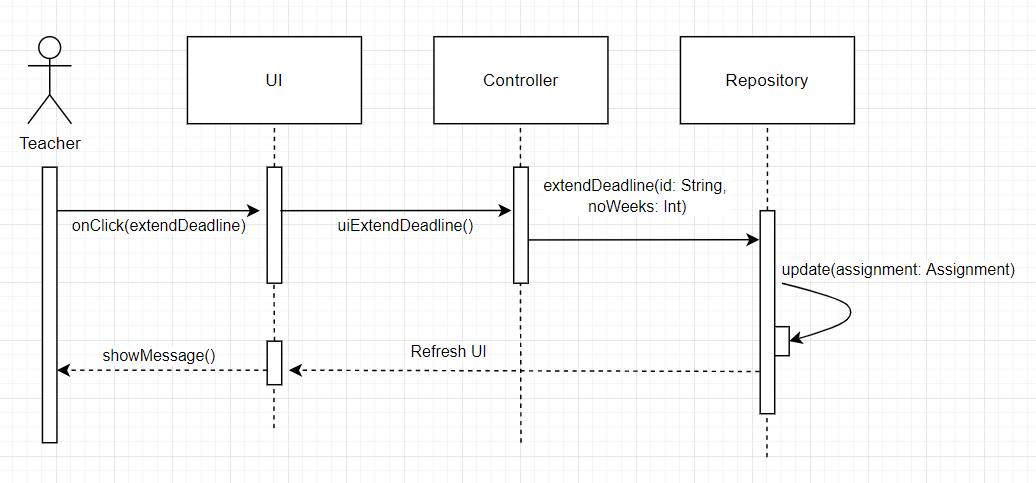
* **Add Assignment Sequence Diagram**

****

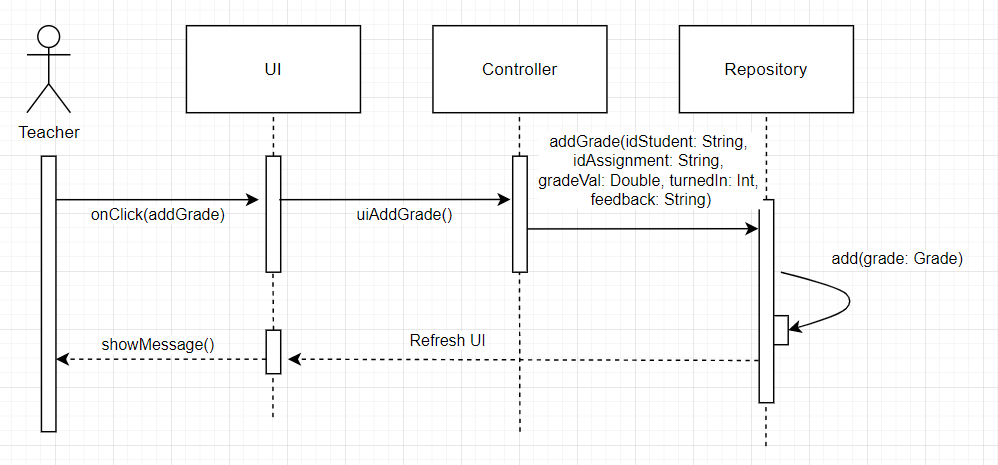
* **Delete Assignment Sequence Diagram**

****

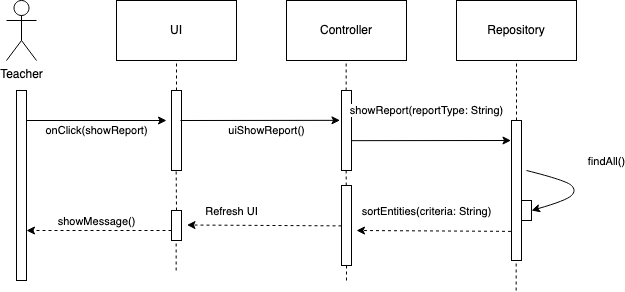
* **Extend Deadline Sequence Diagram**

****

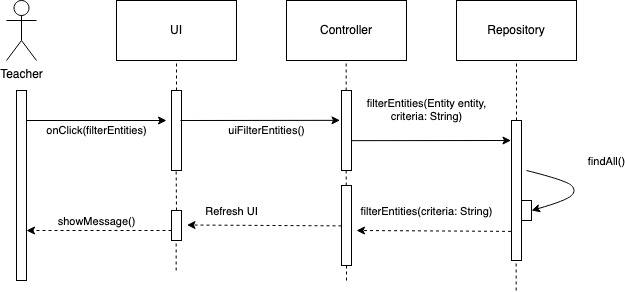
* **Grade Student for Assignment Sequence Diagram**

****

* **Show Report Sequence Diagram**



* **Filter Entries Sequence Diagram**



## GRASP

GRASP is set of exactly 9 **G**eneral **R**esponsibility **A**ssignment **S**oftware **P**atterns:

1. Information Expert

2. Creator

3. Controller

4. Low Coupling

5. High Cohesion

6. Indirection

7. Polymorphism

8. Pure Fabrication

9. Protected Variations