**Laboratory Assignment AND Assessment Requirements Specification**

Version 1.0

20 March 2024

Developed by:

Iulia Groza

Robert Harangus

933/1

Version History

| Version | Description of Change | Author | Date |
|---|---|---|---|
| V01 | Initial/Modification of document | Iulia Groza | 18 March 2024 |
| V02 | Completion of document | Robert Harangus | 20 March 2024 |

## Contents

## Analysis and design Document

## 1 Functional Requirements

List the functional requirements (FR) of the system.

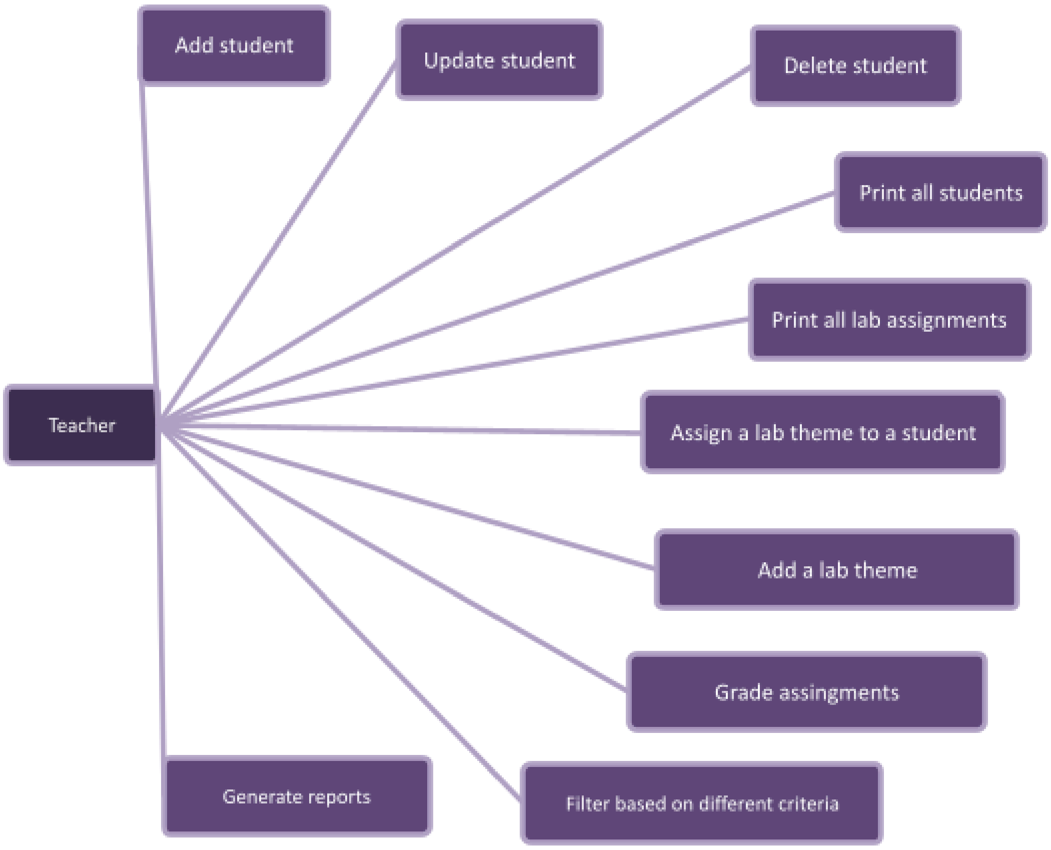
| Section/<br>Requirement ID | Requirement Definition |
|---|---|
| FR1.0 | Implement CRUD operations for the Student entity |
| FR1.1 | Add student |
| FR1.2 | Edit information about a student |
| FR1.3 | Delete student |
| FR1.4 | Show all students |
| FR1.5 | Find student |
| FR2.0 | Manage laboratory assignments and subjects. |
| FR2.1 | Add laboratory assignment |
| FR2.2 | Edit information about a laboratory assignment |
| FR2.3 | Delete laboratory assignment |
| FR2.4 | Show all laboratory assignments |
| FR2.5 | Extend the deadline for an existing subject |
| FR3.0 | Manage grading |
| FR3.1 | Assign a grade to a student on an assignment |
| FR3.2 | Edit a grade to a student on an assignment |
| FR3.3 | Delete a grade to a student on an assignment |
| FR3.4 | Show all grades |
| FR3.5 | Find grade of a student on an assignment |
| FR3.0 | Filter students based on different criteria |
| FR4.0 | Generate reports |
| FR4.1 | Generate reports of laboratory grades for each student |
| FR4.2 | Generate reports for hardest assignments |
| FR4.3 | Generate reports of students who can enter the exam |
| FR4.4 | Generate reports of students who delivered all assignments on time |
| FR5.0 | Notify students by email |

| Section/<br>Requirement ID | Requirement Definition |
|---|---|
| FR5.1 | Notify students by email when adding a new laboratory assignment |
| FR5.2 | Notify students by email when modifying the delivery date of an existing assignment |

## 2 Actors

- Teachers for the MAP subject

## 3 Use cases – diagram



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

Actors: Teacher

Description: Add student

Precondition: User enters valid data about a student

Postcondition: If a student is valid, then it is added to the student list.

| User action | System response |
|---|---|

| | |
|---|---|
| 1 Completes the necessary fields for adding | |
| | 2 Checks if the addition was successful. Adds the student if it is ok or displays an error message otherwise. Then it returns to menu |

Exceptions: When the fields aren't filled, id already exists, one of the fields has an invalid data type.

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

Actors: Teacher

Description: Edit information about a student

Precondition: User enters the id and the fields he wants to modify

Postcondition: If student with that id exists, then its data is updated

| User action | System response |
|---|---|
| 1 Completes the necessary fields for updating | |
| | 2 Checks if the student exists and all the fields are valid. If so, it updates it, otherwise it displays an error. Then it returns to menu |

Exceptions: The student with that id doesn't exist, one of the fields has an invalid data type, or one of the fields has not been filled.

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

Actors: Teacher

Description: Delete student

Precondition: User gives the id

Postcondition: If student with that id exists, then it is removed from the students list

| User action | System response |
|---|---|
| 1 Inputs the id | |
| | 2 Checks if the student exists. If so, it deletes it, otherwise it displays an error. Then it returns to menu |

Exceptions: The student with that id doesn't exist, or id is not filled.

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

Actors: Teacher

Description: Show all students

Precondition: -

Postcondition: All students are displayed on the screen.

| User action | System response |
|---|---|
| 1 | |
| | 2 Prints all students. Then it returns to menu |

### 3. 5 Use case number 5 (Description of the use case)

Actors: Teacher

Description: Find information about a specific student

Precondition: User gives a valid student id (existing one)

Postcondition: Data about the student is shown to the screen

| User action | System response |
|---|---|
| 1 Gives student id | |
| | 2 Check if the student exists. If it does, it prints the data, otherwise it displays an error. Then it returns to the menu. |

Exception: Student with id doesn't exist

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

Actors: Teacher

Description: Add a lab theme

Precondition: User gives valid information about theme

Postcondition: Theme is added to the repository

| User action | System response |
|---|---|
| 1 Completes the necessary fields for adding | |

| | |
|---|---|
| | 2 Checks if everything is alright. Adds the theme if it is ok or displays an error message otherwise. Then it returns to menu |

Exceptions: When the fields aren't filled, id already exists, one of the fields has an invalid data type.

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

Actors: Teacher

Description: Edit laboratory theme

Precondition: User enters a valid id of the theme he wants to modify, then enters valid data to all the fields of the theme

Postcondition: Data about the theme is updated

| User action | System response |
|---|---|
| 1 Completes the necessary fields for updating | |
| | 2 Check if the assignment exists and if all necessary fields are valid . If it does, it updates with a new set of data, otherwise it displays an error.<br>Then it returned to the menu. |

Exception: Theme with id doesn't exist, fields of invalid type

### 3.8 Use case number 8 (Description of the use case)

Actors: Teacher

Description: Delete laboratory theme

Precondition: User gives a valid theme id (existing one)

Postcondition: Data about the theme is deleted

| User action | System response |
|---|---|
| 1 Gives theme id | |

| | |
|---|---|
| | 2 Check if the assignment exists. If it does, it is deleted, otherwise it displays an error. Then it return to the menu. |

Exception: Theme with id doesn't exist

### 3.9 Use case number 9 (Description of the use case)

Actors: Teacher

Description: Print all laboratory themes

Precondition: -

Postcondition: All themes are shown on the screen

| User action | System response |
|---|---|
| 1 | |
| | 2 Prints all assignments, then it returns to menu |

### 3.10 Use case number 10 (Description of the use case)

Actors: Teacher

Description: Add a grade to a specific student on a specific laboratory theme.

Precondition: User gives a valid student (student without grade), theme and grade

Postcondition: Grade is added to the given student

| User action | System response |
|---|---|
| 1 Completes info about student, theme and grade | |
| | 2 If the student and the given theme exist and the grade is valid, it adds the grade to the student. Otherwise, it displays an error. Then it returns to menu |

Exceptions: When a student or theme doesn't exist, or student already has a grade, or one of the fields is not filled, or grade invalid type.

### 3.11 Use case number 11 (Description of the use case)

Actors: Teacher

Description: Delete a grade to a specific student on a specific laboratory theme.

Precondition: User gives a valid student (student with grade), theme

Postcondition: Grade is deleted from the given student

| User action | System response |
|---|---|
| 1 Completes info about student and theme | |
| | 2 If the student and the given theme exist and a grade is assigned, it deletes the grade of the<br>student. Otherwise, it displays an error. Then it returns to menu |

Exceptions: When a student or theme doesn't exist, or student doesn't have a grade, or one of the fields is not filled, or grade invalid type.

### 3.12 Use case number 12(Description of the use case)

Actors: Teacher Description: Print all grades

Precondition: -

Postcondition: All grades are shown to the screen

| User action | System response |
|---|---|
| 1 | |
| | 2 Prints all grades. Then it returns to menu |

### 3.13 Use case number 13 (Description of the use case)

Actors: Teacher

Description: Filter the students, assignments, themes and grades based on different criteria

Precondition: -

Postcondition: -

| User action | System response |
|---|---|

| | |
|---|---|
| 1 Completes the necessary fields | |
| | 2 Returns the result set of the selected filter.<br>Then returns to menu |

### 3.14 Use case number 14 (Description of the use case)

Actors: Teacher

Description: Generate reports

Precondition: Select report type

Postcondition: Information about report is shown to the screen

| User action | System response |
|---|---|
| 1 Choose one of the given reports types | |
| | 2 Shows the corresponding report |

# 4 Analysis

## 4.1 Entities

- Student
- Laboratory assignments
- Grades

## 4.2 Relations between entities

A student can have more assignments and an assignment can be assigned to more students.

A grade can be given for a specific student on an assignment.

## 4.3 Attributes

- Student
  - id: String
  - name: String
  - group: Int
  - email: String
  - teacher: String
- Grade
  - id: Map<String, Int>
  - st: Student
  - assign: Assignment

  - value: Float
  - date: Int
- Assignment
  - id: Int
  - description: String
  - deadline: Int
  - delivery_week: Int

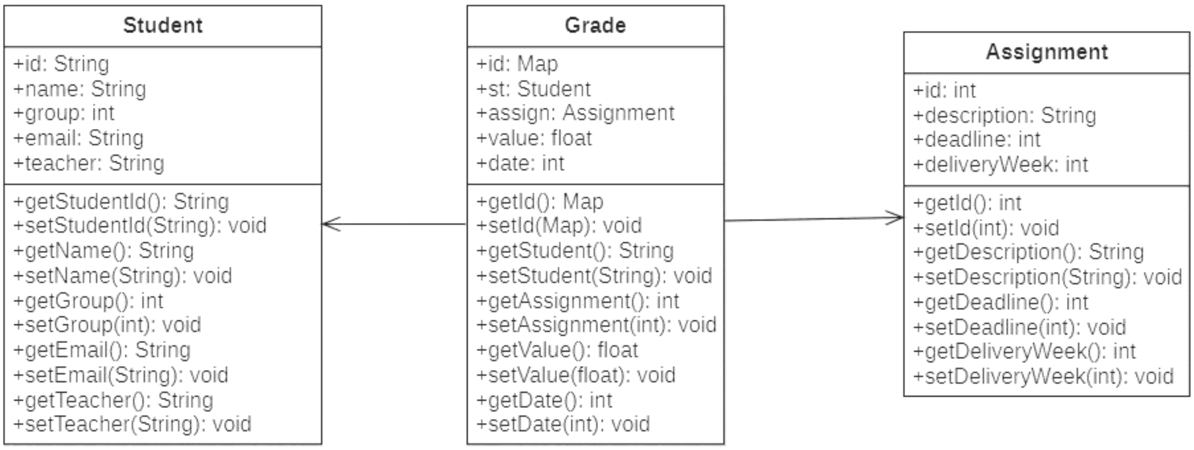
## 4.4 System behavior

### 4.4.1 Use case 1-2-3

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

# 5 Design

## 5.1 Class diagram


Student
+id: String
+name: String
+group: int
+email: String
+teacher: String
+getStudentId(): String
+setStudentId(String): void
+getName(): String
+setName(String): void
+getGroup(): int
+setGroup(int): void
+getEmail(): String
+setEmail(String): void
+getTeacher(): String
+setTeacher(String): void
Grade
+id: Map
+st: Student
+assign: Assignment
+value: float
+date: int
+getId(): Map
+setId(Map): void
+getStudent(): String
+setStudent(String): void
+getAssignment(): int
+setAssignment(int): void
+getValue(): float
+setValue(float): void
+getDate(): int
+setDate(int): void
Assignment
+id: int
+description: String
+deadline: int
+deliveryWeek: int
+getId(): int
+setId(int): void
+getDescription(): String
+setDescription(String): void
+getDeadline(): int
+setDeadline(int): void
+getDeliveryWeek(): int
+setDeliveryWeek(int): void