Laboratory Management

Student: Oltean Victor

**Group: 30431**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 3

5. Class Design 3

6. Data Model 3

7. System Testing 3

8. Bibliography 3

1. Requirements Analysis

# Assignment Specification

The objective of this assignment is to allow students to become familiar with MVC architectural pattern, services, repository and unit tests.

# Functional Requirements

*A Teacher should be able to*

* *CRUD on laboratories*
* *CRUD on students*
* *CRUD on grades for assignments*
* *CRUD on attendance*
* *CRUD on assignments*
* *When a student is created, he receives an email with a token*
* *Login*

*A Student should be able to*

* *Add an assignment*
* *Create an account using the token received by email*
* *See all laboratories*
* *See all assignments for a laboratory*
* *Search for a keyword in assigments*
* *Login*

# Non-functional Requirements

*All requests should load in under 20 ms*

2. Use-Case Model

*Use-Case description format:*

*Use case: teacher*

*Level: user-goal level*

*Primary actor: teacher*

*Main success scenario: The Teacher performs CRUD on almost everything*

*Extensions: CRUD fails because the connection to the database was not properly made*

[Teacher]-(Sign In),

[Teacher]-(CRUD Students),

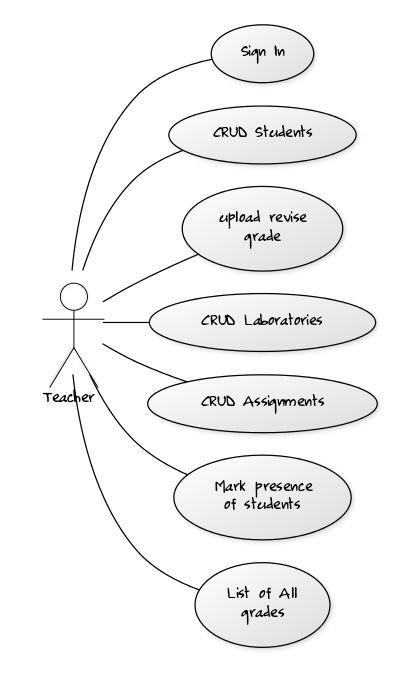
[Teacher]-(upload revise grade),

[Teacher]-(CRUD Laboratories),

[Teacher]-(CRUD Assignments),

[Teacher]-(Mark presence of students),

[Teacher]-(List of All grades)



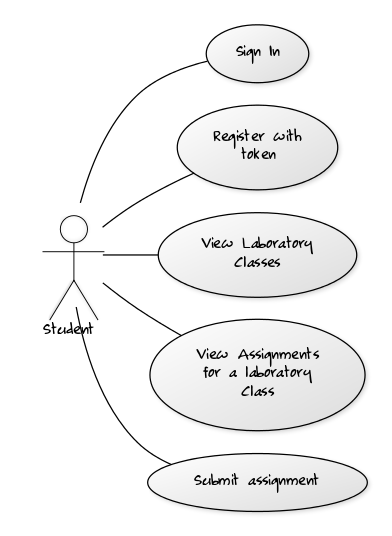
*Use case: student*

*Level: user-goal level*

*Primary actor: student*

*Main success scenario: Student registers and uploads an assignment submission*

*Extensions: Submission fails because the connection to the DB couldn’t be made*



[Student]-(Sign In),

[Student]-(Register with token),

[Student]-(View Laboratory Classes),

[Student]-(View Assignments for a laboratory Class),

[Student]-(Submit assignment)

3. System Architectural Design

**3.1 Architectural Pattern Description**

*MVC*

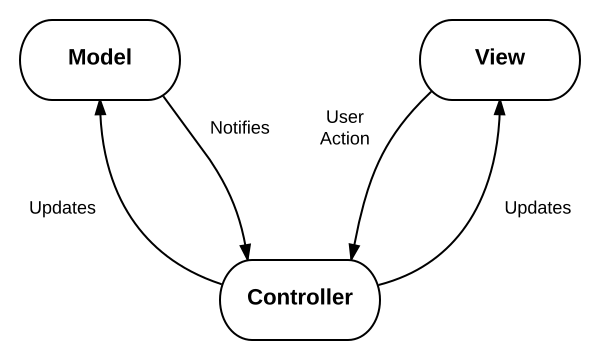
*Model – Here, the model of the application is put, including the business logic and the repository. Also, the main components are represented, eg. a User class with fields username, password etc.*

*Controller – The controller uses the Model in order to update the View, and is being used by the View in order to update the Model. Here, everything which the application can do is put through commands. Our project uses a RESTful API for the controller.*

*View – The Front End of the application. It uses the controller in order to get the data which it is going to display.*

**3.2 Diagrams**

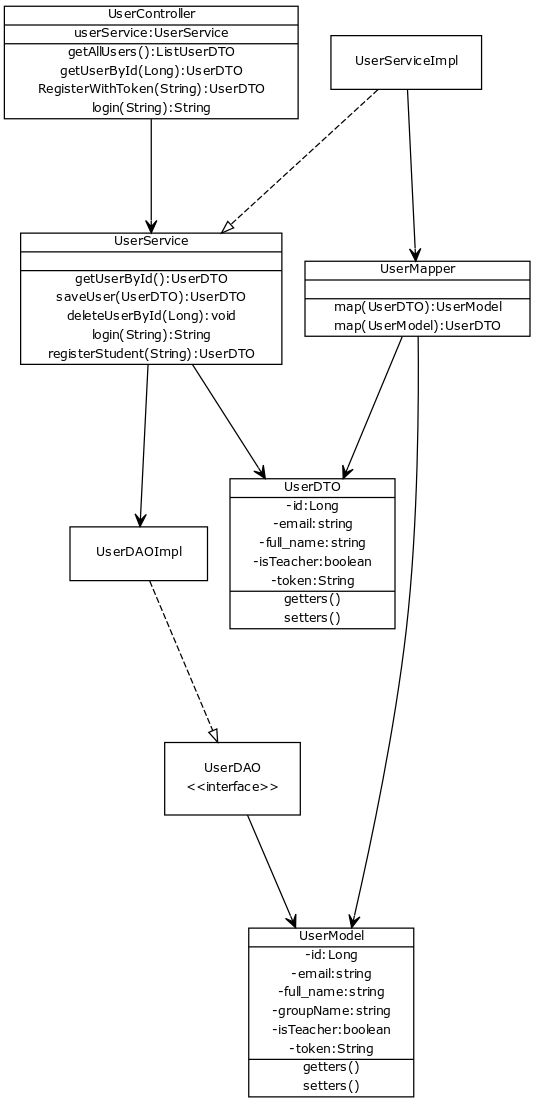
**MVC Diagram**



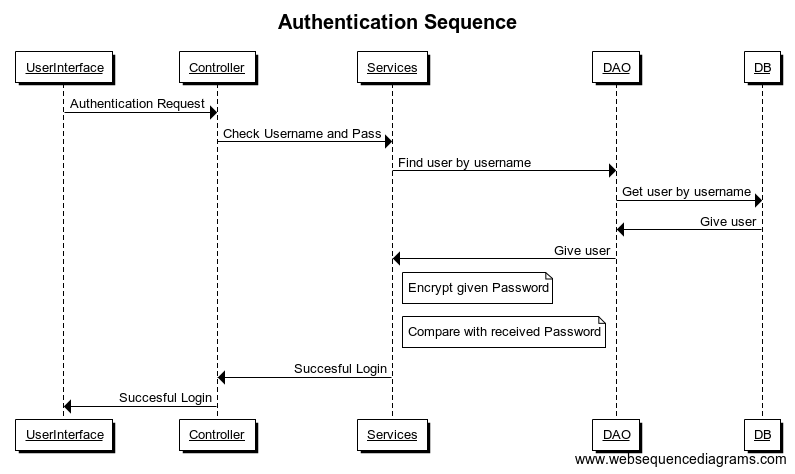
**The BusinessLogic package in the Model package**

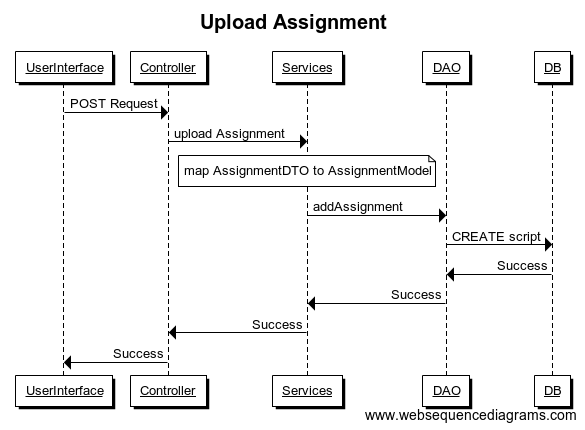
**

**Class diagram for a model**



4. UML Sequence Diagrams





5. Class Design

**5.1 Design Patterns Description**

***Creator****: Assignment Controller uses Assignment Services, therefore it should be the one to create it. However, in order to get a bottom up dependency model, the AssignmentService instance is injected using* ***Dependency Injection*** *into AssigmentController, in order to make the Service dependent on the Controller, not vice-versa, because higher level packages shouldn’t depend on lower level ones.*

6. Data Model

**UserModel:**

**Long iduser;**

**String fullName;**

**String email;**

**String password;**

**String groupName;**

**String hobby;**

**boolean teacher;**

**String token;**

**LaboratoryModel:**

**Long idlaboratory;**

**Long labNb;**

**Date dateOfLab;**

**String title;**

**String curricula;**

**String longDescription;**

**AttendanceModel:**

**Long idattendance;**

**int bonusPoints;**

**Long iduser;**

**Long idlaboratory;**

**AssignmentModel:**

**Long idassignment;**

**String name;**

**Date deadline;**

**String longDescription;**

**Long idlaboratory;**

**SubmissionModel:**

**Long idassignuser;**

**int grade;**

**String gitrepo;**

**String remark;**

**Long iduser;**

**Long idassignment***;*

7. System Testing

Testing has been made using unit testing on one of the APIs, the one for LaboratoryController, by creating and then deleting a laboratory.

8. Bibliography

stackoverflow

spring.io