Software Design Laboratory Tracker

Student: Antonescu Maria-Cristina

**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

In this assignment, the task is developing a Laboratory Activity Tracker Application. The application will be designed to track the laboratory activity for the Software Design laboratory. The application will have two types of users - students and teachers. Teachers will be able to create, read, update, and delete student records, add/edit/delete laboratory classes, CRUD on attendance for each lab, CRUD on assignments, and grade the submitted assignments individually. Students will be able to register using a token generated by the teacher, log in with their username and password, view a list of laboratory classes, view the assignments for a laboratory class, and create an assignment submission by providing a link to a git repository and a short comment for the teacher.

# Functional Requirements

*Authentication and Authorization*

* *The application should allow users to log in with their email address and password.*
* *The application should have two types of users - students and teachers.*
* *Only authenticated users should be allowed to access the application features.*

*CRUD on Students*

* *Teachers should be able to create, read, update, and delete student records.*
* *When a new student record is created, a 128-character token should be generated.*
* *The student should be able to register using the token.*

*Add/Edit/Delete Laboratory Classes*

* *Teachers should be able to add/edit/delete laboratory classes.*
* *For each class, the application should track the laboratory number, date, title, curricula, and a long description with the laboratory text.*

*CRUD on Attendance for each lab*

* *Teachers should be able to create, read, update, and delete attendance records for each lab.*

*CRUD on Assignments*

* *Teachers should be able to create, read, update, and delete assignments for each lab.*
* *For each assignment, the application should track the name, deadline, and a long description with the assignment text.*

*Grade the Submitted Assignments Individually*

* *Teachers should be able to grade the submitted assignments individually.*

*Register with Token*

* *Students should be able to register using the token generated by the teacher.*

*View a List of Laboratory Classes*

* *Students should be able to view a list of laboratory classes.*

*View Assignments for a Laboratory Class*

* *Students should be able to view the assignments for a laboratory class.*

*Create an Assignment Submission*

* *Students should be able to create an assignment submission by providing a link to a git repository and a short comment for the teacher.*

# Non-functional Requirements

1. Security - The application should use secure authentication and authorization mechanisms to ensure only authenticated users have access to the application features.
2. Performance - The application should be able to handle a large number of users and data without any performance issues.
3. Usability - The application should have an easy-to-use interface that is intuitive and easy to navigate.
4. Availability - The application should be available 24/7, with minimal downtime for maintenance.
5. Scalability - The application should be scalable to handle an increasing number of users and data.
6. Reliability - The application should be reliable and robust, with a low risk of failure or downtime.
7. Maintainability - The application should be easy to maintain and update, with clear documentation and code structure.

2. Use-Case Model

*Use case: Add new laboratory class*

*Level: User-goal level*

*Primary actor: Teacher*

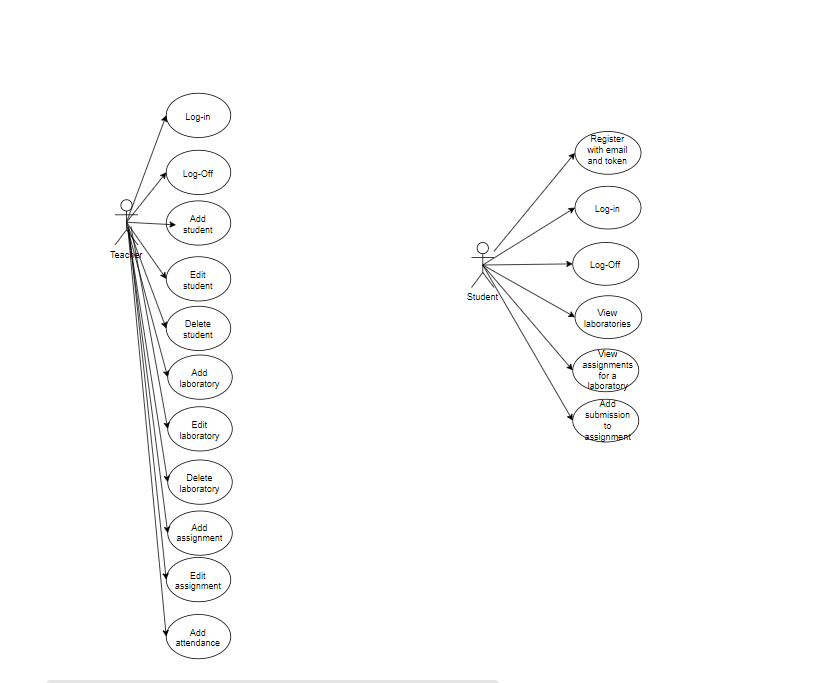
*Main success scenario:*

1. *The Teacher logs in to the application.*
2. *The Teacher selects "Add new class" from the menu.*
3. *The Teacher enters the laboratory number, date, title, curricula, and a long description with the laboratory text.*
4. *The application verifies that the laboratory number is unique.*
5. *The Teacher confirms the details are correct and submits the form.*
6. *The application saves the new laboratory class and displays a success message to the Teacher.*

*Extensions:*

1. *The Teacher logs in to the application.*
2. *The Teacher selects "Add new class" from the menu.*
3. *The Teacher enters the laboratory number, date, title, curricula, and a long description with the laboratory text.*
4. *The application verifies that the laboratory number is already taken and displays an error message to the Teacher.*
5. *The Teacher corrects the laboratory number and repeats the main success scenario.*

*Use case diagrams:*

**

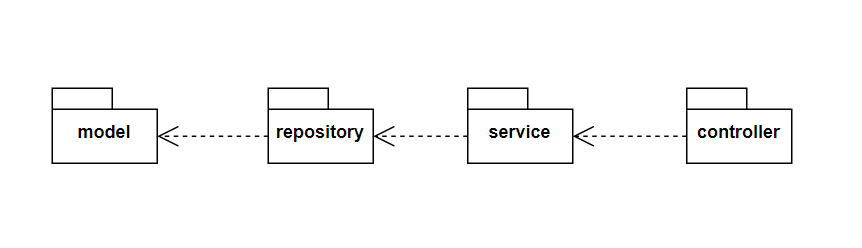
3. System Architectural Design

**3.1 Architectural Pattern Description**

*The application will use the Layers architectural pattern, which separates the application into layers, each with its own set of responsibilities. The layers include the Presentation layer (user interface), Business layer (application logic), and Data Access layer (data storage and retrieval).*

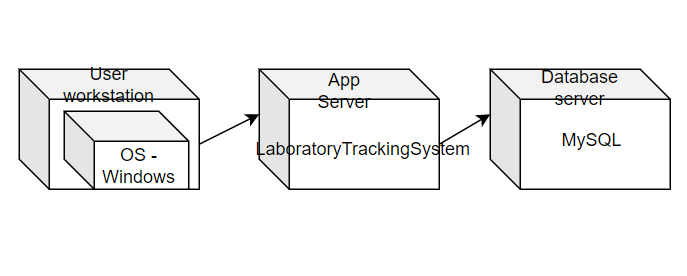
**3.2 Diagrams**

*Package diagram:*

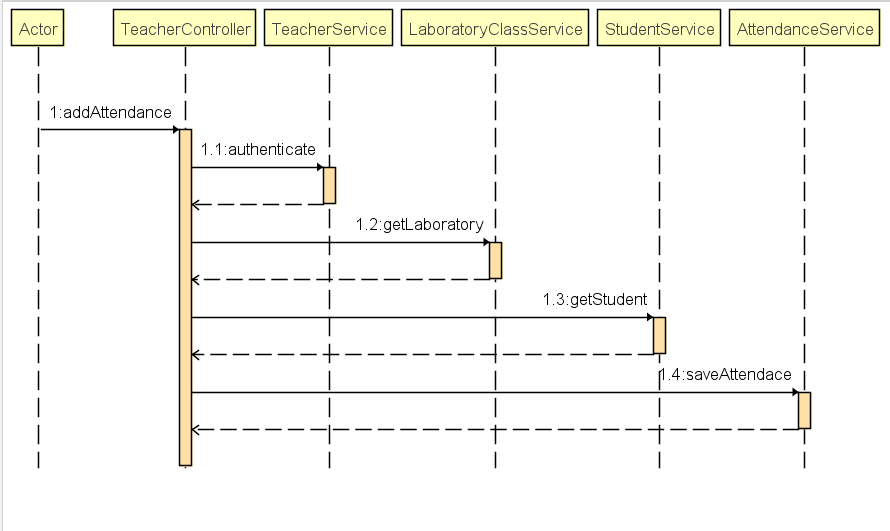
**

*Component diagram:*

*Deployment diagram:*

**

4. UML Sequence Diagrams

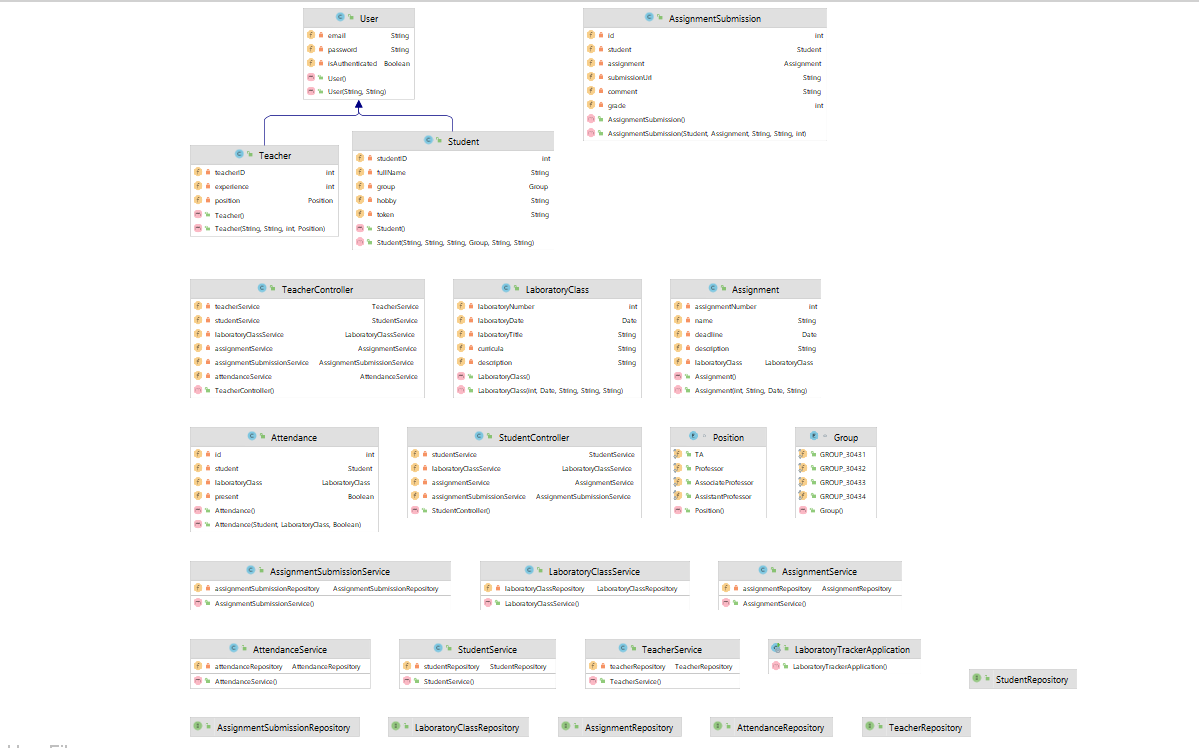


5. Class Design

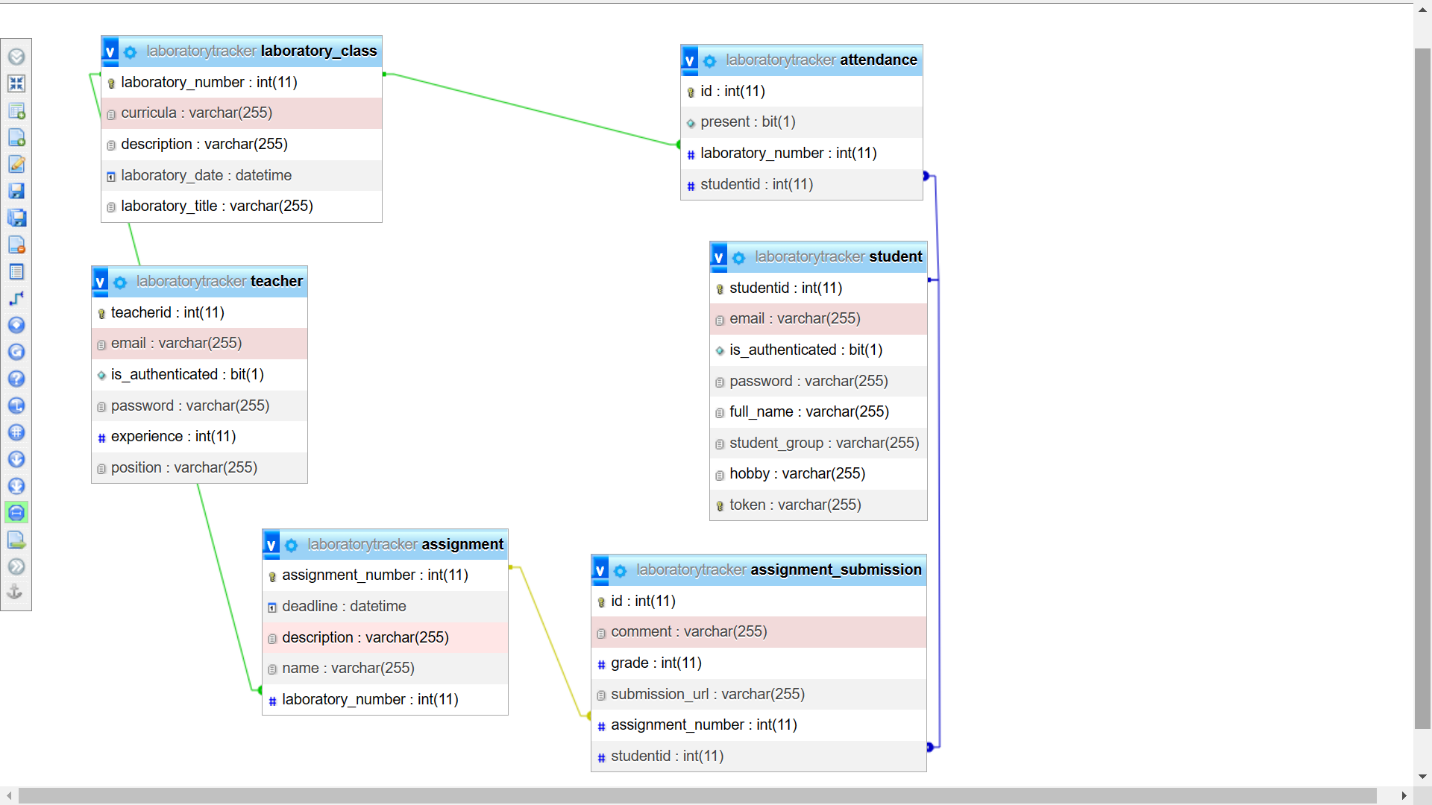
**5.1 Design Patterns Description**

1. *Repository Pattern - We will use this pattern to interact with the database. The repository pattern will allow us to abstract the data access layer from the rest of the application.*
2. *Service Layer Pattern - We will use this pattern to handle the business logic of the application. The service layer pattern will allow us to abstract the business logic from the rest of the application.*

**5.2 UML Class Diagram**

**

6. Data Model

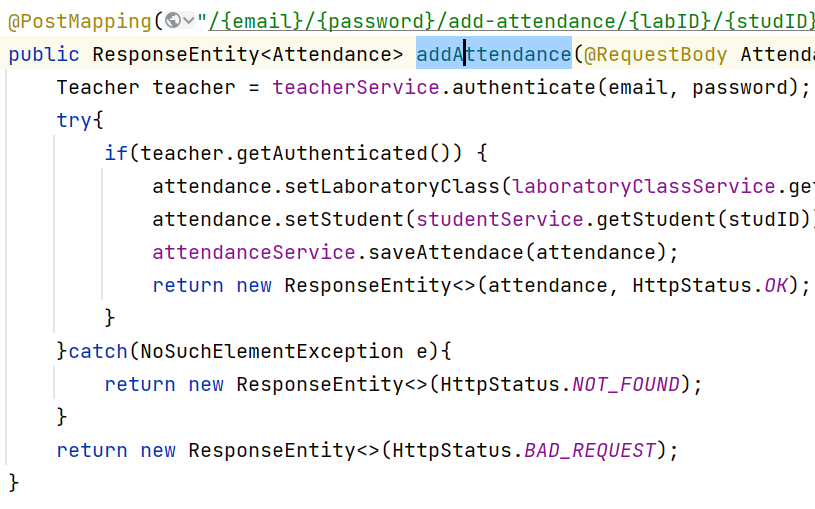
**

*There are 6 entities in the system:*

* *Teacher – is a type of User, has email, password, experience and position;*
* *Student – is the second type of User, which is characterized by username, password, full name, group, hobby and a token;*
* *Attendance – is an entity which records the presence/absence for a student to a lab;*
* *Assignment – is an assignment created by the teacher to a lab, it has a description, name and a deadline;*
* *AssignmentSubmission – is an entity which records a submission of a student for an assignment;*
* *Laboratory\_Class – is created by the Teacher, it records the description, number and curricula of a lab.*

7. System Testing

*For this laboratory tracker system, I tested the requests in Postman. I returned a ResponseEntity to make sure that my info is update correctly. I also return HTTP codes to make testing much easier. Here is how a class from the controller looks:*

**

8. Bibliography