Student: Crisan Oana Andra

**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 application is a Lab Class Management System that allows teachers to create, manage and grade lab classes. Students can view their class schedules and submit their lab reports to the teacher.

# Functional Requirements

Teachers should be able to login.

Teachers should be able to create, edit and delete a Lab Class with a unique number, date, title, curricula, and lab text.

Teachers should be able to add, delete, and edit students from a Lab Class.

Teachers should be able to view a list of students who attended a Lab Class.

Teachers should be able to grade lab submissions.

Students should be able to view their Lab Class schedule.

Students should be able to submit lab reports to the teacher.

Students should be able to register with a token given by the teacher.

# Non-functional Requirements

*-* Data will be stored in a relational database.

- Use the Layers architectural pattern to organize your application.

- Passwords must be encrypted when stored to the database with a one-way encryption algorithm (base 64).

- Postman collection of operations

- ORM Hibernate framework to access the database

- API design should be RESTful, not SOAP

2. Use-Case Model

Use case: Create a new Lab Class

Level: User-goal level

Primary actor: Teacher

Main success scenario:

1.The teacher logs into the system.

2.The teacher selects "Create a new Lab Class."

3.The system prompts the teacher to enter the Lab Class details (number, date, title, curricula, and lab text).

4.The teacher enters the Lab Class details.

5.The system saves the Lab Class in the database.

USE CASE DIAGRAM FOR STUDENT:

Diagram

Description automatically generated

USE CASE DIAGRAM FOR TEACHER:

Diagram

Description automatically generated

3. System Architectural Design

**3.1 Architectural Pattern Description**

A layered architecture is a design pattern in which software components are organized into horizontal layers, with each layer having a specific responsibility and communicating only with its adjacent layers. Typically, the layers are ordered from top to bottom, with the top layer being the user interface layer, the middle layer being the business logic layer, and the bottom layer being the data access layer. This architecture promotes modularity, separation of concerns, and maintainability, as changes made to one layer do not affect the other layers.

**3.2 Diagrams**

Package Diagram:

*Diagram

Description automatically generated*

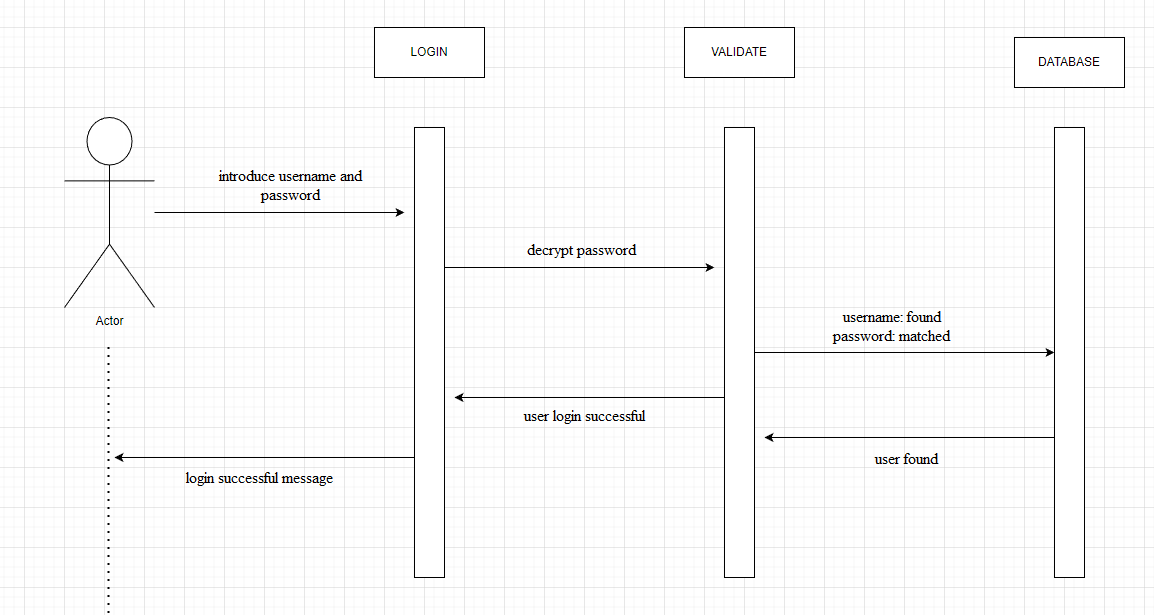
Component Diagram:

Diagram

Description automatically generated

4. UML Sequence Diagrams

Login scenario for teacher:

**

5. Class Design

**5.1 Design Patterns Description**

The Repository Design Pattern is a widely used approach in layered software architectures for managing data access and storage. It follows the principle of encapsulation, where each data entity or object has a corresponding repository that acts as an intermediary between the data source and the rest of the application. The repository abstracts the details of data storage and retrieval from the rest of the application by providing a set of methods for creating, reading, updating, and deleting data.

In a Java application with Spring, the Repository pattern can be implemented using Spring Data. This framework offers a set of interfaces and annotations that simplify data access and storage. Spring Data repositories provide a consistent API for performing CRUD (Create, Read, Update, Delete) operations, and can be easily customized and extended to meet the specific needs of your application.

Using the Repository pattern and Spring Data can provide several benefits for your application, including improved code organization, reduced complexity, and increased maintainability. Additionally, this approach can help you easily switch between different data sources, such as databases or external APIs, without impacting the rest of your application.

**5.2 UML Class Diagram**

Graphical user interface

Description automatically generated with medium confidence

6. Data Model

The data models used in this application are: Student, Teacher, LabClass, Assignment and Submission:

*A picture containing text

Description automatically generated* *Graphical user interface

Description automatically generated* Graphical user interface

Description automatically generated Graphical user interface

Description automatically generated with medium confidence Graphical user interface, application

Description automatically generated

7. System Testing

All the operations can be tested using Postman and Datagrip to check in the Database. Postman is a popular tool for API development and testing. It allows users to make HTTP requests to RESTful APIs and provides features such as request parameterization, response validation, and automated testing. With Postman, users can create collections of requests, organize and share them with other team members, and run tests on each request to ensure the API is working as expected. It also provides a friendly user interface that makes it easy to inspect and debug requests and responses. Postman supports a wide range of HTTP request methods, including GET, POST, PUT, DELETE, PATCH, and more, and it can be integrated with many popular development tools and services.

In combination, DataGrip and PostgreSQL provide a powerful toolset for developers to work with relational databases. DataGrip provides an intuitive interface for managing PostgreSQL databases, while PostgreSQL provides a powerful and scalable database management system. Together, they enable developers to build robust and scalable web applications that can handle large volumes of data.

8. Bibliography

<https://github.com/CristinaMadalinaMihai/UTCNSoftwareDesignLaboratory/blob/main/Assignments/Assignment2_2023.pdf>

* PT courses last year

https://www.youtube.com/watch?v=Gx4iBLKLVHk&t=2983s