<Assignment Name>
Analysis and Design Document

Student: Neagoi Mihai
Group: 30431

Table of Contents

1. Requirements Analysis

1.1 Assignment Specification

Design and implement an application for the tracking the laboratory activity for the Software Design laboratory. The application should have two types of users (student and teacher) which must provide an email and a password to use the application.

1.2 Functional Requirements

- Student can authenticate
- Student can submit assignments
- Student can view laboratories
- Student can view assignments
- Student can register using the token generated by the teacher
- Teacher can authenticate
- Teacher can CRUD Students
- Teacher can CRUD Laboratories
- Teacher can CRUD Attendance on laboratories
- Teacher can CRUD Assignments on laboratories
- Teacher can grade Submissions

1.3 Non-functional Requirements

- *The system should be fast and responsive*
- *The system should be secure and require authentication and authorization*

2. Use-Case Model

Use-Case description format: authenticate user
Use case: authentication
Level: user-goal level
*Prim**ary actor: any user***
Main success scenario:

1. ***Make a JSON for the request with email and password***
2. ***Send POST request to users/sign_in endpoint***
3. ***Store received cookie***

Use-Case description format: delete a student
Use case: delete student
Level: user-goal level
*Prim**ary actor: teacher***
Main success scenario:

1. ***Login as teacher***
2. ***Fill header with Cookie: _session_id:<received cookie upon login>***
3. ***Send DELETE request to /student/ <student_id> endpoint***

Use-Case description format: edit a laboratory
Use case: edit laboratory
Level: user-goal level
*Prim**ary actor: teacher***
Main success scenario:

1. ***Login as teacher***
2. ***Fill body of request with a JSON format -> "title": <title of laboratory>,***

"laboratory_number": <laboraotry_number>,
"description": <lab_description>,
"start_date": <start_date>

3. ***Fill header with Cookie: _session_id:<received cookie upon login>***
4. ***Send PUT request to /laboratories/ <laboratory_id> endpoint***

Use-Case description format: see a assignments
Use case: edit laboratory
Level: user-goal level
*Prim**ary actor: teacher***
Main success scenario:

1. ***Login as student***
2. ***Fill header with Cookie: _session_id:<received cookie upon login>***
3. ***Send get request to /assignments endpoint***

3. System Architectural Design

3.1 Architectural Pattern Description

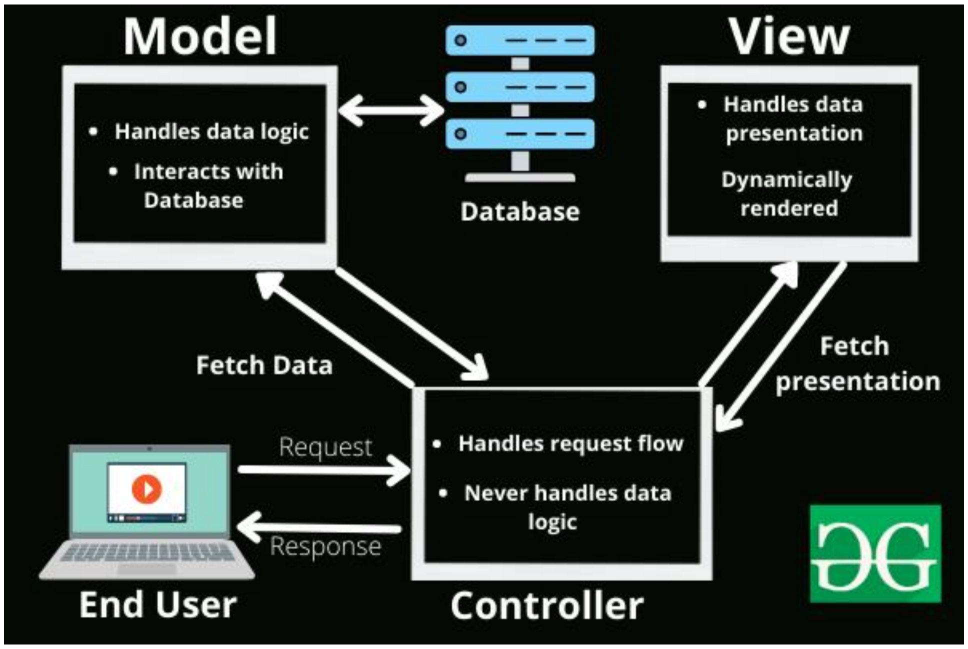
MVC stands for Model-View-Controller, and it is a software architecture pattern that separates an application into three interconnected components: the model, the view, and the controller.

The Model is responsible for managing the data and business logic of the application. It represents the underlying data and provides methods to access and modify that data. The model is independent of the user interface and is designed to be reusable.

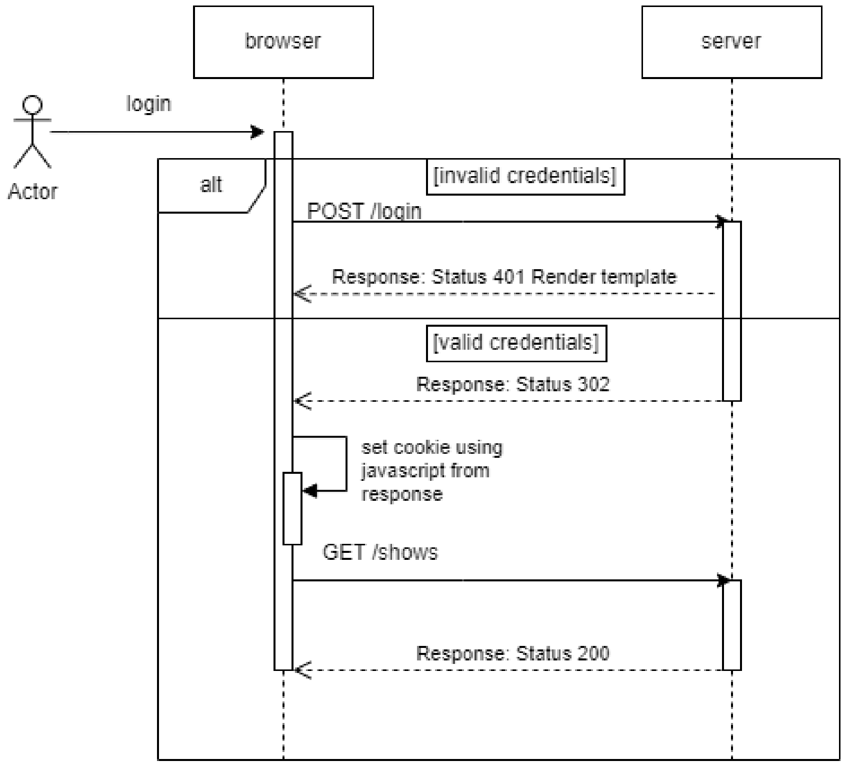
The View is responsible for presenting the data to the user. It represents the user interface and is designed to be user-friendly. The view displays the data that is provided by the model.

The Controller acts as an intermediary between the Model and View components. It handles user input and updates the model and view accordingly. The controller is responsible for interpreting user actions and deciding what to do with them. It updates the model based on the user's input and notifies the view to update the UI accordingly.

3.2 Diagrams



4. UML Sequence Diagrams

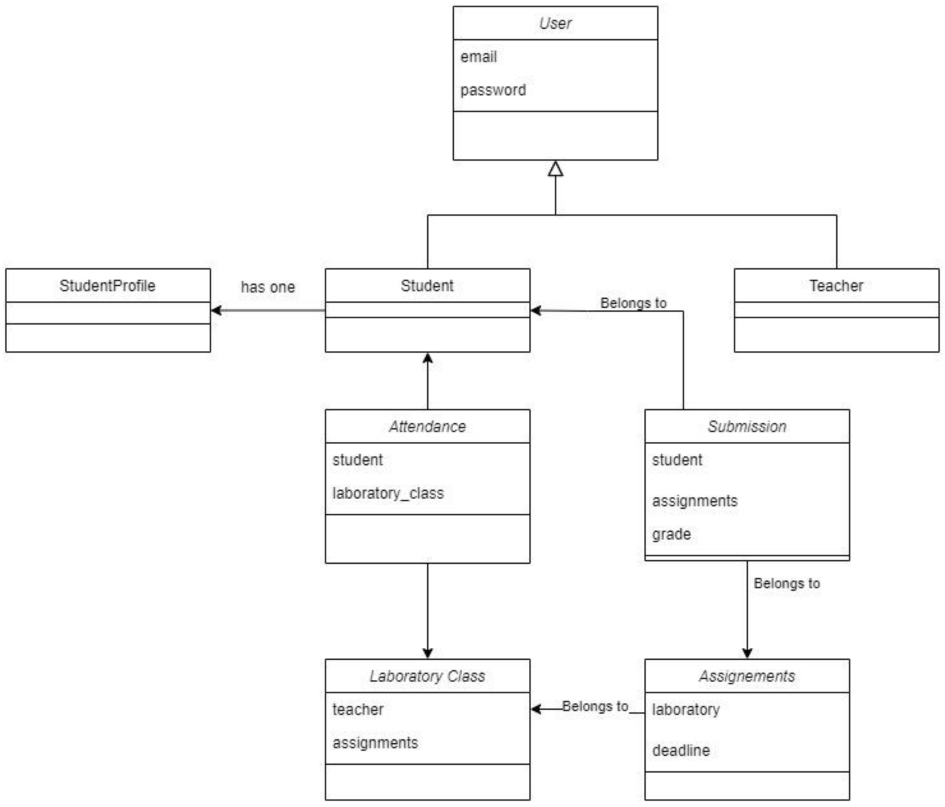


5. Class Design

5.1 Design Patterns Description

The factory pattern is a creational design pattern that provides an interface for creating objects in a superclass and allows subclasses to alter the type of objects that are instantiated.

5.2 UML Class Diagram



6. Data Model

The system implementes 8 data models: Teacher, Student, Submission, User, StudentProfile, LaboratoryClass, Attendance, Assignment.

7. System Testing

For unit testing I used RSpec and to generate mock data I used FactoryGirl. To test the application manually I used postman.

8. Bibliography

- https://api.rubyonrails.org/
- https://rspec.info/