<Online Math Learning Platform>

Project documentation

Student:

* Francesca Dițulescu

**Group: 30431**

**Contents**

[I Project specification](#_heading=h.gjdgxs) **3**

[1.1 Domain Model Diagram](#_heading=h.30j0zll) 3

[II Use-Case model](#_heading=h.1fob9te) **3**

[2.1 Users and stakeholders](#_heading=h.f2irg1azp7rm) 4

[2.2 Use-Case identification](#_heading=h.3znysh7) 4

[2.3 UML Use-Case diagram](#_heading=h.2et92p0) 4

[III Architectural design](#_heading=h.tyjcwt) **5**

[3.1 Conceptual architecture](#_heading=h.3dy6vkm) 5

[3.2 Package diagram](#_heading=h.1t3h5sf) 5

[3.3 Class diagram](#_heading=h.4d34og8) 5

[3.4 Database (E-R/Data model) diagram](#_heading=h.2s8eyo1) 6

[3.5 Sequence diagram](#_heading=h.17dp8vu) 6

[3.6 Activity diagram](#_heading=h.3rdcrjn) 6

[IV Supplementary specifications](#_heading=h.26in1rg) **6**

[4.1 Non-functional requirements](#_heading=h.lnxbz9) 6

[4.2 Design constraints](#_heading=h.35nkun2) 6

[V Testing](#_heading=h.1ksv4uv) **7**

[5.1 Testing methods/frameworks](#_heading=h.44sinio) 7

[5.2 Future improvements](#_heading=h.2jxsxqh) 7

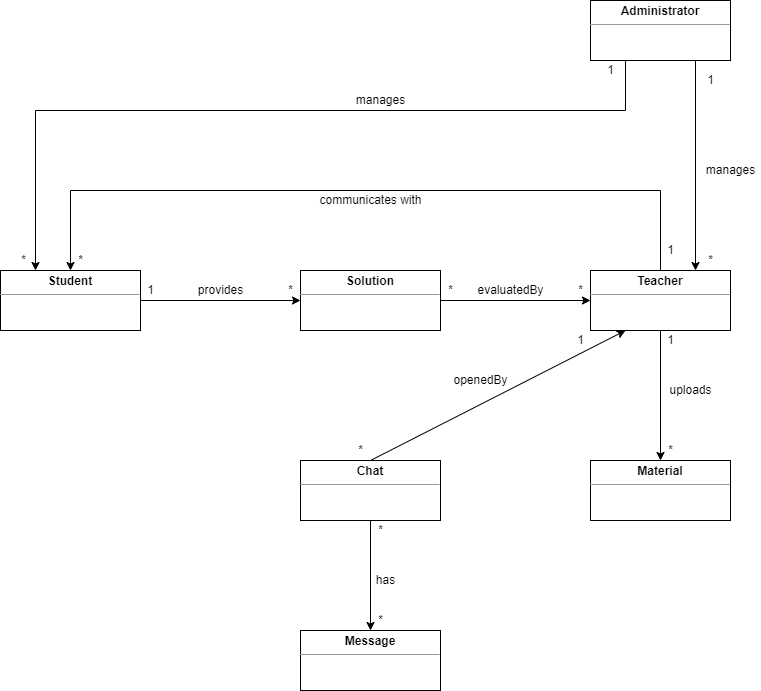
[VI Bibliography](#_heading=h.z337ya) **7**

# I Project specification

This application comes as a support for the people that want to learn math at their own pace in a relaxing and non-competitive environment. It provides students with materials written and uploaded by experts (teachers) and different kinds of exercises - single-choice answers, multiple choice answers and with fully written solutions. After a written solution is uploaded, one of the teachers takes that solution to analyze and, after checking the student’s approach, it proves feedback. Feedback is a mandatory requirement from teachers.

The process of feedback is supported by a communication chat that is opened by the teacher after he/she has gone through the student’s solution and is ready to discuss everything that is unclear to the student.

## 1.1 Domain Model Diagram



# II Use-Case model

## 2.1 Users and stakeholders

The application will have 2 types of users, each with a different role – Student and Teacher.

Students are the ones that see the theory, solve exercises, receives feedback, and communicate with teachers.

Teachers are the ones that upload materials, provide feedback, and communicate with students.

As stakeholders, we can consider that this application is useful for people of all ages who wants to learn math (mostly students in university and in high school) and for the teachers that want to help other students grow and learn. In addition, we can also consider as stakeholders the developers that work on building similar applications.

## 2.2 Use-Case identification

Use case name: Solve single-choice exercises

Level: Student-Goal

Main actor: Student

Main success scenario:

* Student accesses the exercises section for the current theory material through a button
* Student views a list of categories of exercises (single-choice, multiple-choice, fully written solutions)
* Student selects the single-choice category
* Student views a list of single-choice exercises
* Student inserts its answer in a specific box
* Student checks its answer

Extension: The application colours the result in green if the answer is correct and red otherwise. If red, the Student inserts another answer in the specific box and checks its answer again.

Use case name: Send fully written solutions

Level: Student-Goal

Main actor: Student

Main success scenario:

* Student accesses the exercises section for the current theory material through a button
* Student views a list of categories of exercises (single-choice, multiple-choice, fully written solutions)
* Student selects the fully written solutions category
* Student views a list of problems
* Student selects one problem from the list
* Student views the problem’s description, an UPLOAD SOLUTION button and a SEND BUTTON
* Student selects the button
* Student uploads a file selected from its device
* Student presses the SEND SOLUTION button to send the solution for review

Extension: Student presses the SEND SOLUTION button without uploading a material. A message will be printed on the screen warning him that no file was updated. Student can press the upload button again to place a file and try to send a solution again.

Use case name: Open chat

Level: Teacher-Goal

Main actor: Teacher

Main success scenario:

* Teacher logs in the application
* Teacher can view notifications about students that have uploaded solutions
* Teacher clicks on a notification
* Teacher is redirected to a page from where he can download the solution and has a SEND FEEDBACK button
* Teacher presses the SEND FEEDBACK button which opens the chat with the student
* Teacher continues to talk there with the student in case of a reply

Use case name: upload materials

Level: Teacher-Goal

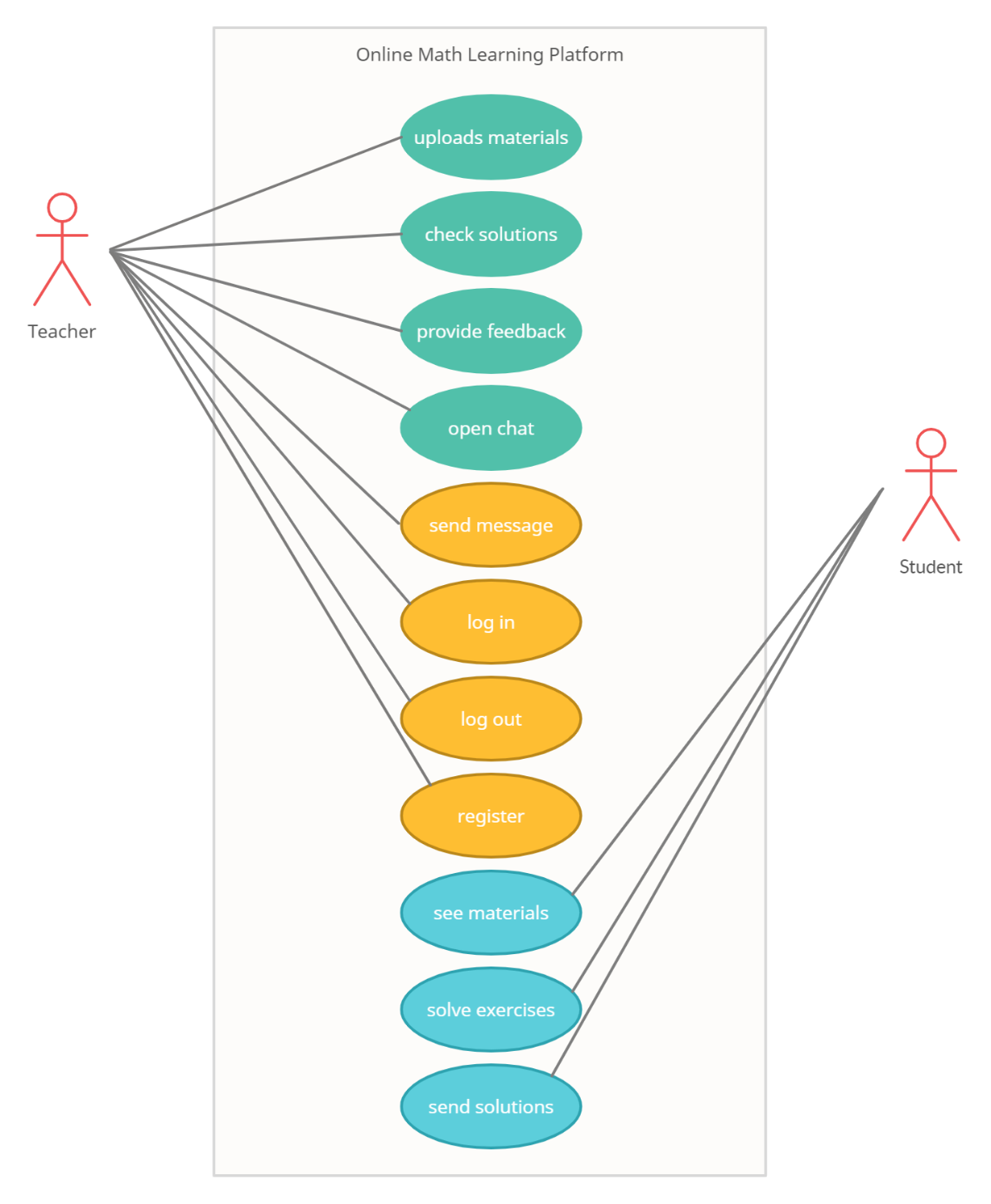
Main actor: Teacher

Main success scenario:

* Teacher logs in the application
* Teacher checks the menu
* Teacher selects the UPLOAD MATERIALS section
* Teacher views a new window where he/she has to insert details about the material (section, name, description), followed by an UPLOAD MATERIAL button
* Teacher selects the UPLOAD MATERIAL button

Extension: Teacher does not upload any material. In this case, the teacher will be notified there has been no material uploaded and he can try to upload again.

## 2.3 UML Use-Case diagram



# III Architectural design

## 3.1 Conceptual architecture

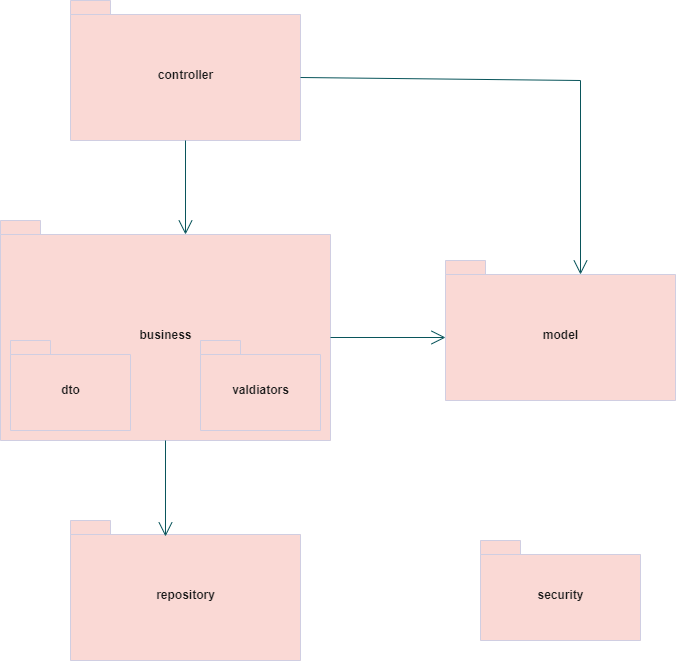
Type of application: WEB

Architecture: Layered architecture

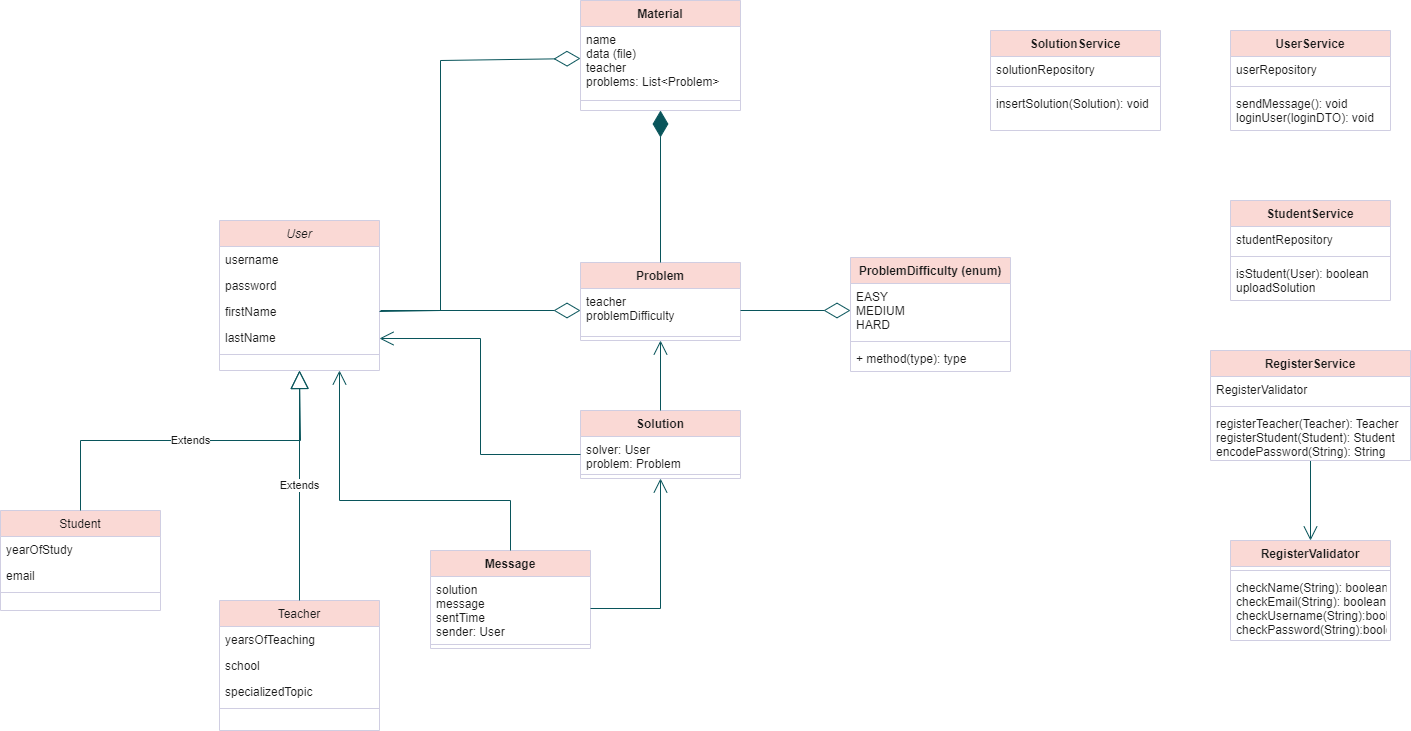
Database information: it will consist of minimum 3 tables – one for teachers, one for students and one for materials. In addition to those 3 tables, I am thinking about how to store the messages that are sent by both users in a chat. I will create a table for each conversation that happens between 2 users and store a message/record.

Why layered architecture? Because I mostly have CRUD operations on the Teacher and a few on the Student use cases. In addition, it is an architecture I am already familiar with and it helps me divide my business logic, the database access, the models, etc.

## 3.2 Package diagram

**

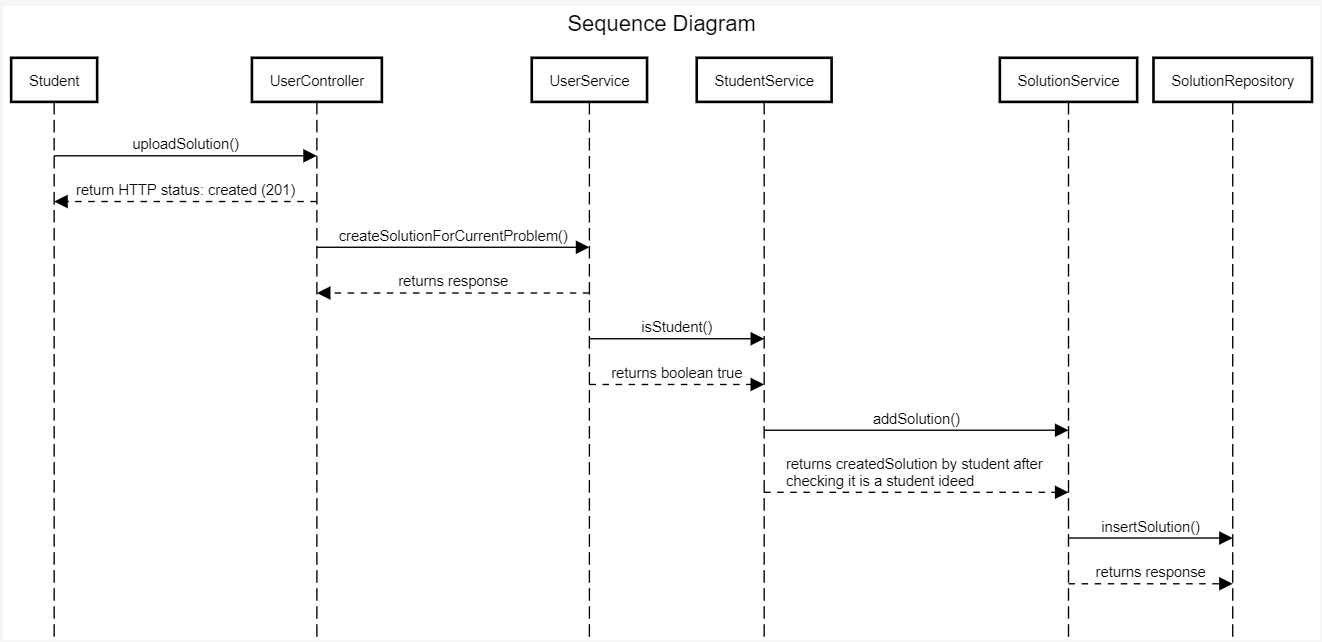
## 3.3 Class diagram

**

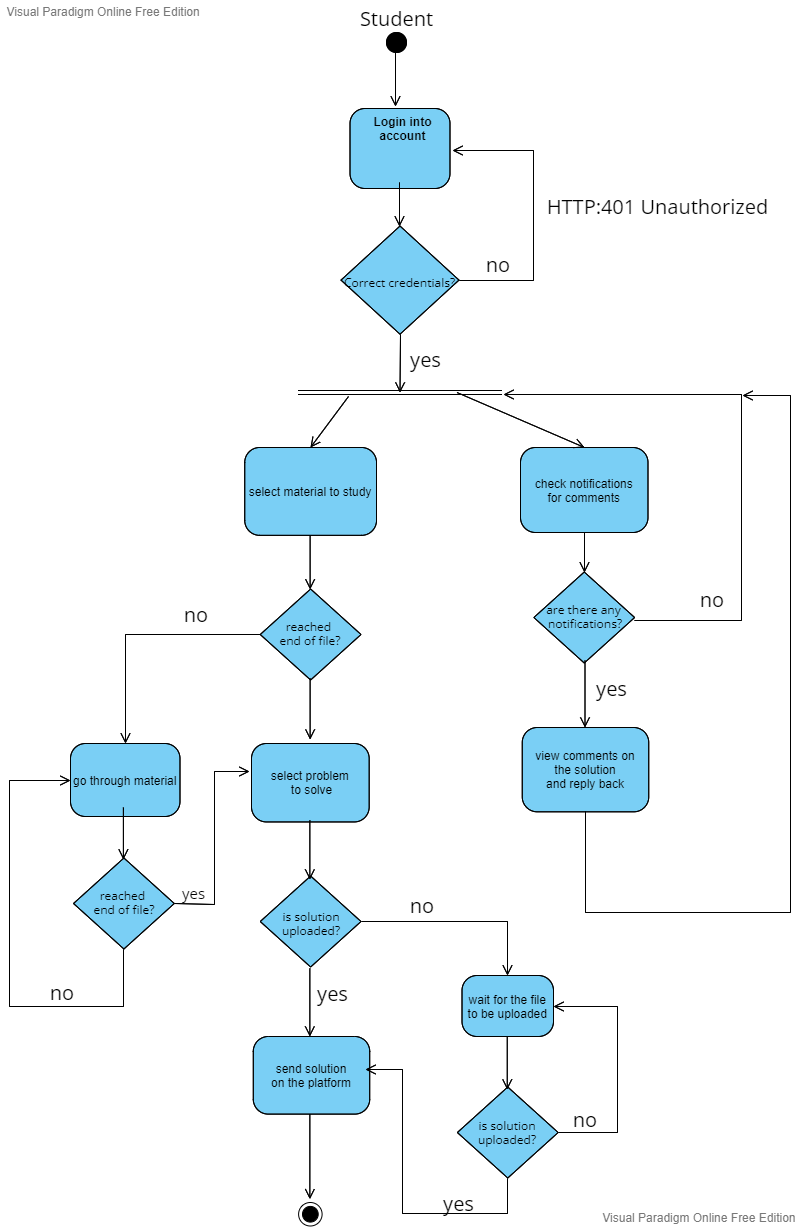
## 3.4 Database (E-R/Data model) diagram

## *O imagine care conține text, captură de ecran, interior, computer Descriere generată automat*

## 3.5 Sequence diagram



## 3.6 Activity diagram



# IV Supplementary specifications

*< Se va scrie o mica introducere./>*

## 4.1 Non-functional requirements

Availability: it can be accessed all the time

Scalability: should be able to support a big community of teachers and students

Performance: maximum upload file size (for the students’ solutions)

Accessibility across devices: mobile and desktop (since it is a web application)

## 4.2 Design constraints

Programming Language used: Java

Framework used: Spring

Security: password encoded in the database and unique usernames

Database: mySQL and possibly noSQL

Note: there will be no visitor user – everyone has to create an account before logging in the application (they are either students or teachers)

# V Testing

*< Se va discuta la laborator./>*

## 5.1 Testing methods/frameworks

## 5.2 Future improvements

# VI Bibliography