

Polytechnic Institute of Cávado and Ave

Multimedia and Web Technology

Web-User-Manager

Course leader:

Eva Ferreira Oliveira

Students:

Romano Keser

Laurenz Gebauer

Academic year 2020/2021

June 2021

CONTENT

Table of Contents

1. Brief introduction.....	3
2. User interface	4
3. Students.....	7
Use Case:.....	9
Tools Used :.....	11

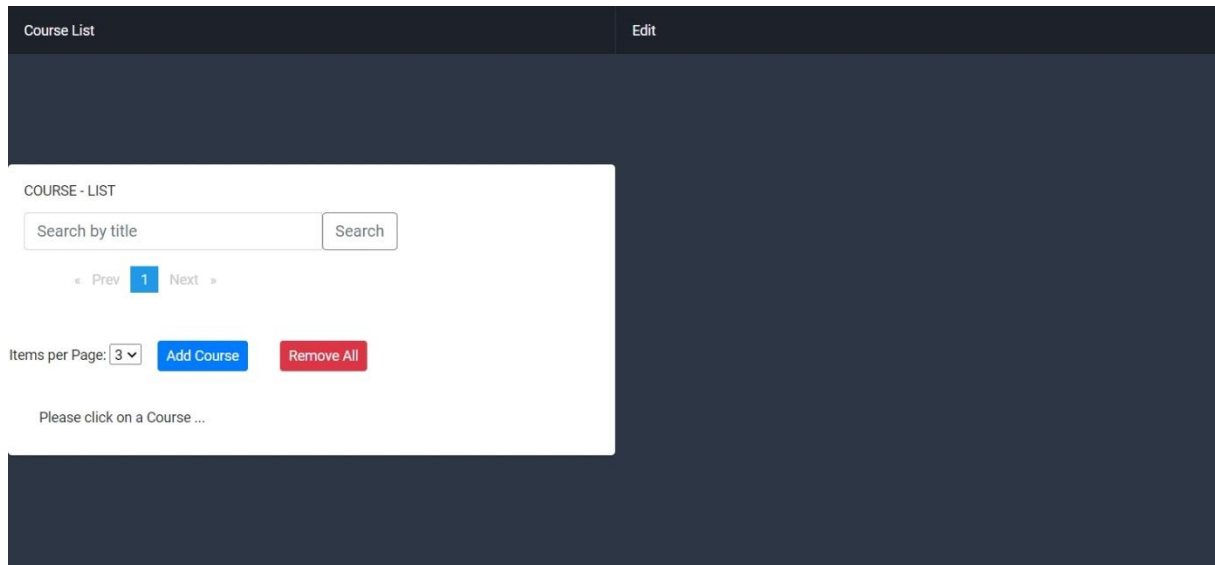
1. Brief introduction

The project for the subject Multimedia and Web Technology we made is called Web-User-Manager. It is made using Angular and Node.js. The whole platform is connected to the MongoDB database we separately made. The app allows the admin to add classes and students as well as edit a list of students and subjects. It is also possible to add and edit student's grades for each subject and notes. In this report, we will go through everything that we have done and what technologies we used.

The first section of the report shows every function and how the app can be used, step by step. From adding the new course to editing the students and grades.

2. User interface

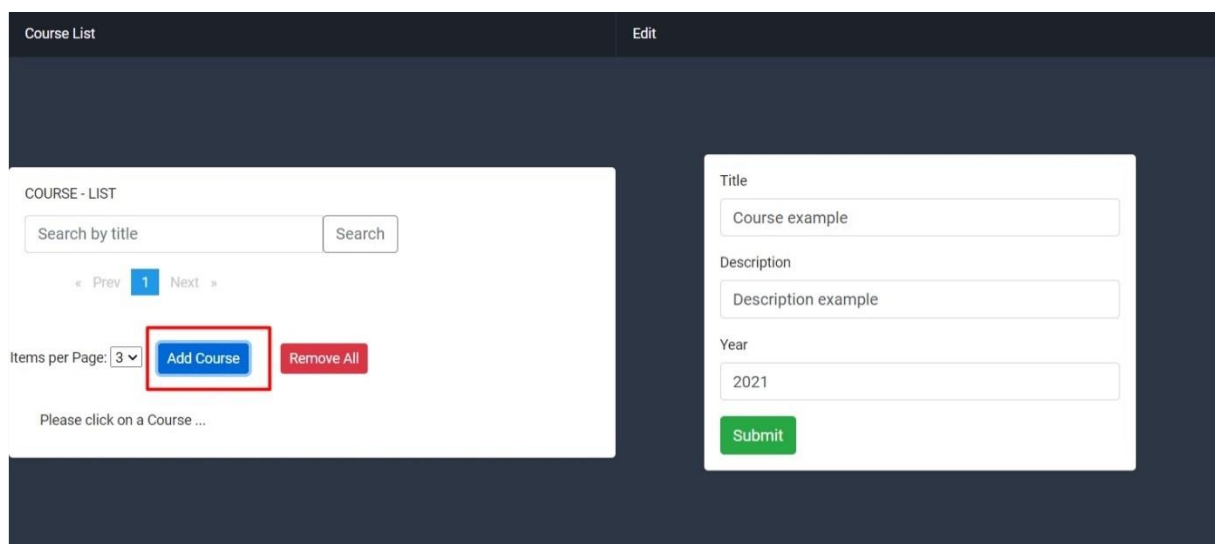
The first page contains three different blocks. Course list which lists all the courses and their info. Edit section for editing the students, their grades and the subjects.



The screenshot shows a web interface with a dark blue header. The header has two tabs: "Course List" (active) and "Edit". Below the header, there is a white sidebar on the left and a large dark blue main area on the right. The sidebar contains the following elements: a "COURSE - LIST" section with a search bar labeled "Search by title" and a "Search" button; pagination controls showing "« Prev 1 Next »"; "Items per Page: 3" with a dropdown arrow; a blue "Add Course" button; a red "Remove All" button; and a prompt "Please click on a Course ...".

Picture 1) Course list and edit section

The *edit* section activates when the admin wants to add/edit either student's grade or the subjects.



The screenshot shows the same web interface as Picture 1, but the "Edit" tab is active. The sidebar remains the same. The main area now features a white form for adding a new course. The form has the following fields: "Title" with the value "Course example"; "Description" with the value "Description example"; "Year" with the value "2021"; and a green "Submit" button.

Picture 2) Adding new course

Course List

COURSE - LIST

« Prev **1** Next »

Course example

Course example 2

Items per Page: ▼

Class

Title: Course example 2

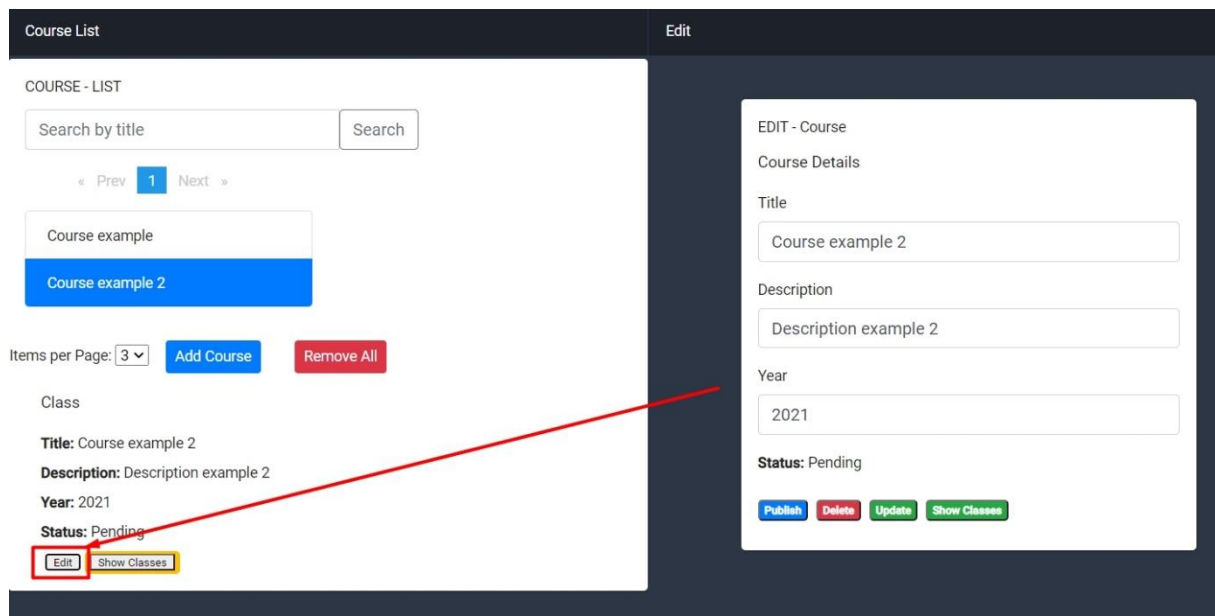
Description: Description example 2

Year: 2021

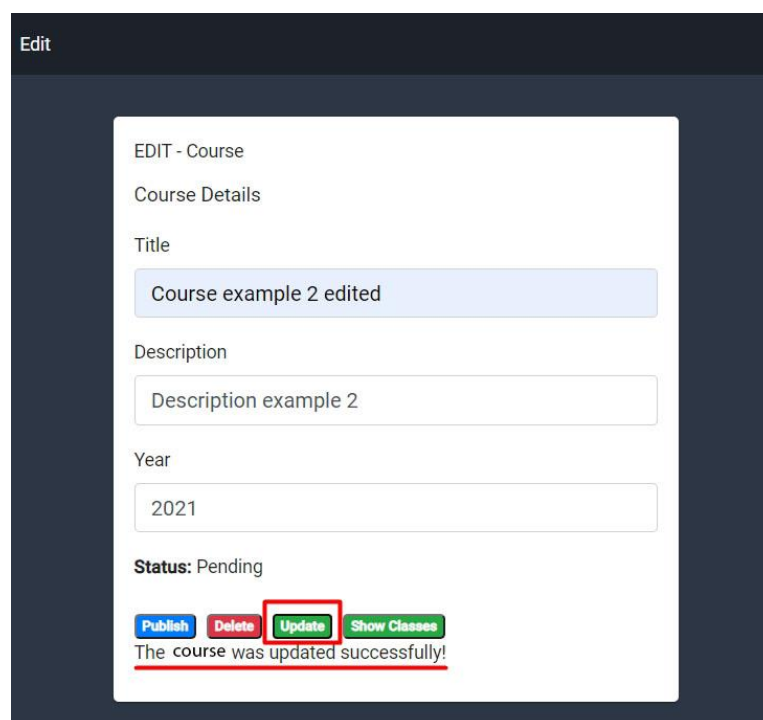
Status: Pending

Picture 1) New course added to the list

Now when the new course is added to the list we can edit the selected course or delete the course by clicking the *edit* button. The buttons *Add Course* and *Remove All* are present the whole time on the first page. *Items per Page* refers to the listing of all subjects added.



Picture 2) Edit course section

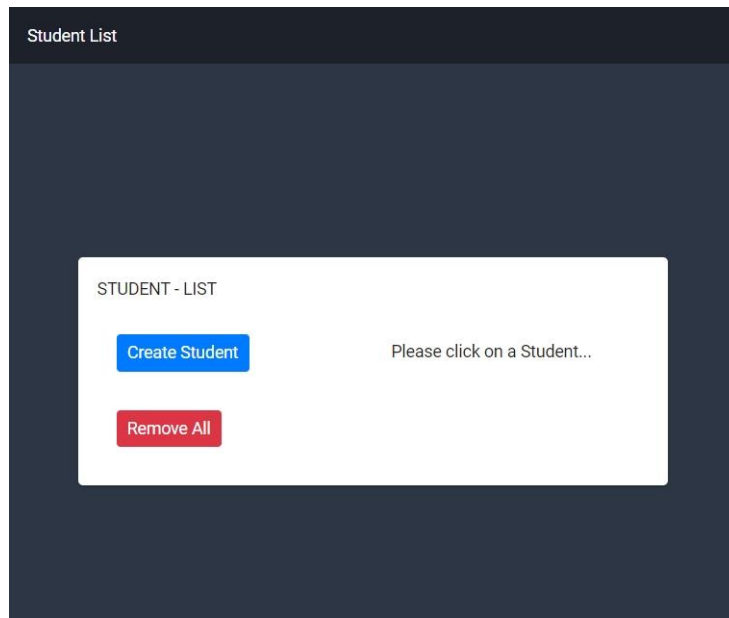


Picture 3) Editing the course

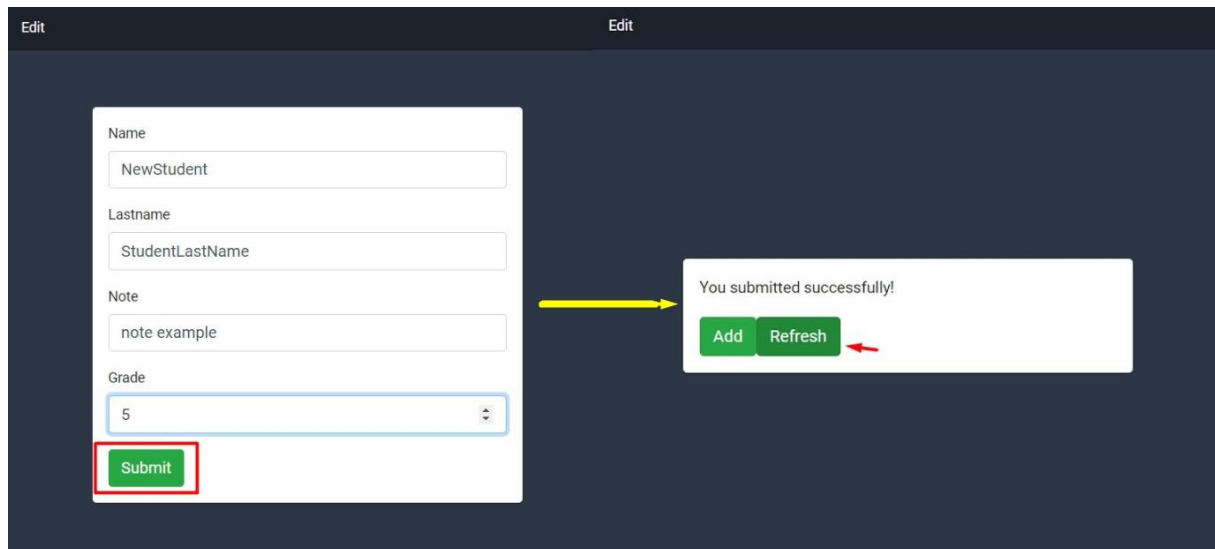
For editing the course admin has to select the course and the *edit* button opens the edit section. The title, description and year can be edited. After the changes are made the *publish* button is clicked and the notification caption shows if it was successfully edited or not.

3. Students

A similar logic is for the student's list. We can add/remove/edit students. The student's section stands below the *course* section and the same *edit* box opens if the user wants to edit or delete the student.

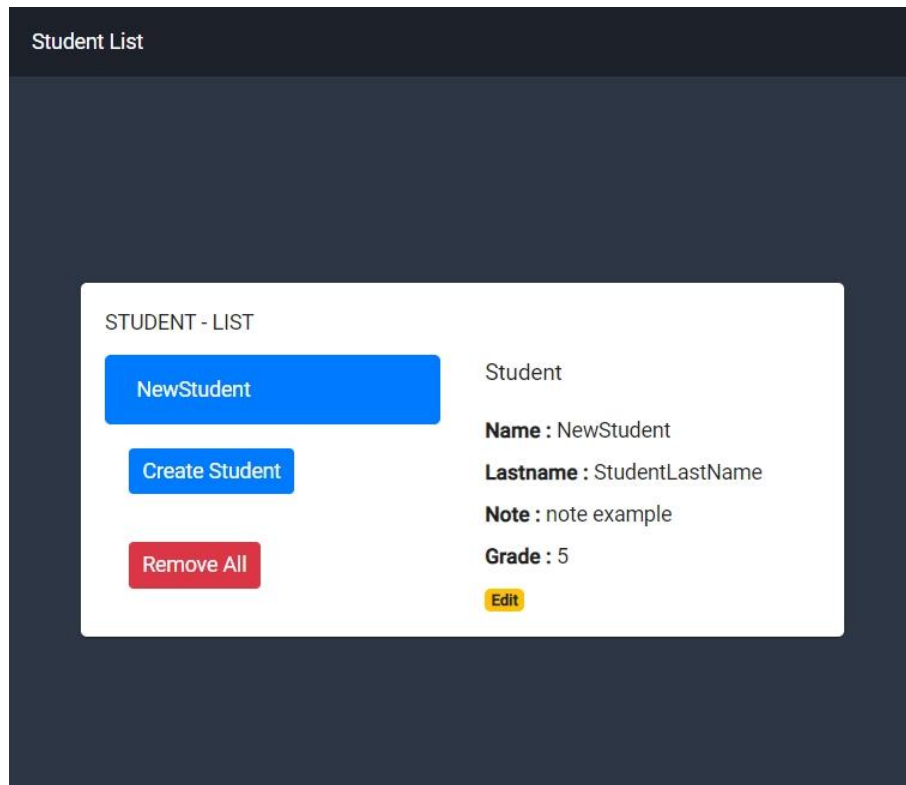


Picture 4) Student list



Picture 5) Adding new student

When new student is added, by clicking the *Refresh* button the new student is available in the *Student's list*.



Picture 6) New student added

After the new student is added to the list it can be edited or deleted, same logic as editing the course. After clicking the *edit* button, the caption below tells if the edit was successful or not.

The screenshot shows a web application titled "Edit" in a dark header. The main content area is dark blue. A white modal box titled "Students Details" is centered. It contains four text input fields: "Name" with the value "NewStudent Changed", "Lastname" with the value "StudentLastName", "Note" with the value "note example", and "Grade" with the value "5". Below the fields are two buttons: a red "Delete" button and a green "Update" button. The "Update" button is highlighted with a red rectangle. Below the buttons, a red-bordered box contains the text "The Evaluation was updated successfully!".

Picture 7) Editing the student

Use Case:

A Teacher can create a Course for example : Computer Engineering. This Course can contain several Classes for Example Software Engineering , Control Systems , Mobile Development , Applied Algorithms for Engineering. Every one of these Classes has its own Evaluations for Example Software Engineering → Project, Exam, Presence, Cooperation. Students can be created and there Progress can be tracked. The Overall Use Case is that with this Applications a teacher can manage all his Courses, Classes, Evaluation and Students grades.

Course:

Title

Description

Year

Classes of Computer Engineering:

Search by title

« Prev 1 Next »

Software Engineering

Control Systems

Mobile Development

Items per Page: 3

Evaluations of Software Engineering

Evaluation - LIST

Project

Exam

Presence

Project

Criteria: Project

% : 40

Student of the Class Software Engineering

Student

Name : Albert

Lastname : Example

Note : Student of Software Engineering:

Project 2 , Exam 1 , Cooperation 3,
Presence 1

Grade : 2

Run the Project :

The Project can be found on Github:

<https://github.com/romanokeser/Web-App-User-Manager>

To start the Server (API Calls) run the following command from the „src“ directory.

```
node server.js
```

Once the Server is started it can be accessed in the Browser :

<http://localhost:8080/api/>

All data which is stored on the MongoDB from each component (Courses, Classes, Students...) can be seen here :

For Example : <http://localhost:8080/api/courses> will print out following :

```
{ "totalItems": 2, "courses": [ { "title": "Course example", "description": "Description example", "year": 2021, "published": false, "createdAt": "2021-06-17T13:45:08.871Z", "updatedAt": "2021-06-17T13:45:08.871Z", "id": "60cb51e4e6e34c0f949222df" }, { "title": "Course example 2 edited", "description": "Description example 2", "year": 2021, "published": false, "createdAt": "2021-06-17T13:46:08.069Z", "updatedAt": "2021-06-17T14:07:55.160Z", "id": "60cb5220e6e34c0f949222e0" } ], "totalPages": 1, "currentPage": 0 }
```

A simple way to test the basic CRUD API Functionality is with Postman Agent:

Methods	Urls	Actions
POST	/api/courses	create new Course
GET	/api/courses	retrieve all Courses
GET	/api/courses/:id	retrieve a Course by :id
PUT	/api/courses/:id	update a Course by :id
DELETE	/api/courses/:id	delete a Course by :id

To start the Angular Application from the „Web-App-User-Manager“ directory:

```
ng serve --port 8081
```

Once the Server is started it can be accessed in the Browser :

<http://localhost:8081/>

Tools Used :

Node.js

Express → to export REST APIs & interacts with MongoDB Database using Mongoose ODM.

Angular Client sends HTTP Requests and retrieves HTTP Responses using HTTPClient, consume data on the components.

Angular Router is used for navigating to pages.

Class- /Course- /Evaluations- / Student List is done with ngx-pagination

For Styling bootstrap was used.