جامعة
الملك سعود
King Saud University

ELM++

Graduation Project, Part-II (SWE 497)
Software Engineering Department
CCIS, KSU

Project Advisor:
Dr. Zohair Chentouf

Submitted by
Faisal Alhumayyd
Mohammed
Abdulrahman
Yousef

4/27/2021

ABSTRACT

This report describes the devolvement of our project which is an educational website aimed for those who are looking to improve their skills in a certain subject by taking courses in that specific field, and for any person that want to conduct a course for free or for a certain amount of fees.

Table of Contents

1. Introduction

Online education in the current situation has become a place of interest due to COVID 19 pandemic, The COVID-19 has resulted in schools shut across the world. Globally, a lot of people are out of the classroom.

As a result, education has changed dramatically, with the distinctive rise of e-learning, whereby teaching is undertaken remotely and on digital platforms.

In this project we will implement a platform that combine both graduate and undergraduate students mostly for people who have a full schedule, and they cannot take courses in their work time, most of the similar websites are foreign, and the local ones only focuses on students not including graduates.

In our project time is not an issue people will be able to increase their knowledge and information any time they want in any topic of their choosing such as basic math, IT & Software, Design, business, language courses etc.

Even more for teachers who wants to lecture a course online paid or free hopefully our project will help them do that also they can test their students understanding by conducting exams.

Students can look up courses easily and see the top-rated courses on the home page also rate their subscribed courses to help other people. Students can access the course content without the need of internet they can basically download the required content and start learning.

2. Domain Analysis

2.1. WEA

WEA is a charity website focuses on adult education for social purpose to increase social knowledge In the United Kingdom, the platform has online courses, and support a wide Range of Course Offerings the goal of the website is aspirations and develop educational opportunities for the most disadvantaged.[3] This includes providing basic math, English, and IT skills for employment. The disadvantage of this course that it is region locked that means it is exclusive for the people on the United Kingdom. [5]

2.2. Udemy

Udemy, Inc. is an American massive open online course provider aimed at professional adults and students.[2] the platform has more than 35 million students and 57,000 instructors teaching courses in over 65 languages.[5] It supports wide range of courses, and upon enrollment, students have instant access to the courses they have chosen, since all course materials, plus 30-Day Money-Back Guarantee If students determine that they do not wish to pursue a course for any reason during the first 30 days following enrollment, Udemy returns 100% of the tuition fees.[2] No Prequalification necessary to take Any course. Students can take any course that might interest them.

2.3. Coursera

Coursera is an American massive open online course provider, offers massive open online courses, specializations, degrees, professional and master track courses [5].
Coursera works with universities and other organizations to offer online courses, certifications, and degrees in a variety of subjects, such as engineering, data science, machine learning, mathematics, business, financing, computer science, digital marketing, humanities, medicine, biology, social sciences, 3000 plus variety of courses giving students a very broad range of information & experience in different fields. [1]

2.4. Shroo7

Is an educational platform that works to create an interactive learning environment by offering Special Services that help student to improving their knowledge and students who have difficulty studying with qualified and trusted teachers and provide online courses.[4]

2.5. Summary

Table 1 is a brief comparison of the main features of our website and the website listed above.

Features	**ElM++**	**WEA**	**Shroo7**	**Coursera**	**Udemy**
live lecturing			●		
course modification	●	●		●	●
course rating	●	●		●	●
automate exam grading	●			●	●

Provide course material.	●				
Print course certificate	●	●		●	

3. Quality Assurance Plan

- **Reviews**:
We plan to do weekly review with our advisor to make sure that we are going on the right track by taking our advisor's notes and apply changings to the project.

- **Testing**:
The system will go through different kinds of testing such as: reliability testing, integrity testing and deployment testing.

-Verification:
Can be assured the developers built the specified design when the actual design output is the same as the expected design output that meets the product specification. It can be assured if developers followed standards by creating specific guidelines, developers training, and using templates and checklists.

- Walkthroughs:
Every team member will go through the artifacts of the project individually to gather enhancement point for the final system.

- Training Team members:
There will be training to each of the group member to increase the overall skills in web development.

4. Requirements

4.1 Glossary

In this section we list and define the four of users of the proposed system

1- **Administrator:** a user takes the control of lecturer and student accounts and approve or deny new courses.
2- **Student:** a user that sign up in the system as student looking for learning from our platform by subscribe to courses he wants.
3- **lecturer:** a user adds courses and courses materials for our platform looking to teach other students.
4- **Visitor:** a user that did not sign up in the system

4.2 Functional Requirements

Administrator-related requirements:

1. Admin shall be able to edit his account password.
2. Admin shall be able to view his account information (Username, email).
3. Admin shall be able to delete lecturer account.
4. Admin shall be able to approve/deny lecturer account.
5. Admin shall be able view Student's username and email.
6. Admin shall be able view lecturer username, email, national Id, name, phone.
7. Admin shall be able to login using his username and password.
8. Admin shall be able to view the following statistics:
   - Number of users.
   - Number of registered courses in each month.
9. Admin shall be able to view courses.
10. Admin shall be able to delete courses.
11. Admin shall be able to review course materials.

Student-related requirements:

1. Student shall be able to edit his account information except username.
2. Student shall be able to view his account information (Username, email).
3. Student shall be able to login using his username and password.
4. Student shall be able to Sign-up using his (Email, Username, password,).
5. Student shall be able to search for a course.
6. Student shall be able to subscribe in a course.
7. Student shall be able to unsubscribe from a course.
8. Student shall be able to report a course.
9. Student shall be able to rate a course.
10. Student shall be able to view course rating.
11. Student shall be able to download the course.
12. Student shall be able to take course exam.
13. Student shall be able to view exam results.
14. Student shall be able to upgrade his account to premium.

lecturer -related requirements:

1. lecturer shall be able to edit his account information except username.
2. lecturer shall be able to view his account information (Username, email, phone).
3. lecturer shall be able to delete his account.
4. lecturer shall be able to login using his username and password.
5. lecturer shall be able to Sign-up using his (Email, Username, password,).
6. lecturer shall be able to delete course.
7. lecturer shall be able to add course.
8. lecturer shall be able to edit the name of the course.
9. lecturer shall be able to delete course materials.
10. lecturer shall be able to add course materials.
11. lecturer shall be able to edit the name of the course material.
12. lecturer shall be able to upload course materials.

13. lecturer shall be able to view the following statistics.
    -Number of participants
    -Course rating
    -Top rated courses

14. lecturer shall be able to conduct an exam.

Visitor-related requirements:

1. Visitor shall be able to Sign-up as lecturer or student.
2. Visitor shall be able to view search course.
3. Visitor shall be able to view top rated courses in given category.

4.3 Non-Functional Requirements

Supportability:

- The system shall support Arabic and English languages.

Usability:

- The user shall be able to learn the interface in no longer than 30 minutes.

Security:

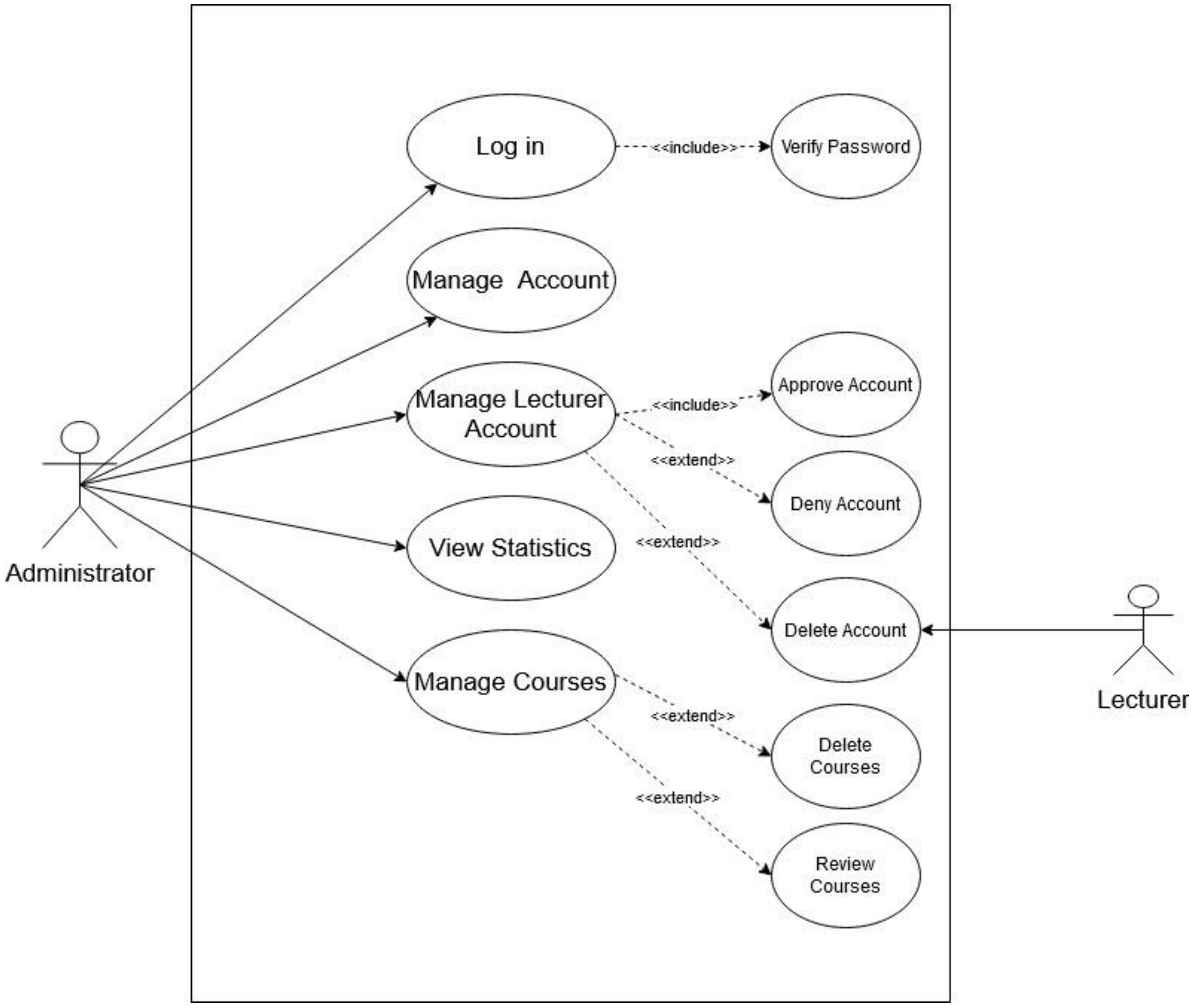
- The system shall store user's personal and payment info in an encrypted form.

Performance:

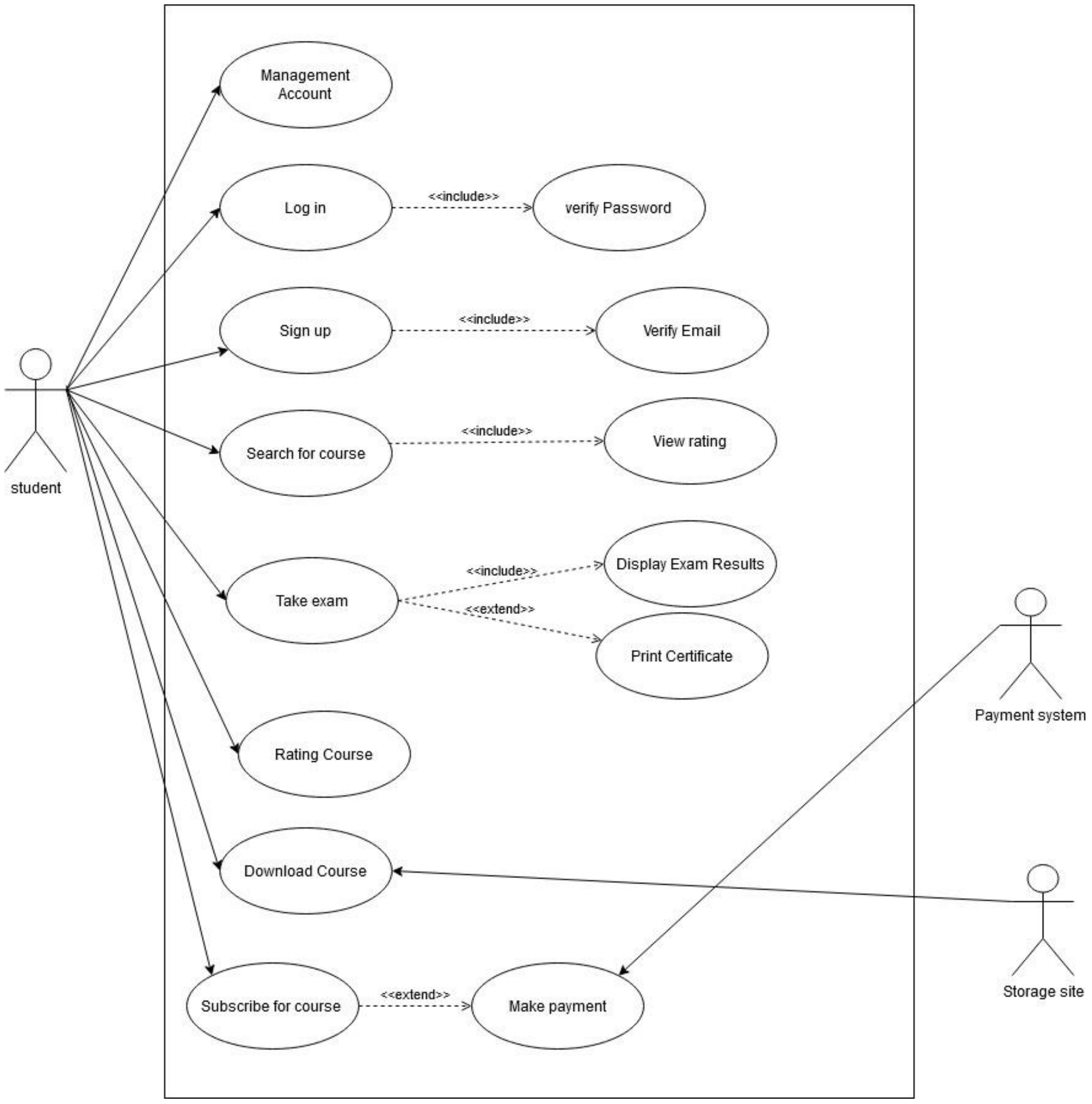
- The system shall be able to be serving 100 at same time.

5. System Use-Cases

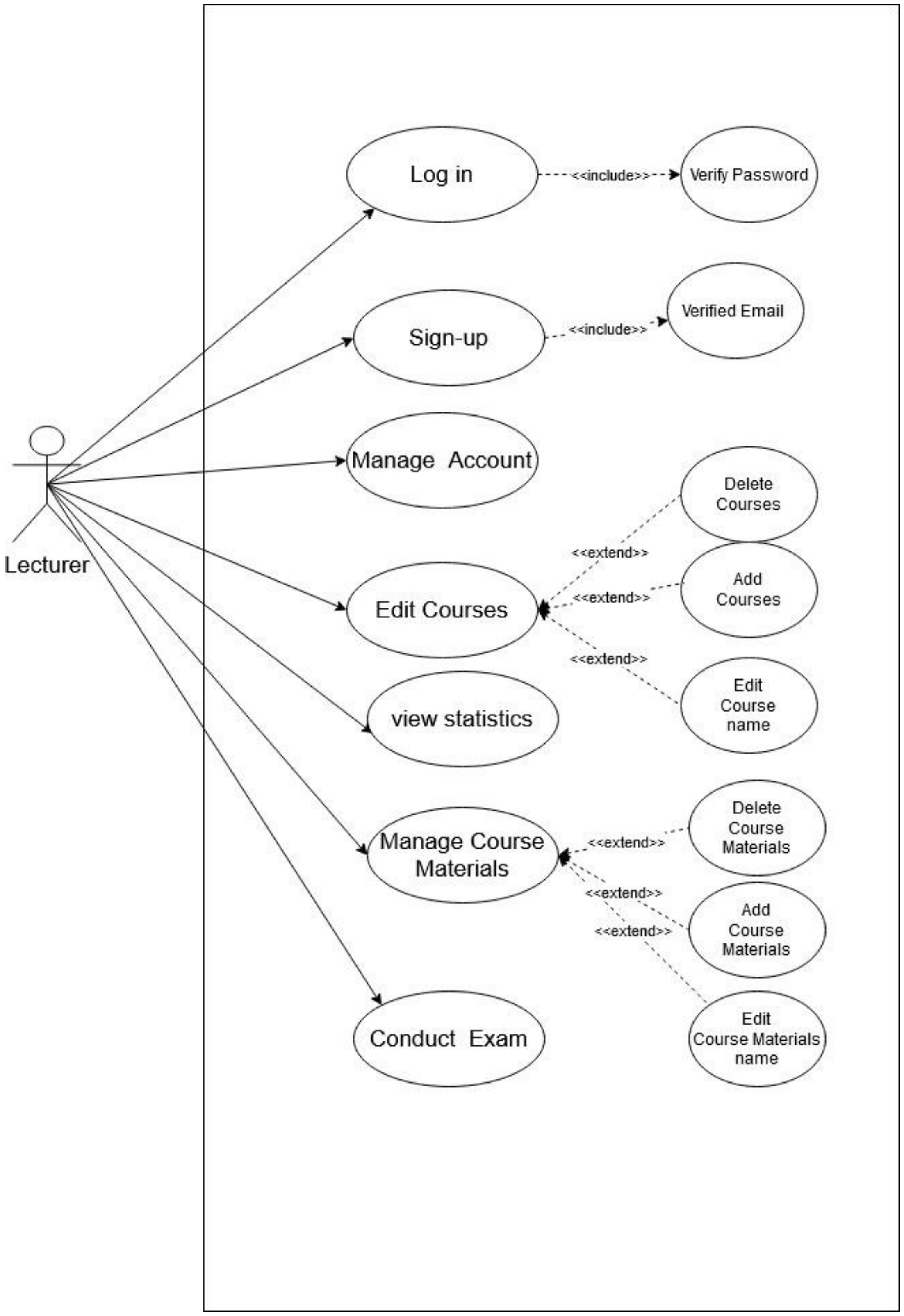
5.1 Administrator Use-Cases



6.2 Student Use-Cases



6.3 Lecturer Use-Cases



6.4 Use case description

Use Case Description	
Use Case Name: Log in	
Actor’s: lecturer, student, Administrator	
Description: let the actor get access into his account	
Relationship’s: Include: Verify Password properties Extends: None	Covered requirements: (R1.2), (R2.3), (R3.4)
Pre-condition: the actor is not log in	
Basic Flow	
Actor: 1. Enter his username and Password	System: 2. Validate the username and password
Alternative Flow: 4.1 the username and Password do not match the record 4.2 continue with step 1	

Use Case Description	
Use Case Name: Sign Up	
Actor’s: lecturer, student	
Description: The actor creates an account in the system	
Relationship’s: Include: none properties Extends: None	Covered requirements: (R2.3) , (R3.5)
Pre-condition: the actor is not log in	
Basic Flow	
Actor: 1. Enter Username, Email, Password, Re-Password.	System: 2. Check if the username available and all data correct.
Alternative Flow: 2.1. If username not available or data does not correct. 2.2. The system clears the rejected field. 2.3. The user enters the rejected field again.	

Use Case Description	
Use Case Name: Rating course	
Actor’s: student	
Description: the student rate curses from 1 to 5	
Relationship’s: Include: none properties Extends: None	Covered requirements: (R2.9)
Pre-condition: none	
Basic Flow	
Actor: 1. select the rate	System: 2. Save course rate.
Alternative Flow: none	

Use Case Description	
Use Case Name: Manage account	
Actor’s: Administrator, Student, Lecturer	
Description: Let the actor view email, username and edit his email or password	
Relationship’s: Include: none properties Extends: None	Covered requirements: (R1.1) (R2.1) (R3.1) (R1.2) (R2.2) (R3.2)
Pre-condition: The user should be already signed in	
Basic Flow	
Actor: 1. insert new email or Password.	System: 2. save the new update
Alternative Flow: none	

Use Case Description	
Use Case Name: Manage Lecturer Account	
Actor’s: Administrator Secondary actor: lecturer	
Description: Let the Administrator approve and deny and delete lecturer account and view accounts information.	
Relationship’s: Include: none properties Extends: 1.Approve account 2. Deny account 3. Delete account	Covered requirements: (R1.4) (R1.3) (R1.6)
Pre-condition: The actor should be already signed in	
Basic Flow	
Actor: 2.choose the lecturer. 4.approve, deny, delete.	System: 1. List lectures accounts. 3.Open the lecturer profile. 5.save the new update.
Alternative Flow: none	

Use Case Description	
Use Case Name: Manage courses	
Actor's: Administrator	
Description: Let the Administrator delete and review and approve courses	
Relationship's: Include: none properties Extends: 1.Delete courses 2. Review courses	Covered requirements: (R1.9) (R1.10) (R1.11)
Pre-condition: none	
Basic Flow	
Actor: 2.open the course	System: 1. List the courses
Alternative Flow: none	

Use Case Description	
Use Case Name: Edit courses	
Actor’s: Lecturer	
Description: Let the Lecturer delete and add and edit courses name	
Relationship’s: Include: none properties Extends: 1.Delete courses 2. add courses 3. edit courses	Covered requirements: (R3.6) (R3.7) (R3.8)
Pre-condition: none	
Basic Flow	
Actor: 2.choose the courses 3. edit name or delete courses	System: 1. List the courses 4.save new update
Alternative Flow: none	

Use Case Description	
Use Case Name: View statistic	
Actor’s: Administrator. Lecturer	
Description: Let the actor view the statistic of number of users and Number of registered courses in each month	
Relationship’s: Include: none properties Extends: none	Covered requirements: (R1.4) (R1.3) (R1.6)
Pre-condition:	
Basic Flow	
Actor: 1. open view statistic	System: 2. Display view statistic.
Alternative Flow: none	

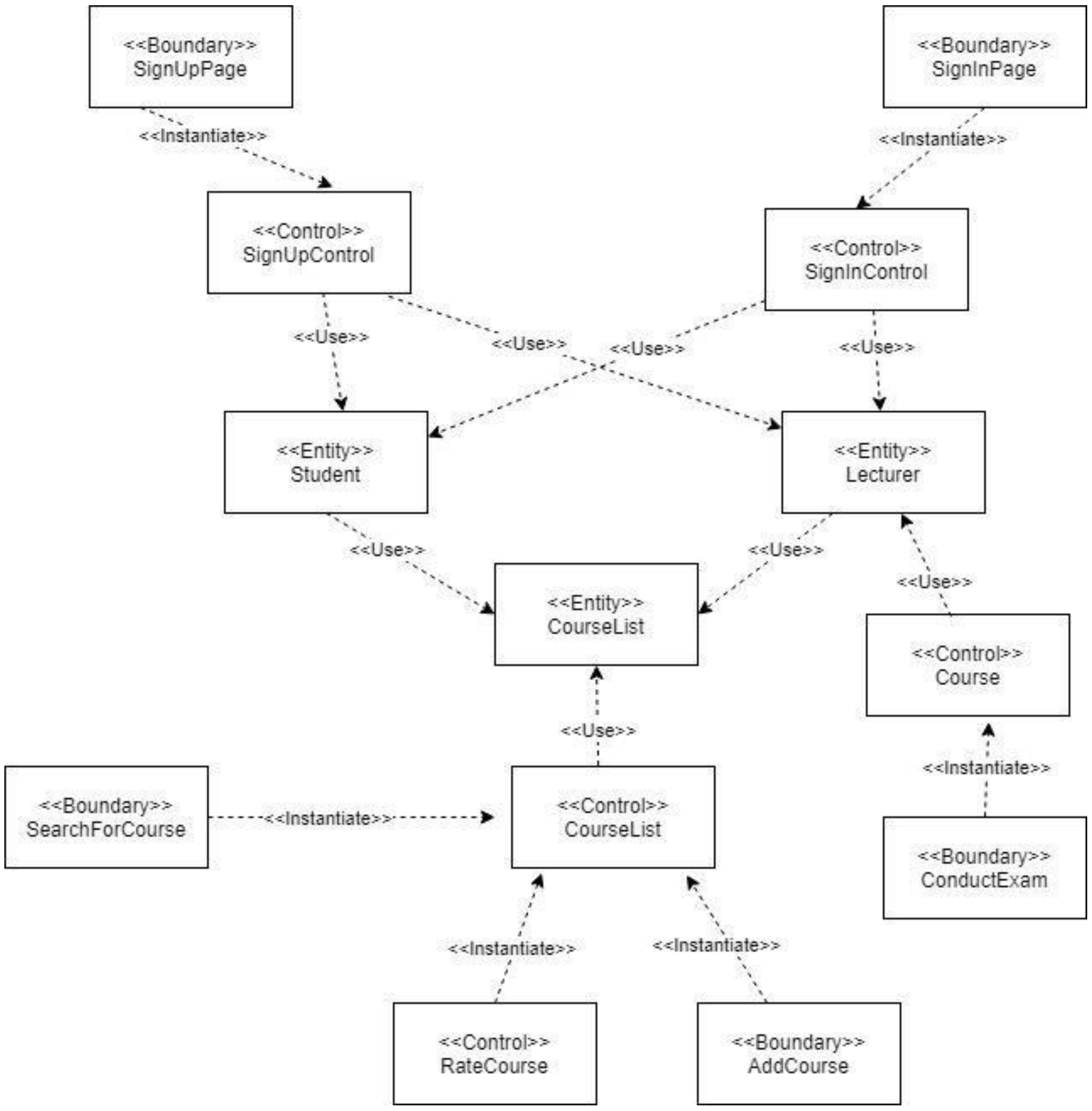
Use Case Description	
Use Case Name: Search for a course	
Actor: Student	
Description: let the student search for a course and view the course’s rating.	
Relationship’s: Include: non properties Extends: View rating	Covered requirements: (R2.5),(R2.10)
Pre-condition: none	
Basic Flow	
Actor: 1.enter key word	System: 2.Display search result.
Alternative Flow: 2.1. If there is no result display ‘not found’.	

Use Case Description	
Use Case Name: Take exam	
Actor: Student	
Description: let the student take the exam after finishing the course the view the exam result.	
Relationship's: Include: Display Exam result. properties Extends: none.	Covered requirements: (R2.12),(R2.13)
Pre-condition: finished the course	
Basic Flow	
Actor: 1.Open exam. 3.Submit	System: 2.Display exam. 4.Display the grade
Alternative Flow: none	

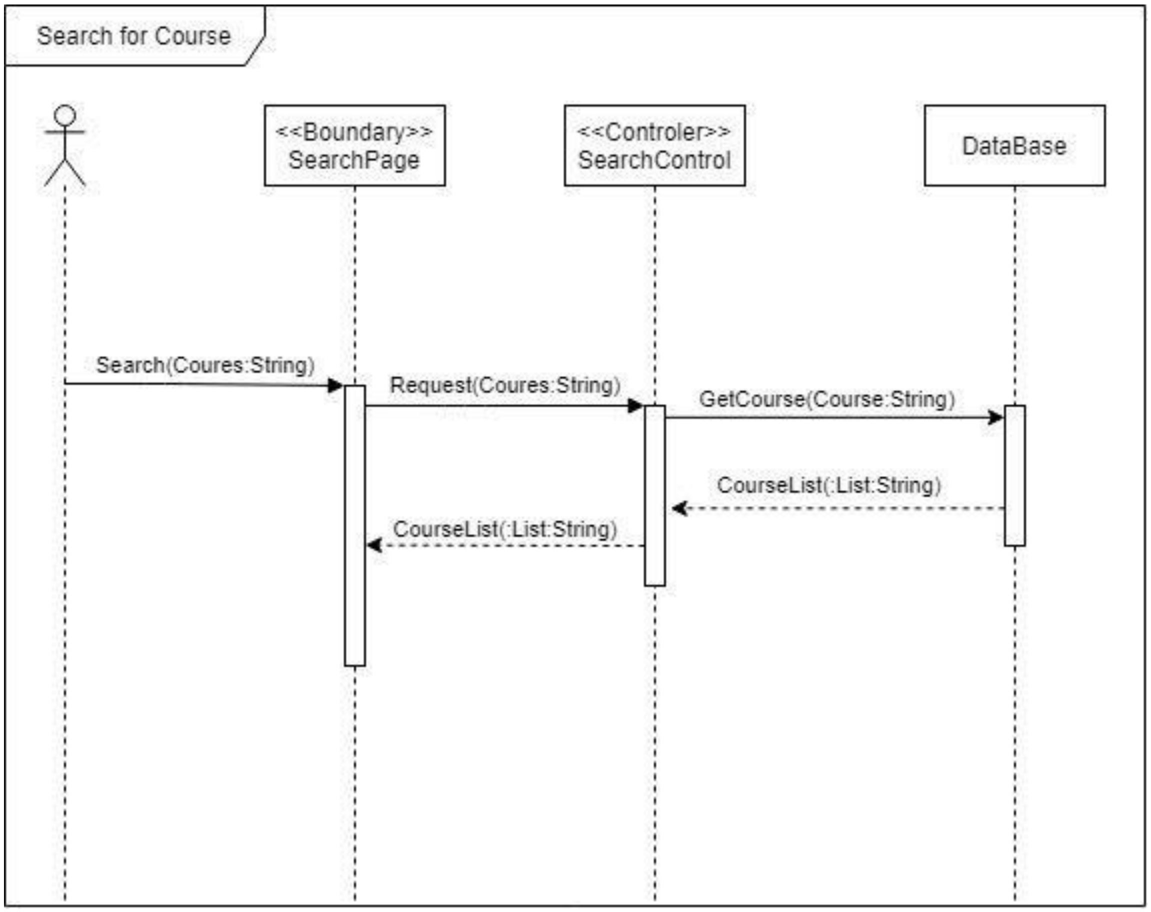
Use Case Description	
Use Case Name: Manage Course Material	
Actor: Lecturer	
Description: Let the lecturer add and edit and delete the course material	
Relationship's: Include: none. properties Extends: 1. Delete Course Material. 2. Add Course Material. 3. Edit Course Material.	Covered requirements: (R2.12), (R2.13)
Pre-condition: The lecturer already in the course page	
Basic Flow	
Actor: 1.Edit, add, delete course material	System: 2.save new update
Alternative Flow: none	

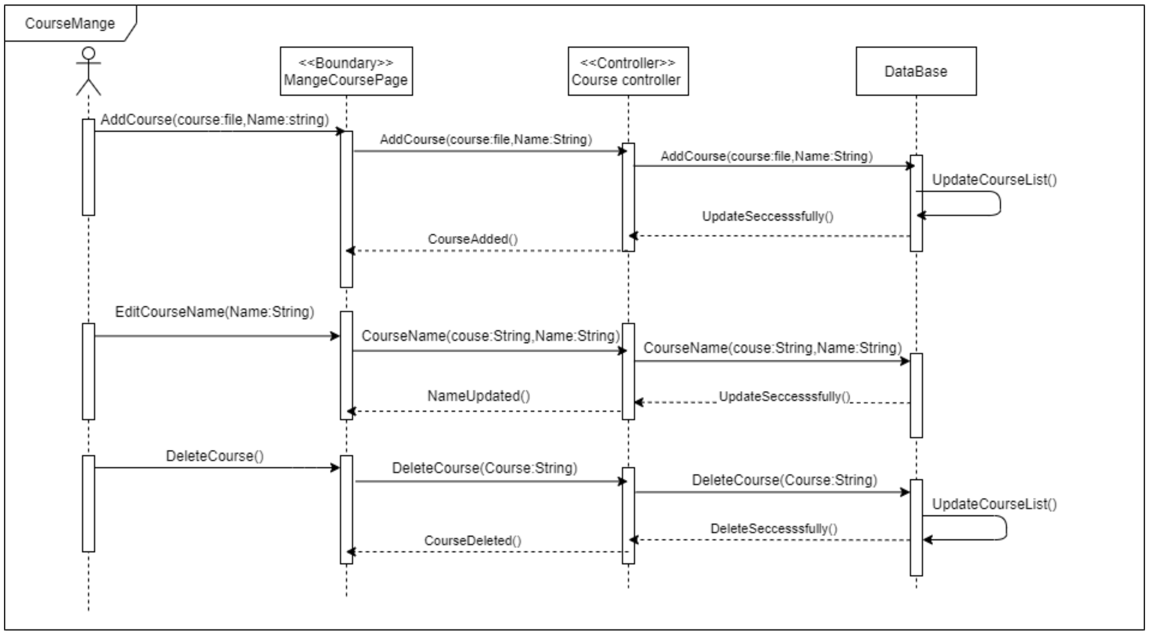
Use Case Description	
Use Case Name: Download Course	
Actor: Student	
Description: let the student to download the subscribed courses.	
Relationship's: Include: none. properties Extends: none.	Covered requirements: (R2.11)
Pre-condition: the actor should be subscripted in the course	
Basic Flow	
Actor: 1.Download course.	System: 2.Open download page
Alternative Flow: none	

6. Analysis Class

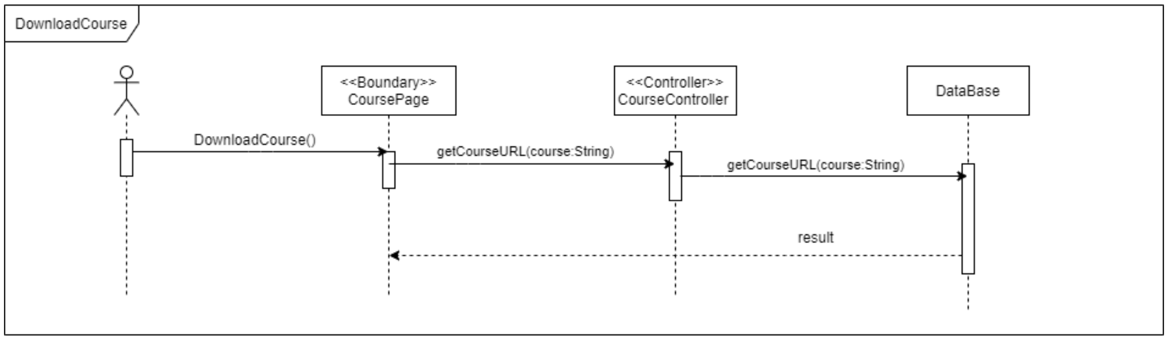


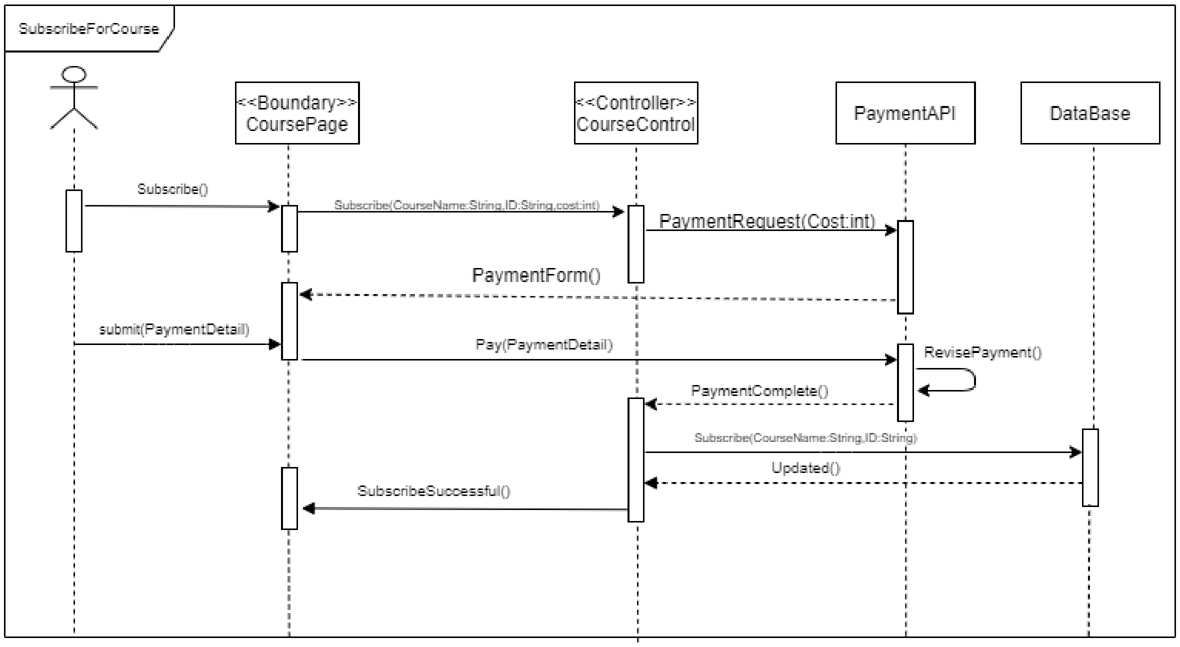
7. Interaction Diagrams

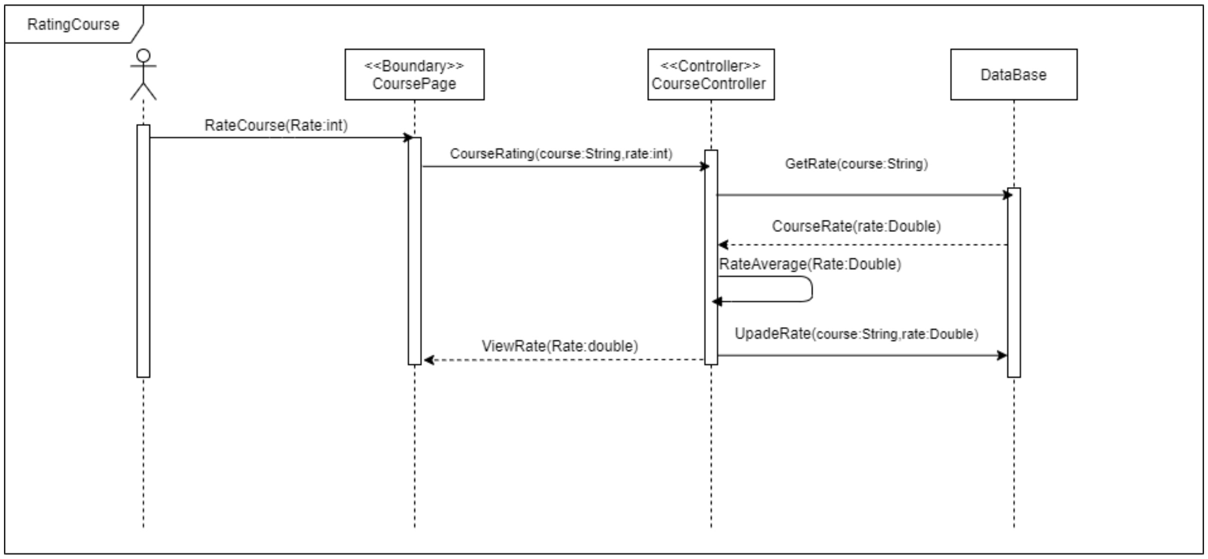




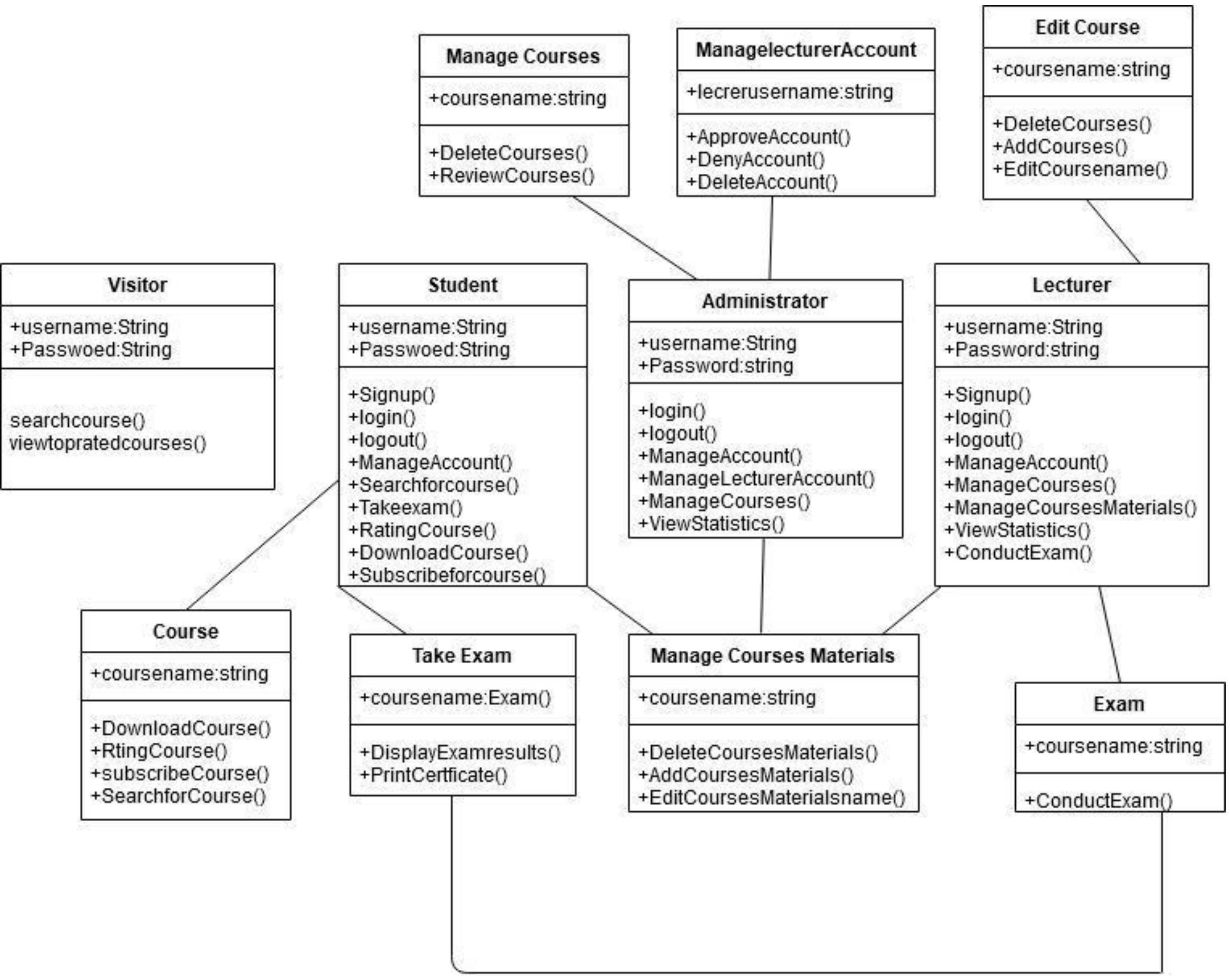
CourseMange
<<Boundary>> MangeCoursePage
<<Controller>> Course controller
DataBase
AddCourse(course:file,Name:string)
AddCourse(course:file,Name:String)
AddCourse(course:file,Name:String)
UpdateCourseList()
UpdateSeccesssfully()
CourseAdded()
EditCourseName(Name:String)
CourseName(couse:String,Name:String)
CourseName(couse:String,Name:String)
NameUpdated()
UpdateSeccesssfully()
DeleteCourse()
DeleteCourse(Course:String)
DeleteCourse(Course:String)
UpdateCourseList()
DeleteSeccesssfully()
CourseDeleted()


DownloadCourse
<<Boundary>> CoursePage
<<Controller>> CourseController
DataBase
DownloadCourse()
getCourseURL(course:String)
getCourseURL(course:String)
result


SubscribeForCourse
<<Boundary>> CoursePage
<<Controller>> CourseControl
PaymentAPI
DataBase
Subscribe()
Subscribe(CourseName:String,ID:String,cost:int)
PaymentRequest(Cost:int)
PaymentForm()
submit(PaymentDetail)
Pay(PaymentDetail)
RevisePayment()
PaymentComplete()
Subscribe(CourseName:String,ID:String)
Updated()
SubscribeSuccessful()


RatingCourse
<<Boundary>> CoursePage
<<Controller>> CourseController
DataBase
RateCourse(Rate:int)
CourseRating(course:String,rate:int)
GetRate(course:String)
CourseRate(rate:Double)
RateAverage(Rate:Double)
UpadeRate(course:String,rate:Double)
ViewRate(Rate:double)

8. Class diagram



9. System Architecture

We went for the MVC architecture, MVC patterns separate the input, processing, and output of an application. This model divided into three interconnected parts called the model, the view, and the controller.

As of why we choose it, simply because it is popular and well-suited architecture for web application, also it supports rapid and parallel development. Therefor one team member can work on the frontend while the other can work on the backend or the controller to create the business logic of the web application.

Another important features that the MVC architecture can also integrate with the JavaScript Framework. This means that MVC applications can be made to work even with PDF files, site-specific browsers, also with desktop widgets. And MVC support of Asynchronous Technique which helps for faster load time.

The most important feature is separation of concerns, it is obvious that you make frequent changes in your web application like changing colors, fonts, screen layouts, and adding new device support for mobile phones or tablets. Moreover, adding a new type of view are very easy in the MVC pattern because the Model part does not depend on the views part. Therefore, any changes in the Model will not affect the entire architecture.

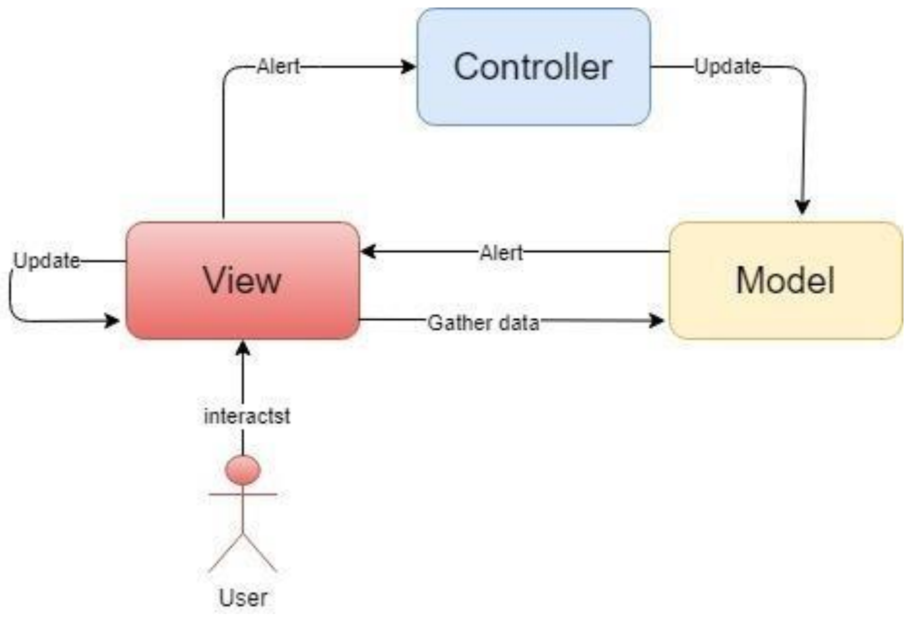


Figure 3. System Architecture

10. Prototype Description

10.1 Implementation Platform

 Visual studio code, brackets

 HTML, CSS, JavaScript,

_ PHP

 MySql RDBMS

 Xampp web server

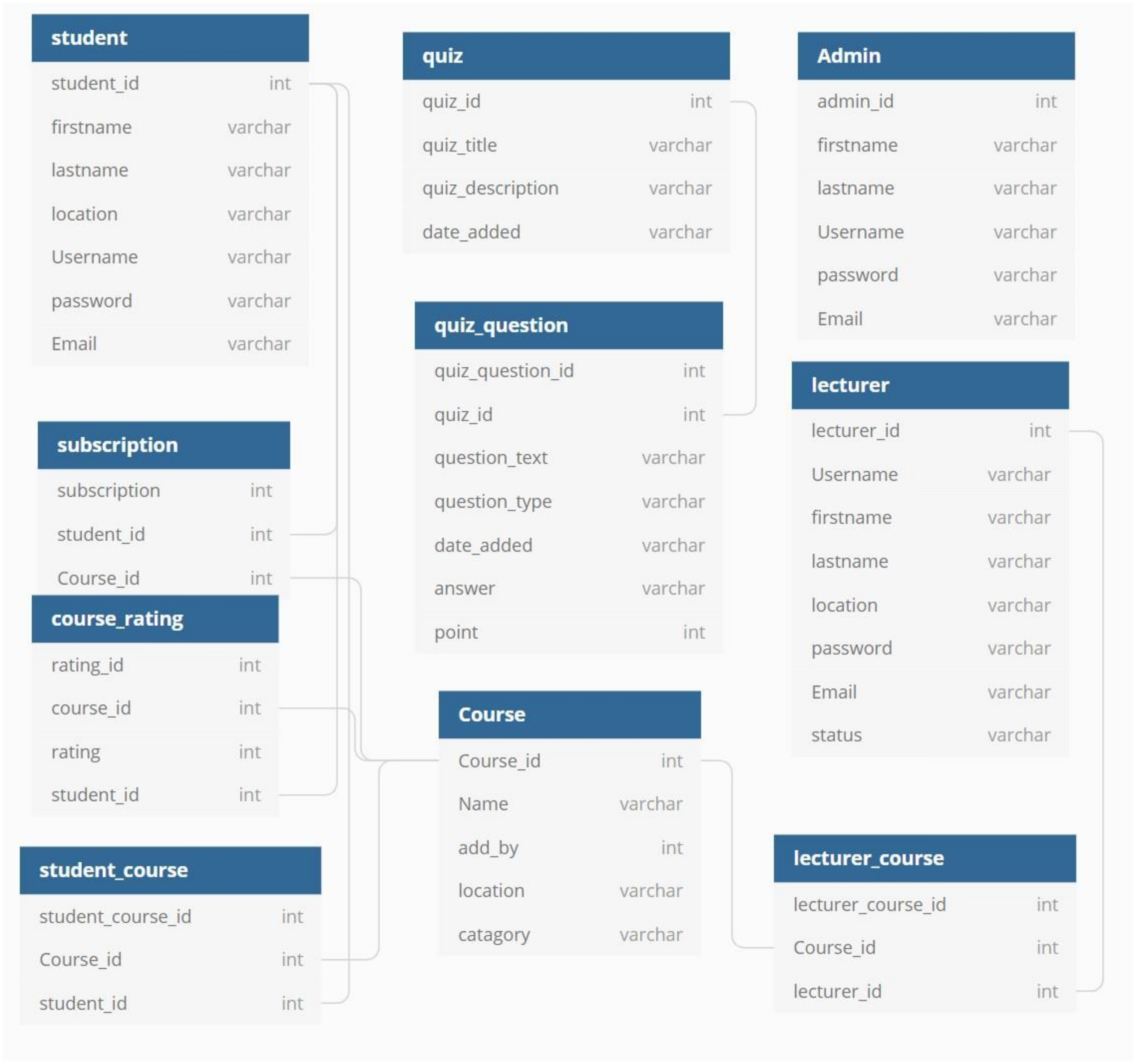
10.2 Mapping between Requirements and Implemented Functions

The following table shows a mapping between the projected functionalities of the system and the corresponding implemented modules.

Table 1. Test Table

Admin	**Functional requirements**	**Modules/Functions/Class that implemented this feature**
	Admin Login	login.php
	Change Password	update_password_admin.php
	Change Profile information	change_admin_.php
	Activate Lecturer	Activate.php
	List Students	student_table.php
	List Courses	course.php
	Delete Course	Delete_course.php
	Stats	Stats.php
Lecturer		
	Login	Login_.php
	Sign up	Lecturer_signup.php
	show Profile	lecturer-profile.php
	Change Profile information	change_lecturer_ php
	Add Course	Add_course.php
	Delete Course	delete_course.php
	Edit Course	edit_course_form.php
	Add Material	upload_save.php
	Delete Material	delete_file.php
Student		
	Login	Login_.php
	Sign up	student_signup.php
	show Profile	Student_profile.php
	Change Profile information	change_student_.php
	Search Course	Student_dashboard.php
	Subscribe Course	subscribe.php
	Unsubscribe Course	Unsubscribe.php
	Rate Course	Add_rate.php

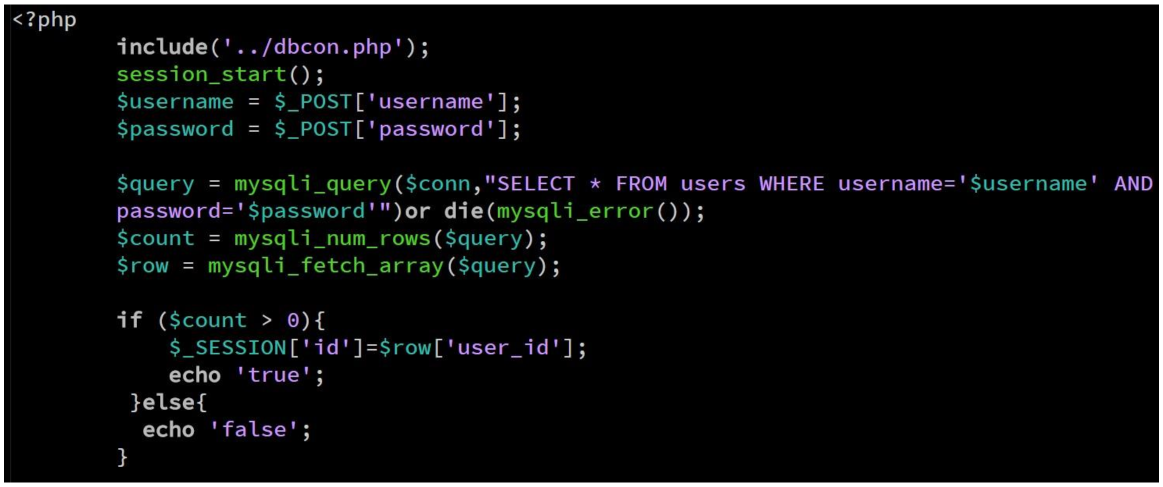
10.3 Database schema



10.4 Implementation Details

- Login

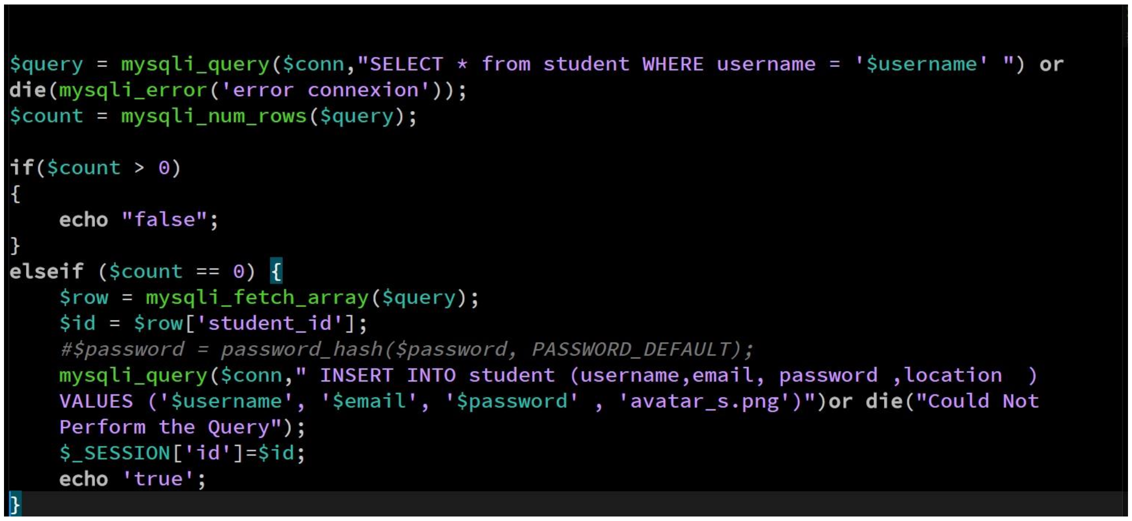
  After Actor (admin, lecturer, or Student) Fill Form, Post data to login.php file and check if enter data is existing on DB or not. If yes, we start session for actor by id.



```
<?php
        include('../dbcon.php');
        session_start();
        $username = $_POST['username'];
        $password = $_POST['password'];

        $query = mysqli_query($conn,"SELECT * FROM users WHERE username='$username' AND
        password='$password'")or die(mysqli_error());
        $count = mysqli_num_rows($query);
        $row = mysqli_fetch_array($query);

        if ($count > 0){
            $_SESSION['id']=$row['user_id'];
            echo 'true';
         }else{
          echo 'false';
        }
```

- Signup

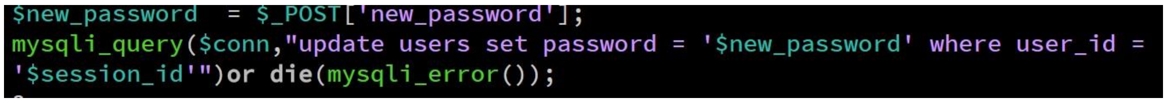
  After Actor (lecturer, or Student) Fill Form, Post data to php file and check if enter username is existing on DB or not. If yes, return error, if not create account.



```
$query = mysqli_query($conn,"SELECT * from student WHERE username = '$username' ") or
die(mysqli_error('error connexion'));
$count = mysqli_num_rows($query);

if($count > 0)
{
    echo "false";
}
elseif ($count == 0) {
    $row = mysqli_fetch_array($query);
    $id = $row['student_id'];
    #$password = password_hash($password, PASSWORD_DEFAULT);
    mysqli_query($conn," INSERT INTO student (username,email, password ,location  )
    VALUES ('$username', '$email', '$password' , 'avatar_s.png')")or die("Could Not
    Perform the Query");
    $_SESSION['id']=$id;
    echo 'true';
}
```

- Change Password

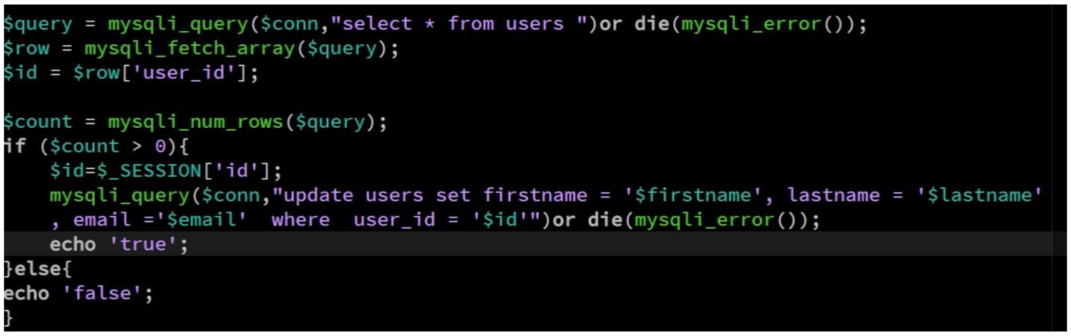
  Actors enter new password and update it on Database.



```
$new_password  = $_POST['new_password'];
mysqli_query($conn,"update users set password = '$new_password' where user_id =
'$session_id'")or die(mysqli_error());
```

- Change Profile Info

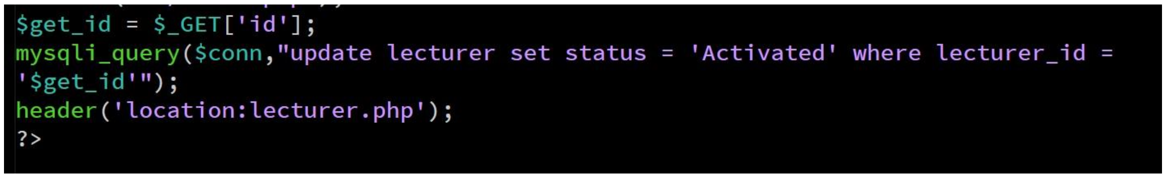
  Update actor information on DB based on id, we get id of actor from session.



```
$query = mysqli_query($conn,"select * from users ")or die(mysqli_error());
$row = mysqli_fetch_array($query);
$id = $row['user_id'];

$count = mysqli_num_rows($query);
if ($count > 0){
    $id=$_SESSION['id'];
    mysqli_query($conn,"update users set firstname = '$firstname', lastname = '$lastname'
    , email ='$email'  where  user_id = '$id'")or die(mysqli_error());
    echo 'true';
}else{
echo 'false';
}
```

- Activate Lecturer

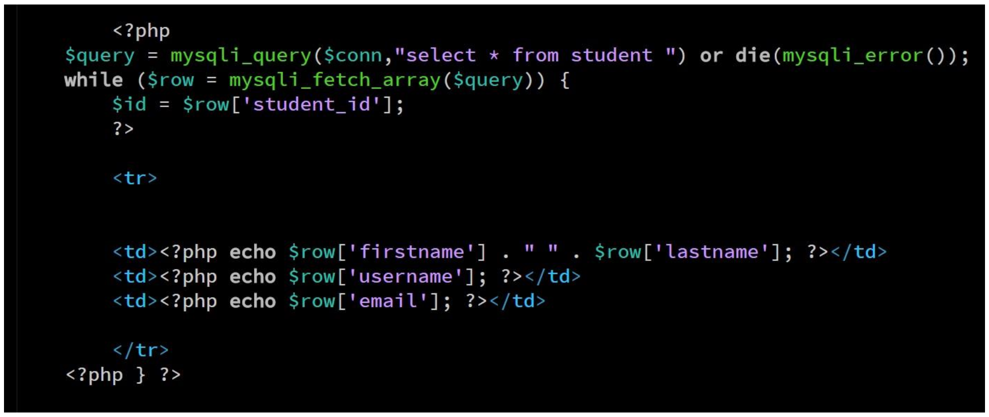
  Update status of lecturer on DB, and Deactive is same operation



```
$get_id = $_GET['id'];
mysqli_query($conn,"update lecturer set status = 'Activated' where lecturer_id =
'$get_id'");
header('location:lecturer.php');
?>
```

- List Student

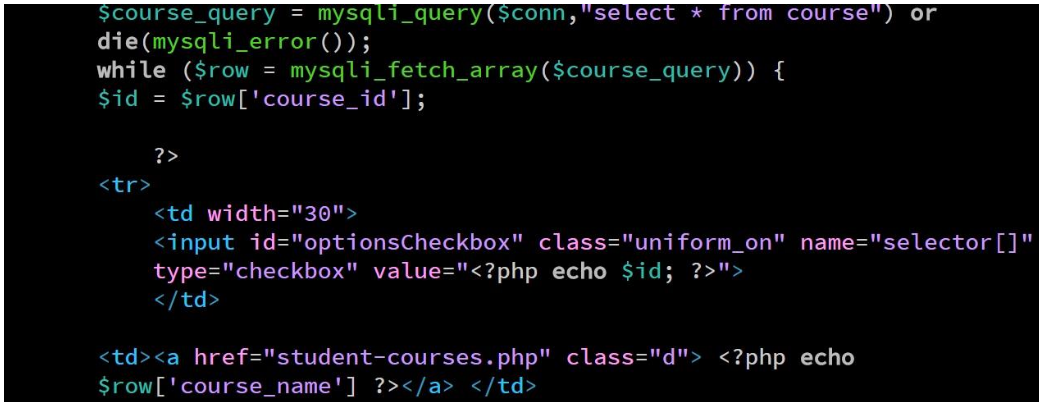
  Get all student from DB and List it on HTML Tags



```
        <?php
    $query = mysqli_query($conn,"select * from student ") or die(mysqli_error());
    while ($row = mysqli_fetch_array($query)) {
        $id = $row['student_id'];
        ?>

        <tr>


        <td><?php echo $row['firstname'] . " " . $row['lastname']; ?></td>
        <td><?php echo $row['username']; ?></td>
        <td><?php echo $row['email']; ?></td>

        </tr>
    <?php } ?>
```

- List Course

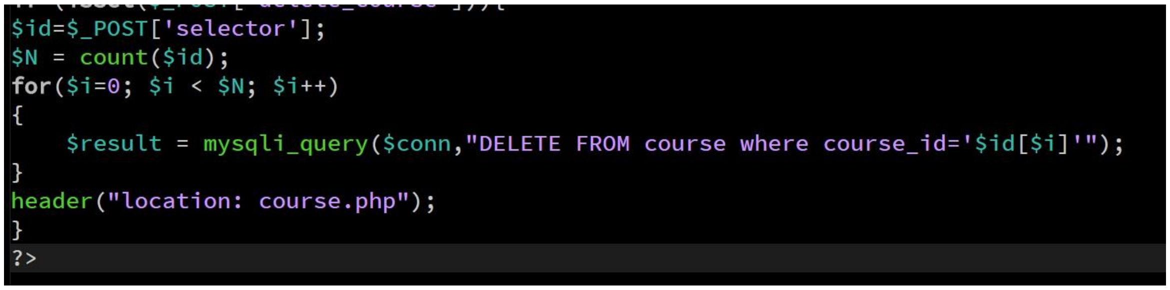
Get all courses from DB and List it on HTML Tags



```
        $course_query = mysqli_query($conn,"select * from course") or
        die(mysqli_error());
        while ($row = mysqli_fetch_array($course_query)) {
        $id = $row['course_id'];

            ?>
        <tr>
            <td width="30">
            <input id="optionsCheckbox" class="uniform_on" name="selector[]"
            type="checkbox" value="<?php echo $id; ?>">
            </td>

        <td><a href="student-courses.php" class="d"> <?php echo
        $row['course_name'] ?></a> </td>
```

- Delete Course

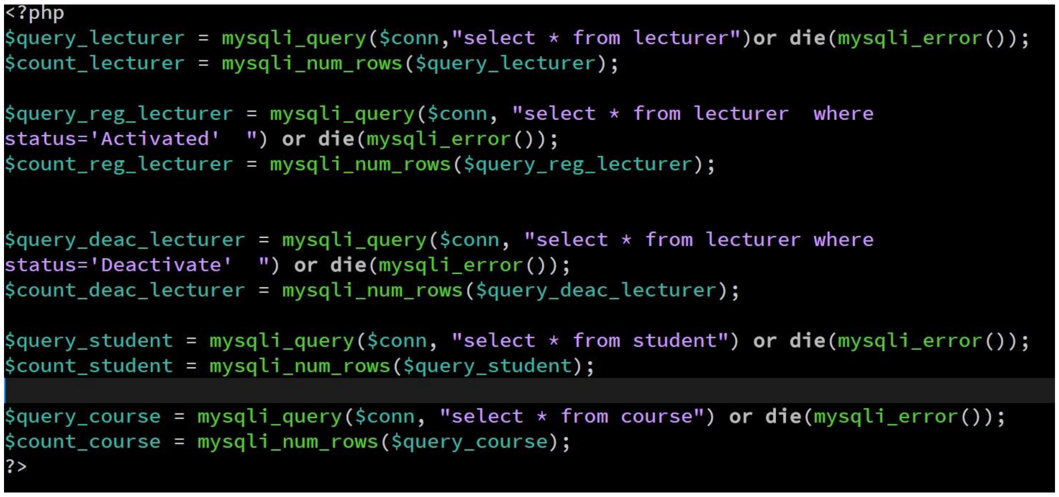
Delete selected courses from db, we get selected courses from checkbox form



```
$id=$_POST['selector'];
$N = count($id);
for($i=0; $i < $N; $i++)
{
    $result = mysqli_query($conn,"DELETE FROM course where course_id='$id[$i]'");
}
header("location: course.php");
}
?>
```

- Stats

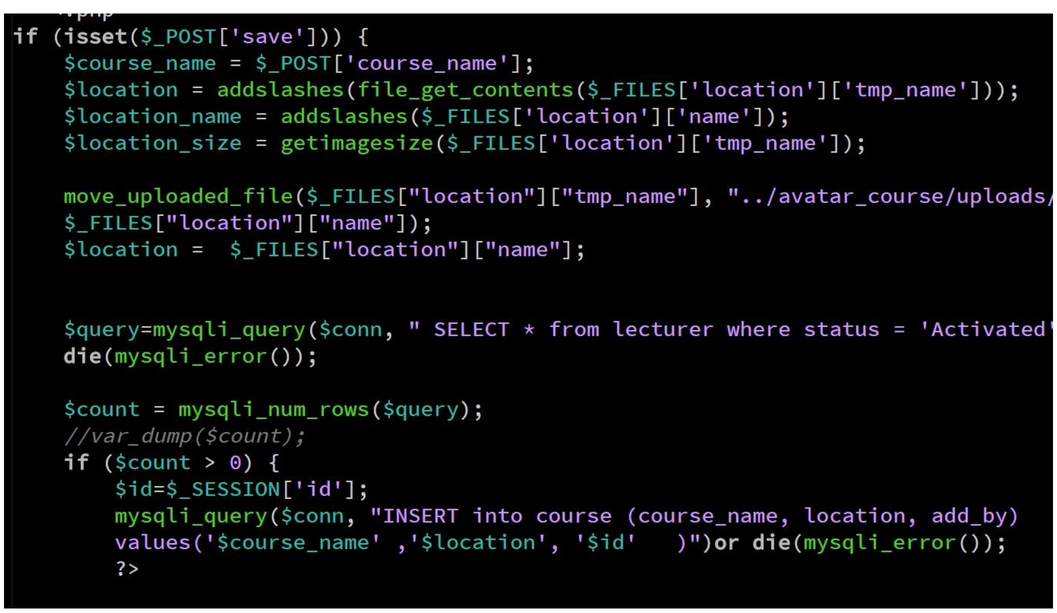
Count numbers of students, courses, active lecturers, and Deactives Lecturers From DB.



```
<?php
$query_lecturer = mysqli_query($conn,"select * from lecturer")or die(mysqli_error());
$count_lecturer = mysqli_num_rows($query_lecturer);

$query_reg_lecturer = mysqli_query($conn, "select * from lecturer  where
status='Activated'  ") or die(mysqli_error());
$count_reg_lecturer = mysqli_num_rows($query_reg_lecturer);


$query_deac_lecturer = mysqli_query($conn, "select * from lecturer where
status='Deactivate'  ") or die(mysqli_error());
$count_deac_lecturer = mysqli_num_rows($query_deac_lecturer);

$query_student = mysqli_query($conn, "select * from student") or die(mysqli_error());
$count_student = mysqli_num_rows($query_student);

$query_course = mysqli_query($conn, "select * from course") or die(mysqli_error());
$count_course = mysqli_num_rows($query_course);
?>
```

- Add Course

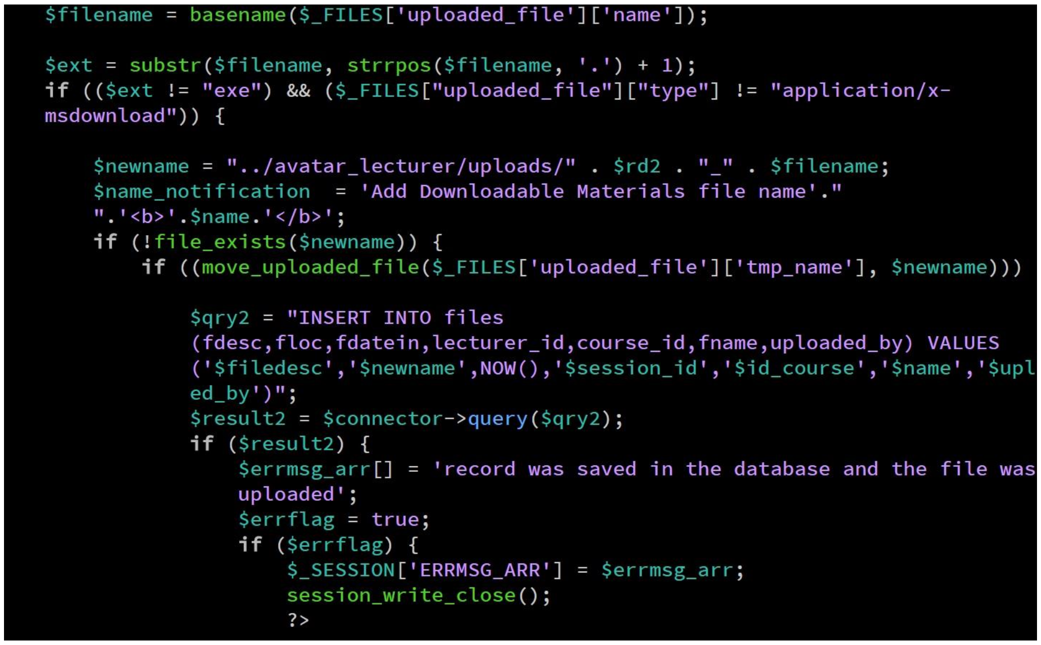
After fill form HTML, add info to DB, and Move Image (course Cover) to Site Folder to call it.



```
if (isset($_POST['save'])) {
    $course_name = $_POST['course_name'];
    $location = addslashes(file_get_contents($_FILES['location']['tmp_name']));
    $location_name = addslashes($_FILES['location']['name']);
    $location_size = getimagesize($_FILES['location']['tmp_name']);

    move_uploaded_file($_FILES["location"]["tmp_name"], "../avatar_course/uploads/
    $_FILES["location"]["name"]);
    $location =  $_FILES["location"]["name"];


    $query=mysqli_query($conn, " SELECT * from lecturer where status = 'Activated'
    die(mysqli_error());

    $count = mysqli_num_rows($query);
    //var_dump($count);
    if ($count > 0) {
        $id=$_SESSION['id'];
        mysqli_query($conn, "INSERT into course (course_name, location, add_by)
        values('$course_name' ,'$location', '$id'   )")or die(mysqli_error());
        ?>
```

- Add Material

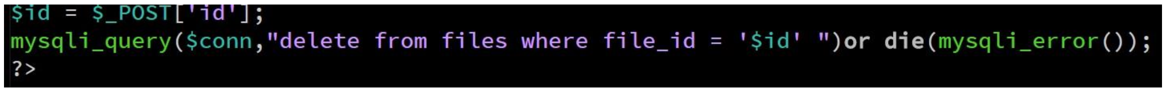
Insert Material name on DB and move file into Site Folder after checking it is not danger file.



```
   $filename = basename($_FILES['uploaded_file']['name']);

   $ext = substr($filename, strrpos($filename, '.') + 1);
   if (($ext != "exe") && ($_FILES["uploaded_file"]["type"] != "application/x-
   msdownload")) {

       $newname = "../avatar_lecturer/uploads/" . $rd2 . "_" . $filename;
       $name_notification  = 'Add Downloadable Materials file name'."
       ".'<b>'.$name.'</b>';
       if (!file_exists($newname)) {
           if ((move_uploaded_file($_FILES['uploaded_file']['tmp_name'], $newname)))

               $qry2 = "INSERT INTO files
               (fdesc,floc,fdatein,lecturer_id,course_id,fname,uploaded_by) VALUES
               ('$filedesc','$newname',NOW(),'$session_id','$id_course','$name','$upl
               ed_by')";
               $result2 = $connector->query($qry2);
               if ($result2) {
                   $errmsg_arr[] = 'record was saved in the database and the file was
                   uploaded';
                   $errflag = true;
                   if ($errflag) {
                       $_SESSION['ERRMSG_ARR'] = $errmsg_arr;
                       session_write_close();
                       ?>
```

- Delete Material

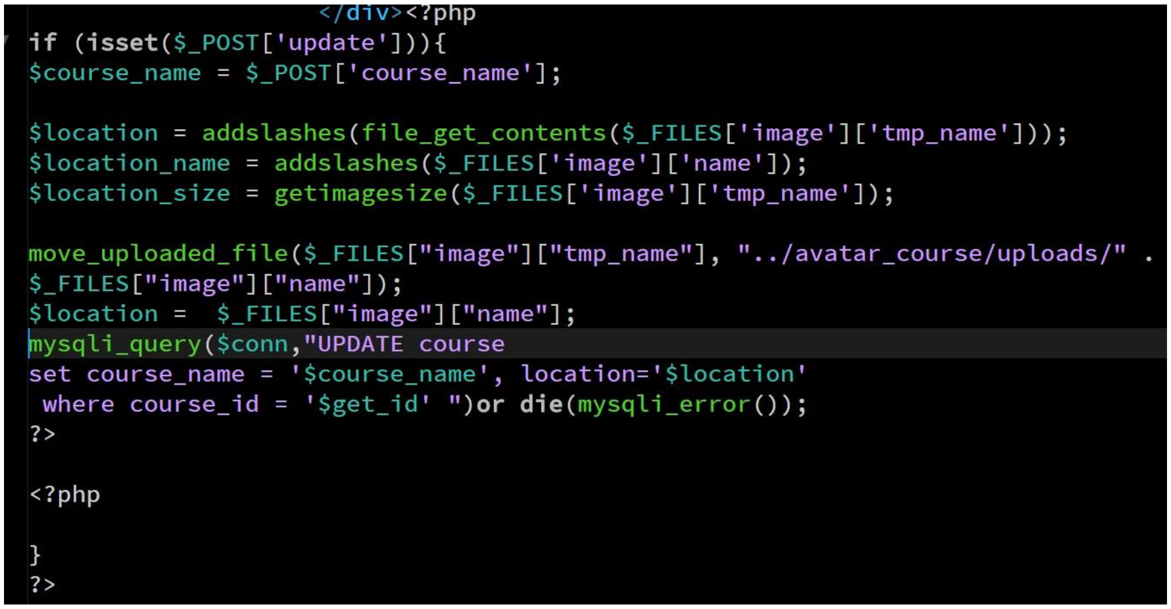
  Delete Material from db, based on id



```
$id = $_POST['id'];
mysqli_query($conn,"delete from files where file_id = '$id' ")or die(mysqli_error());
?>
```

- Edit Course

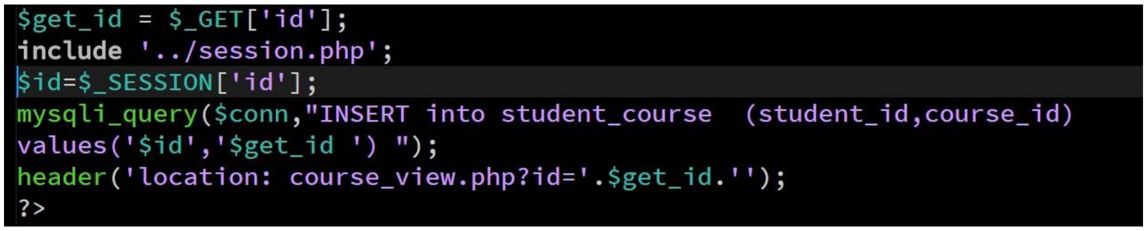
  Get new data and cover and update it on DB.



```
                        </div><?php
if (isset($_POST['update'])){
$course_name = $_POST['course_name'];

$location = addslashes(file_get_contents($_FILES['image']['tmp_name']));
$location_name = addslashes($_FILES['image']['name']);
$location_size = getimagesize($_FILES['image']['tmp_name']);

move_uploaded_file($_FILES["image"]["tmp_name"], "../avatar_course/uploads/" .
$_FILES["image"]["name"]);
$location =  $_FILES["image"]["name"];
mysqli_query($conn,"UPDATE course
set course_name = '$course_name', location='$location'
 where course_id = '$get_id' ")or die(mysqli_error());
?>

<?php

}
?>
```

- Subscribe Course

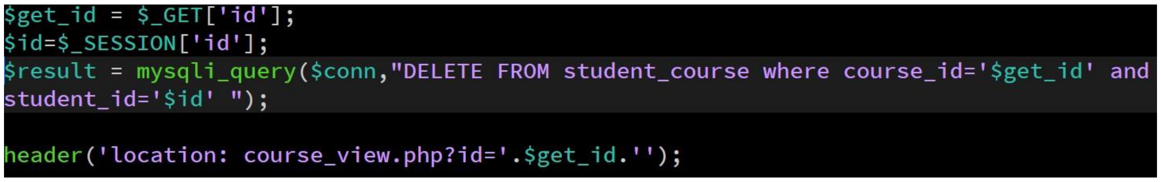
  Insert student id and course id on DB table to build correct relation (many to many).



```
$get_id = $_GET['id'];
include '../session.php';
$id=$_SESSION['id'];
mysqli_query($conn,"INSERT into student_course  (student_id,course_id)
values('$id','$get_id ') ");
header('location: course_view.php?id='.$get_id.'');
?>
```

- unsubscribe Course

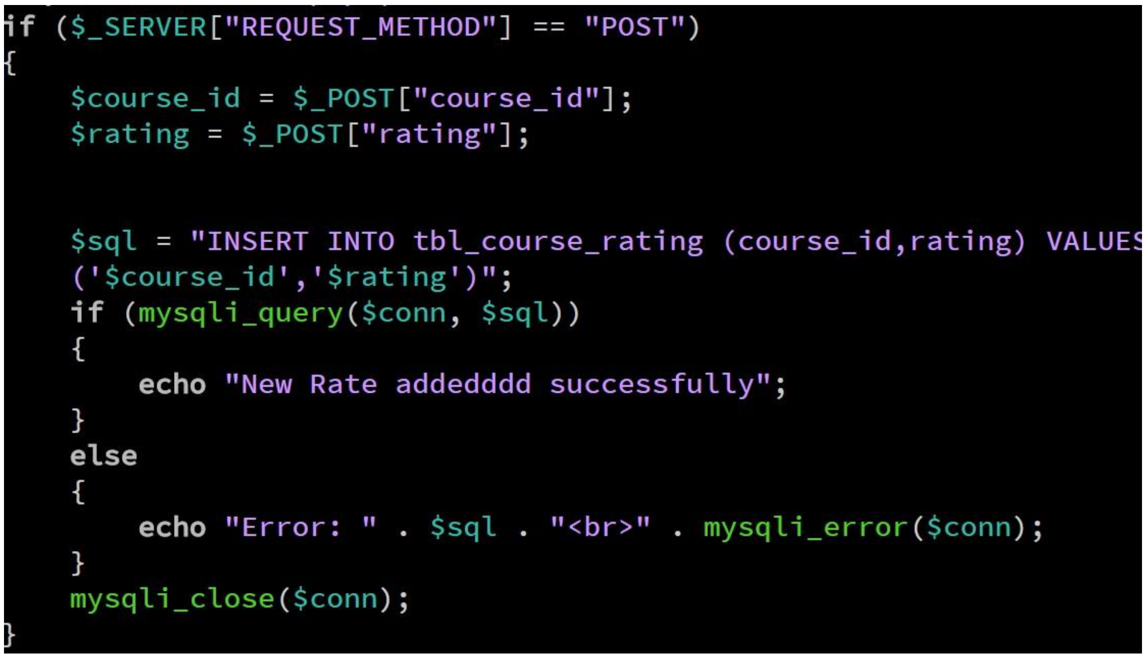
delete selected course from table to cancel student enroll course.



```
$get_id = $_GET['id'];
$id=$_SESSION['id'];
$result = mysqli_query($conn,"DELETE FROM student_course where course_id='$get_id' and
student_id='$id' ");

header('location: course_view.php?id='.$get_id.'');
```

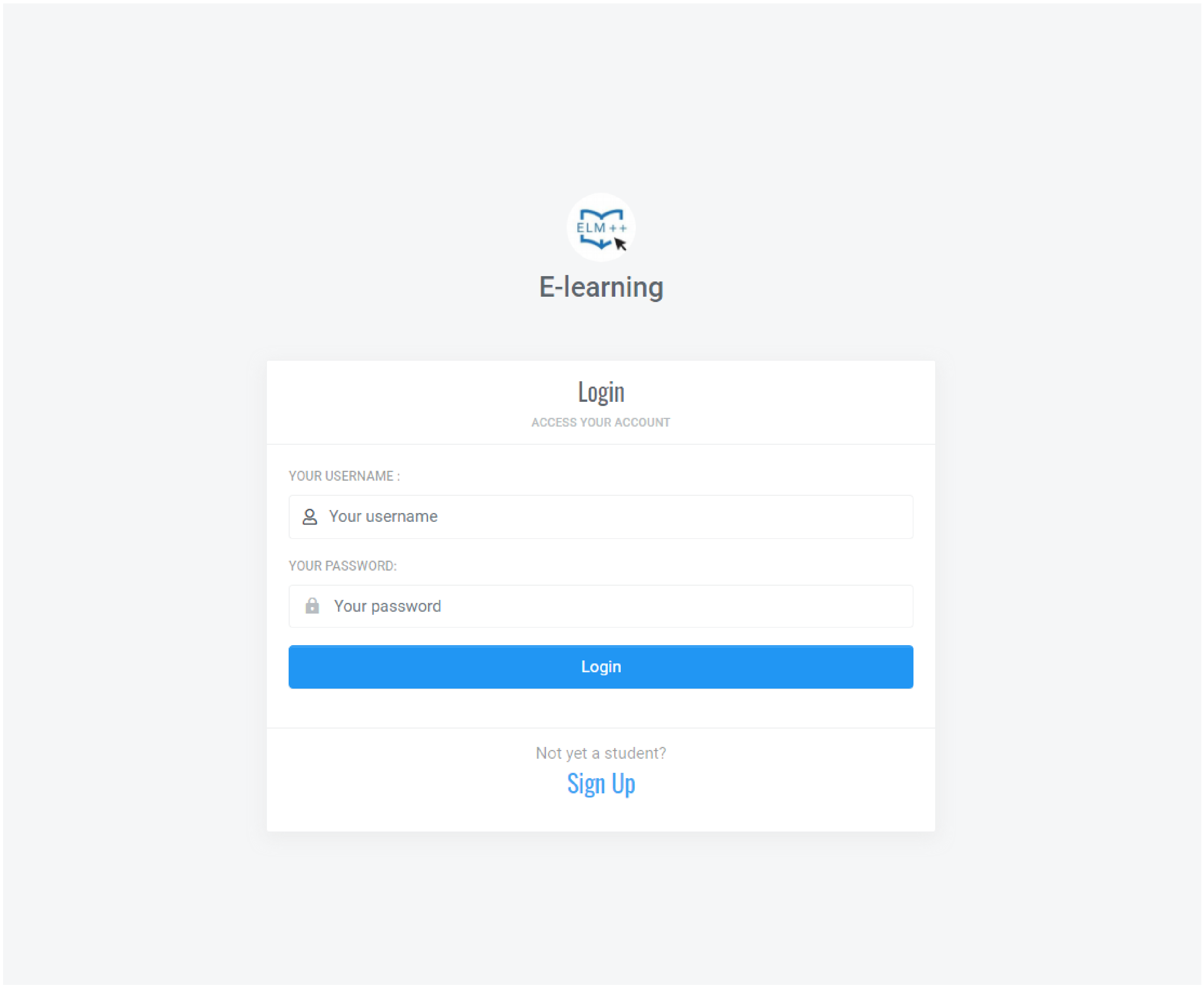
- Rate Course

Take course id, and new rate then add it to rate table.



```
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    $course_id = $_POST["course_id"];
    $rating = $_POST["rating"];


    $sql = "INSERT INTO tbl_course_rating (course_id,rating) VALUES
    ('$course_id','$rating')";
    if (mysqli_query($conn, $sql))
    {
        echo "New Rate addedddd successfully";
    }
    else
    {
        echo "Error: " . $sql . "<br>" . mysqli_error($conn);
    }
    mysqli_close($conn);
}
```

10.5 User Interface



E-learning

Login

ACCESS YOUR ACCOUNT

YOUR USERNAME :

Your username

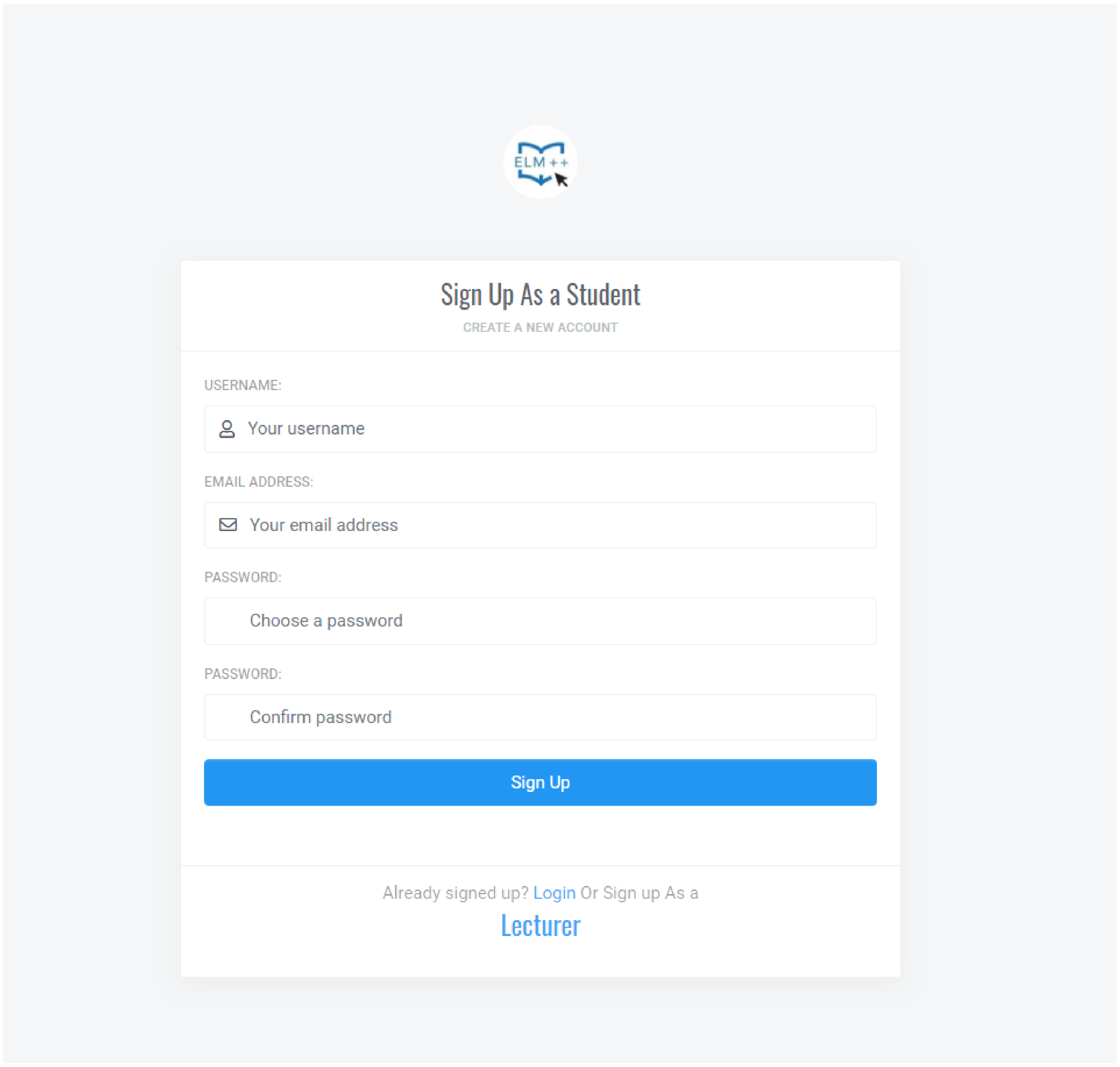
YOUR PASSWORD:

Your password

Login

Not yet a student?

Sign Up



Sign Up As a Student

CREATE A NEW ACCOUNT

USERNAME:

Your username

EMAIL ADDRESS:

Your email address

PASSWORD:

Choose a password

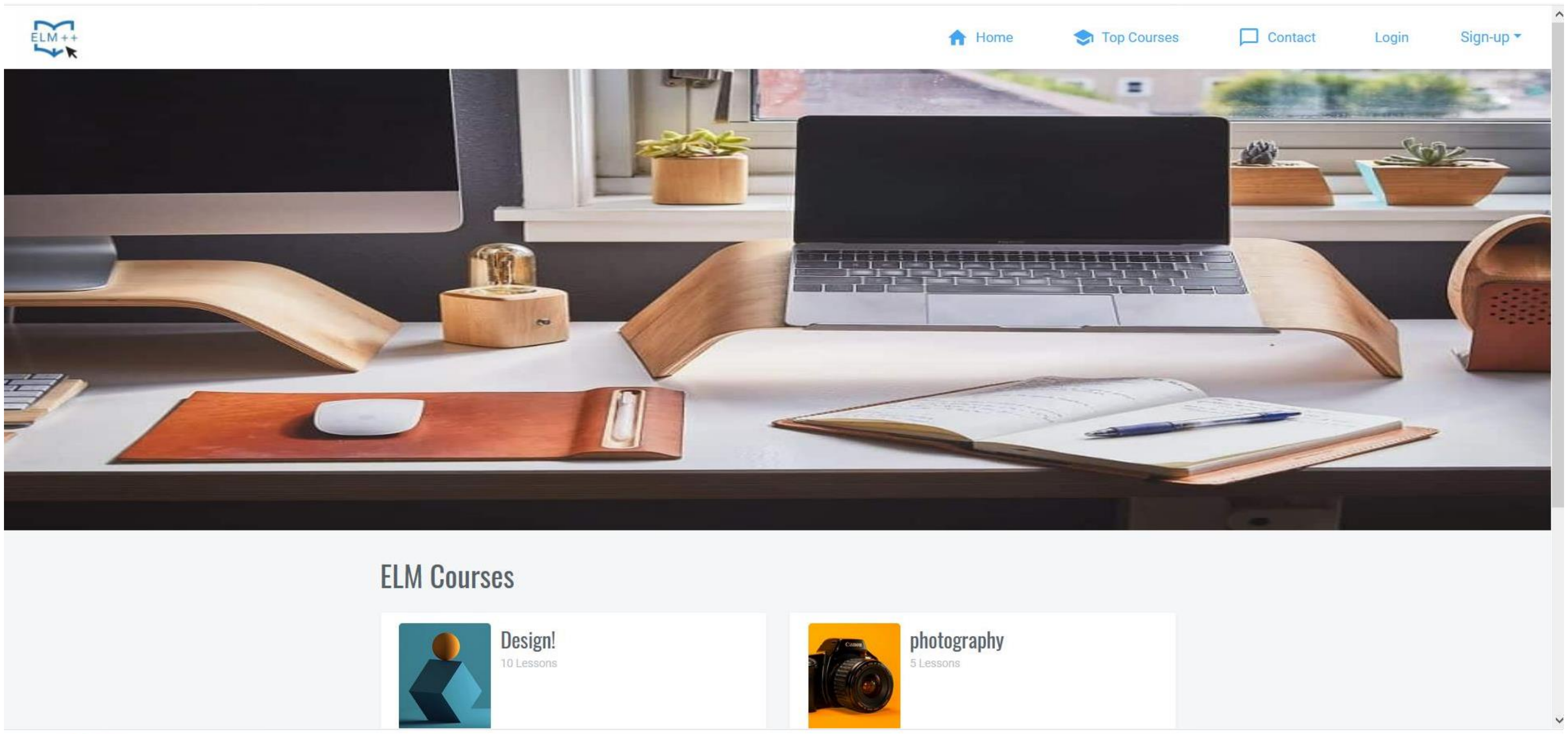
PASSWORD:

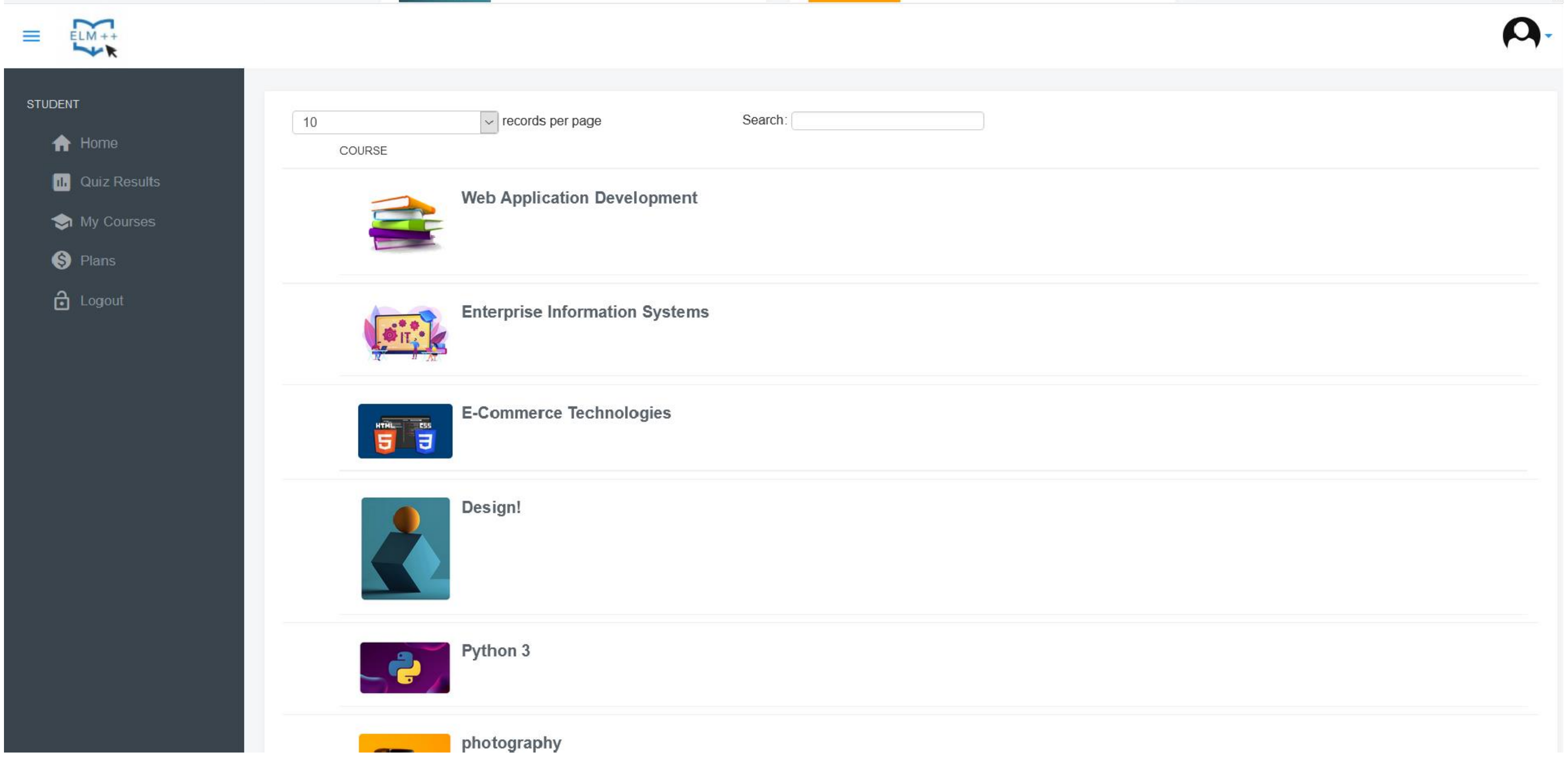
Confirm password

Sign Up

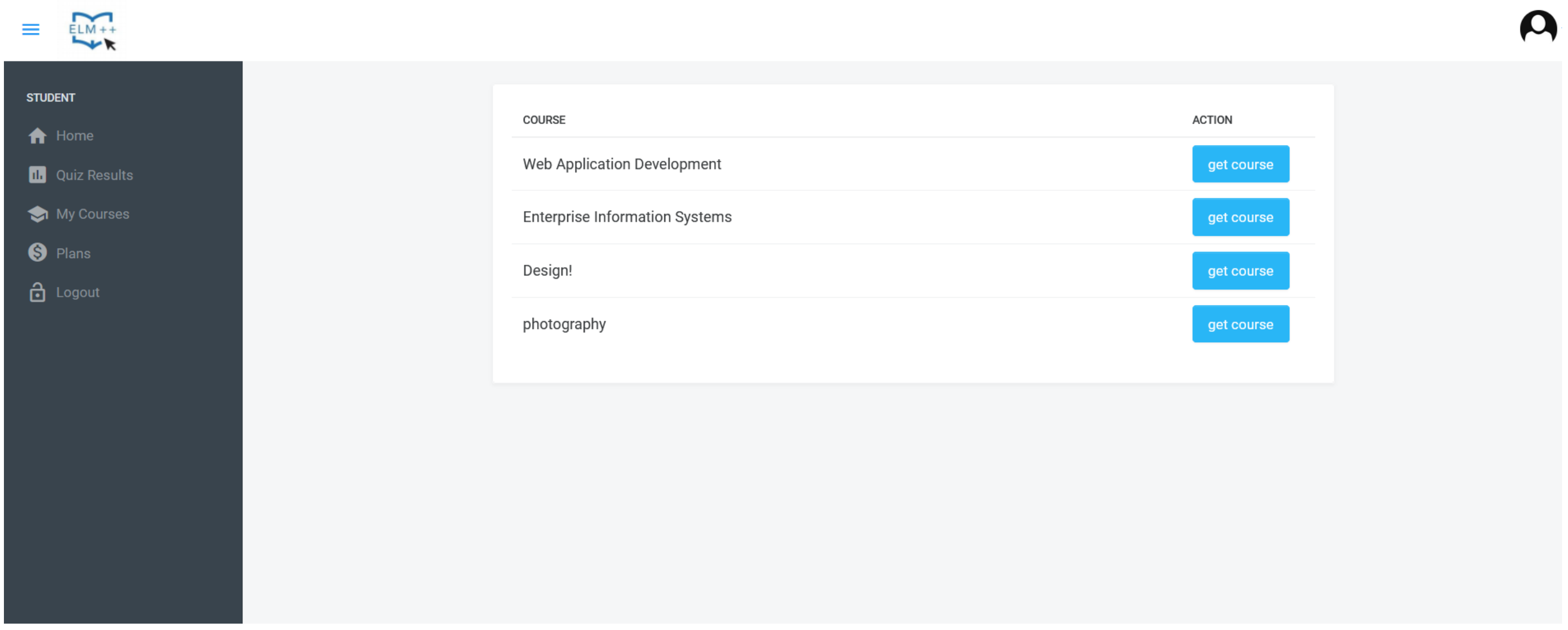
Already signed up? Login Or Sign up As a

Lecturer

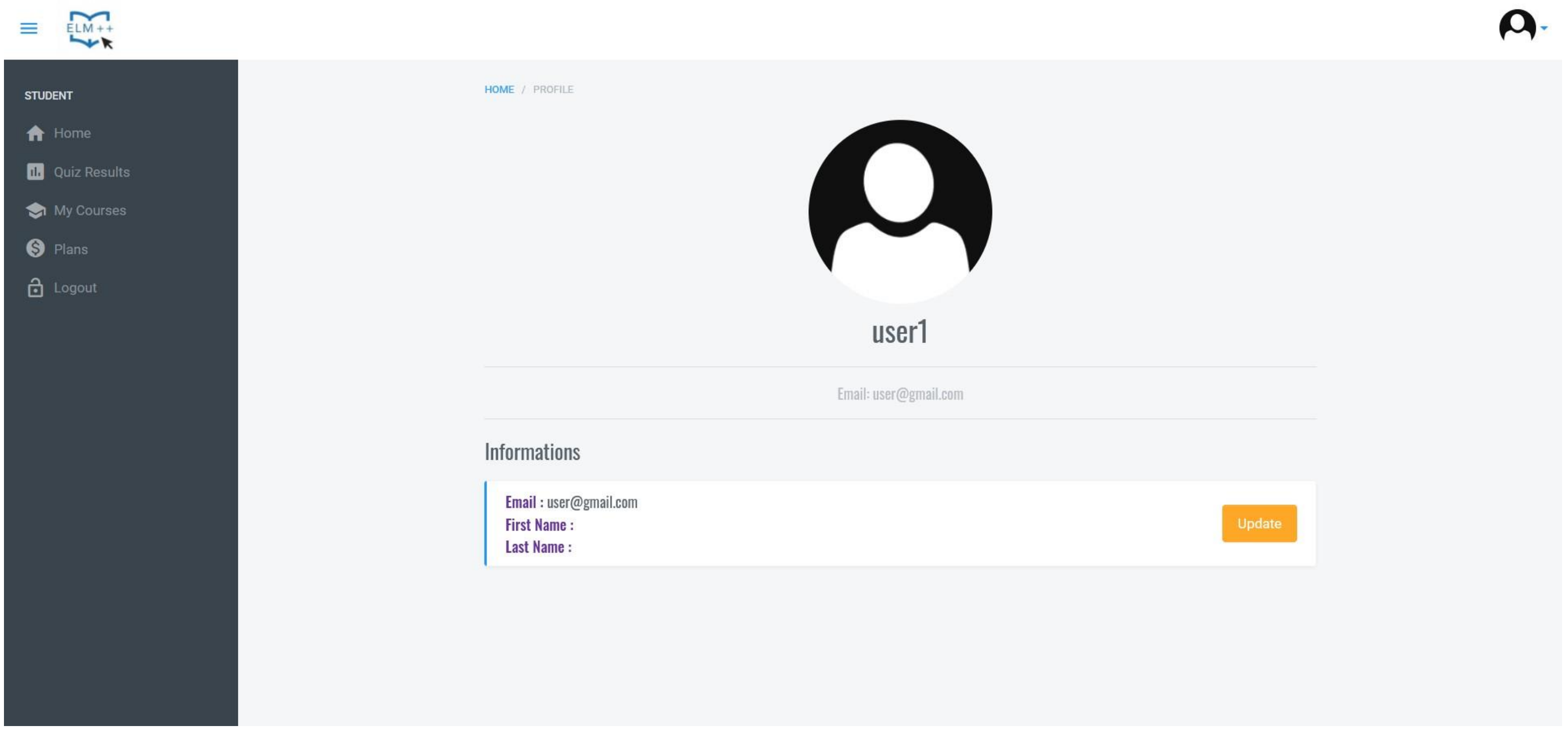




COURSE
Web Application Development
Enterprise Information Systems
E-Commerce Technologies
Design!
Python 3
photography



COURSE	ACTION
Web Application Development	get course
Enterprise Information Systems	get course
Design!	get course
photography	get course

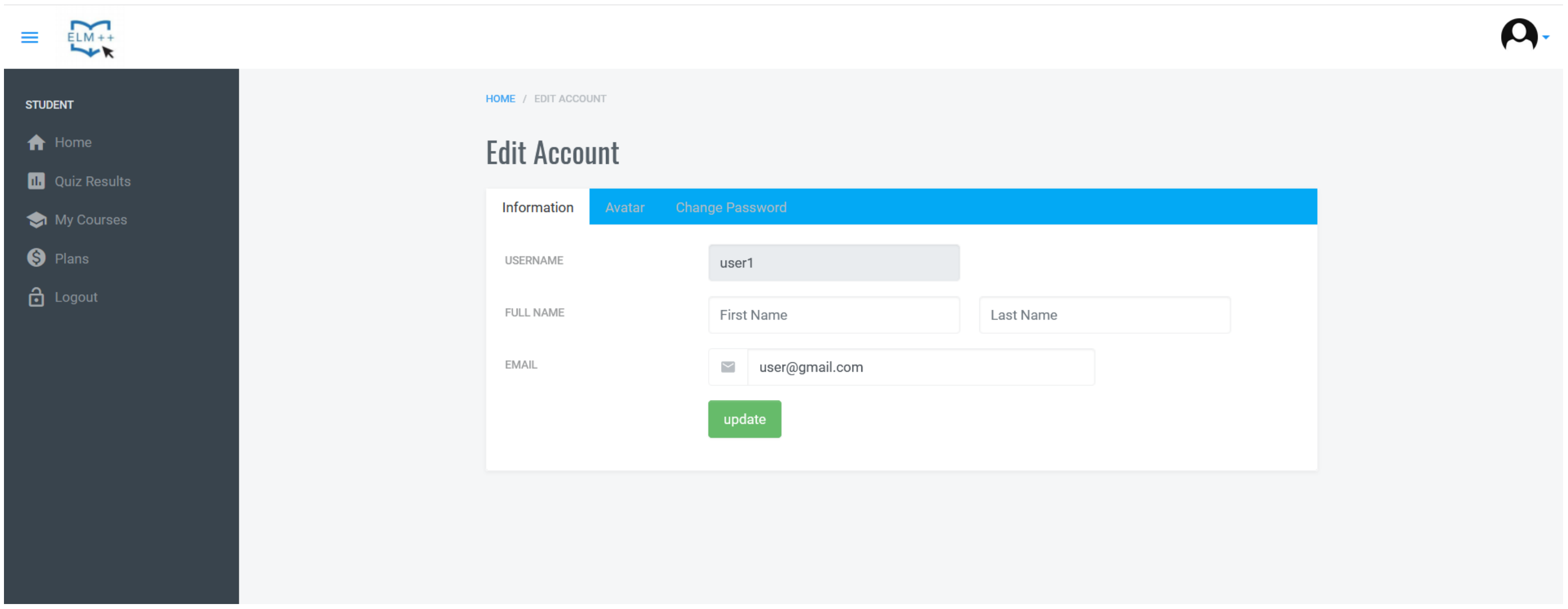


user1

Email: user@gmail.com

Informations

Email : user@gmail.com
First Name :
Last Name :


STUDENT
Home
Quiz Results
My Courses
Plans
Logout

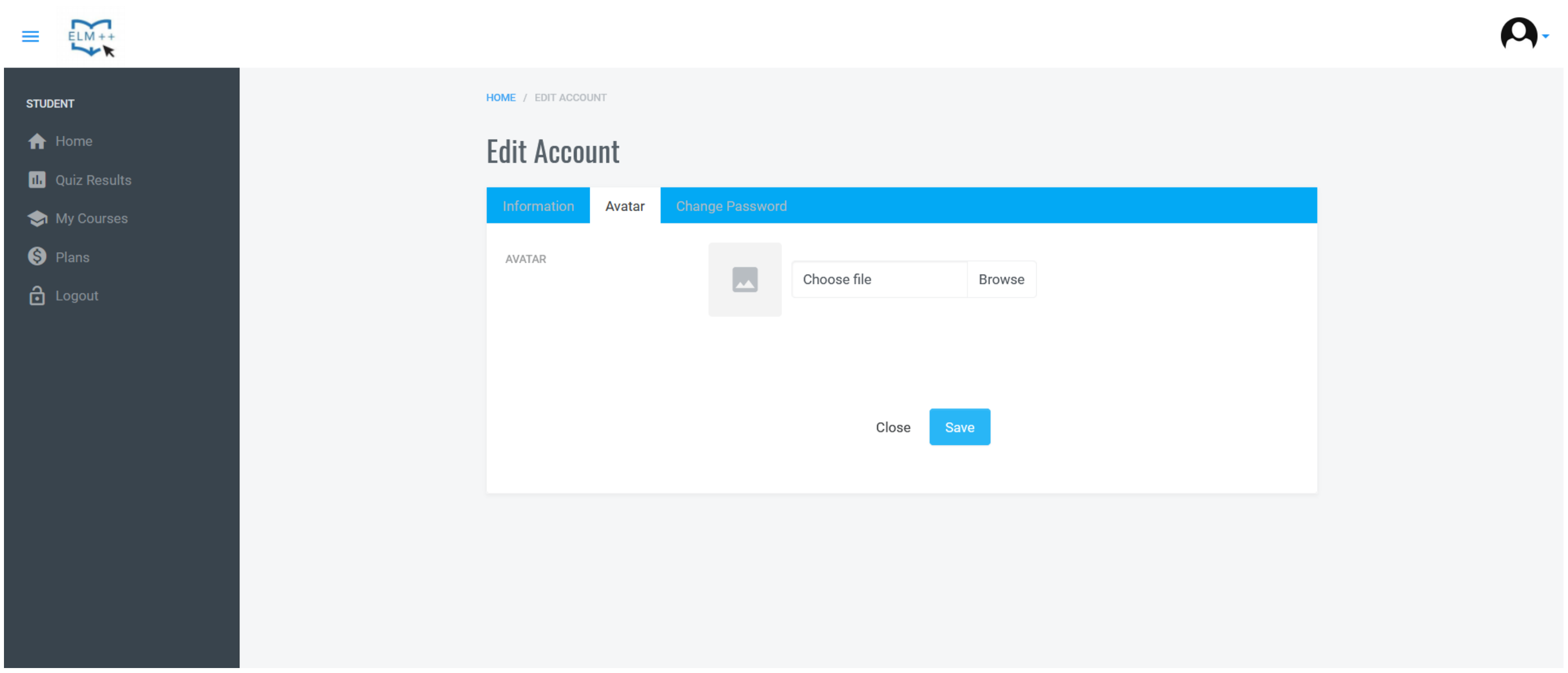
HOME / EDIT ACCOUNT

Edit Account

Information | Avatar | Change Password

USERNAME	user1	
FULL NAME	First Name	Last Name
EMAIL	user@gmail.com	

update


STUDENT
Home
Quiz Results
My Courses
Plans
Logout

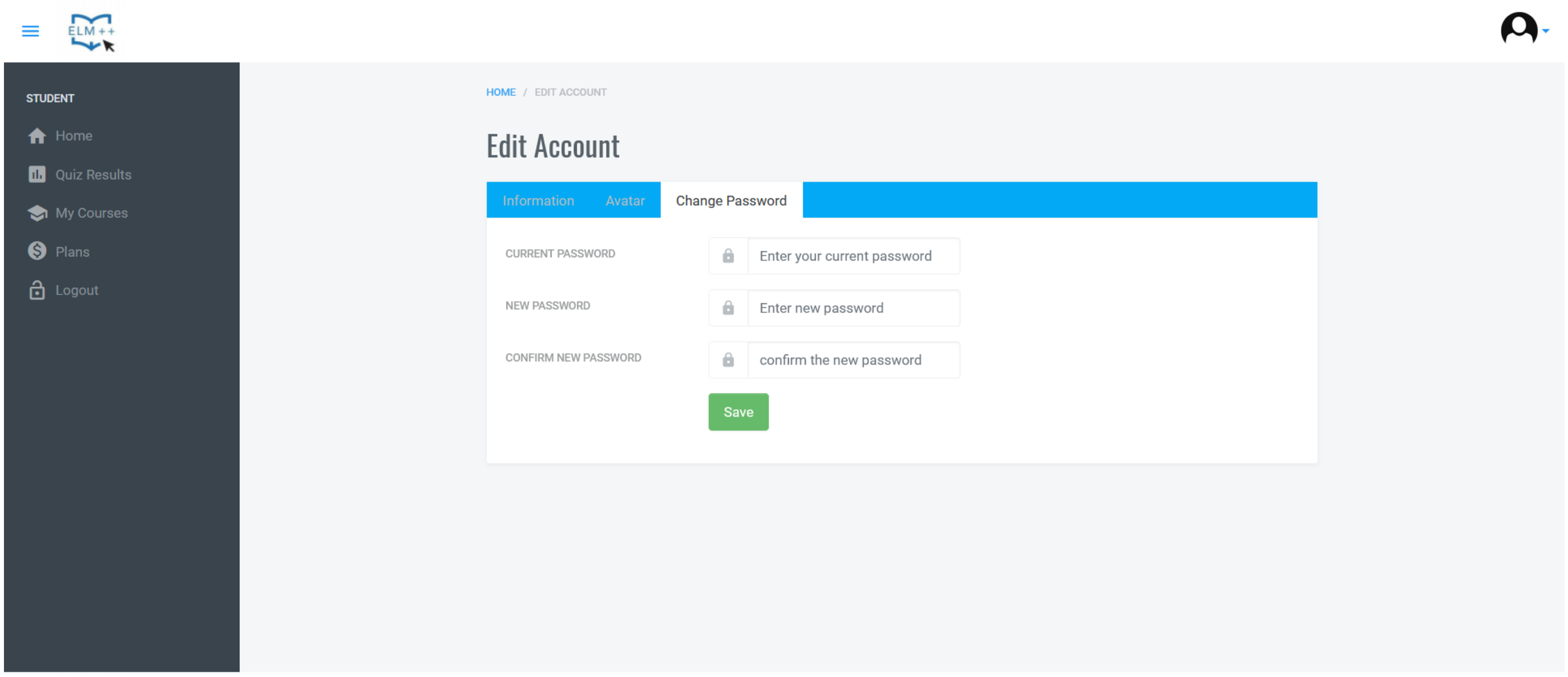
HOME / EDIT ACCOUNT

Edit Account

Information | Avatar | Change Password

AVATAR   Choose file   Browse

Close   Save



ELM++

STUDENT

Home

Quiz Results

My Courses

Plans

Logout

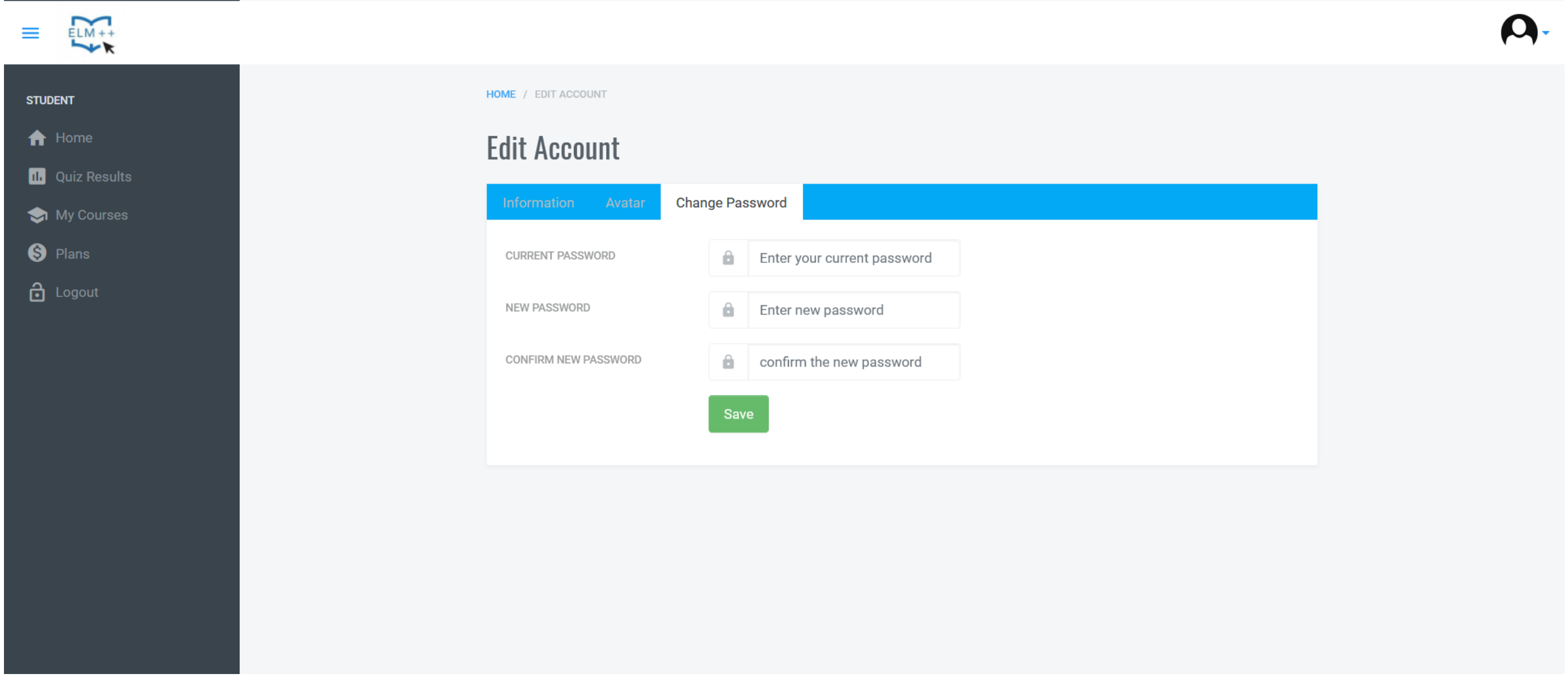
HOME / EDIT ACCOUNT

Edit Account

Information | Avatar | Change Password

CURRENT PASSWORD	Enter your current password
NEW PASSWORD	Enter new password
CONFIRM NEW PASSWORD	confirm the new password

Save



ELM++

STUDENT

Home

Quiz Results

My Courses

Plans

Logout

HOME / EDIT ACCOUNT

Edit Account

Information | Avatar | Change Password

CURRENT PASSWORD	Enter your current password
NEW PASSWORD	Enter new password
CONFIRM NEW PASSWORD	confirm the new password

Save

11. Testing

11.1 Test Scenarios

11.1.1 Sign up.

Test case number	input	output

1.1	email = "mohammed@gmail.com "
	
	username = "Mohammed”
	
	Password = "MoHaMmed12"
	
	Re password = " MoHaMmed12"
	
	Account type = "Student"
		Account created successfully.

1.2	email= "mohammed@gmail.com "
	
	username = " Mohammed”
	
	Password = " MoHaMmed"
	
	Re password = " MoHaMmed”
	
	Account type = "Lecturer"
		error massage.
		
		Password did not have numbers.

1.3	email = "nasser@gmail.com "
	
	username = “Mohammed’
	
	Password = " MoHaMmed12"
	
	Re password = " “
	
	Account type = "Student"
		error massage.
		
		All fields should be filled.

11.1.2 Sign in

Test case number	input	output

1.1	email="mohammed@gmail.com"
	Password =" MoHaMmed12"
		signed in Successfully

1.2	email = "Mohammed"
	Password =" MoHaMmed12"
		Error message
		Wrong Email or password

1.3	email="mohammed@gmail.com"
	Password = ""
		Error message
		All fields should be filled

1.4	email="Mohammed@gmail.com"
	Password ="@(892#010"
		Error message
		Wrong Email or password

11.1.3 Search course

Testcase number	input	output
1.1	Search keyword = "math "	The result should appear successfully.
1.2	Search keyword = ""	Error massage. Please enter a key word to search.

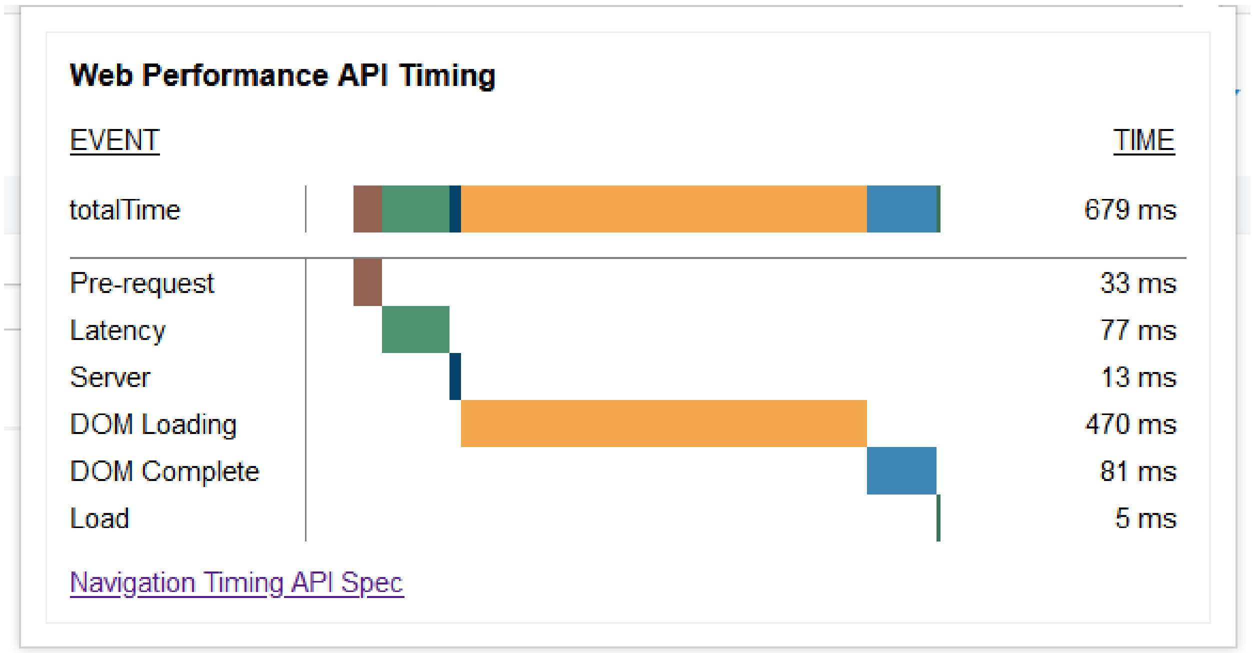
11.1.4 Create a Course

Test case number	input	output

1.1	Title = “Math 106”
	Description = “any text “
	Video = “Video 1”
		Course created successfully.

1.2	Title = “Math 106”
	Description = “”
	Video = “Video 1”
		Error massage.
		
		All fields should be filled

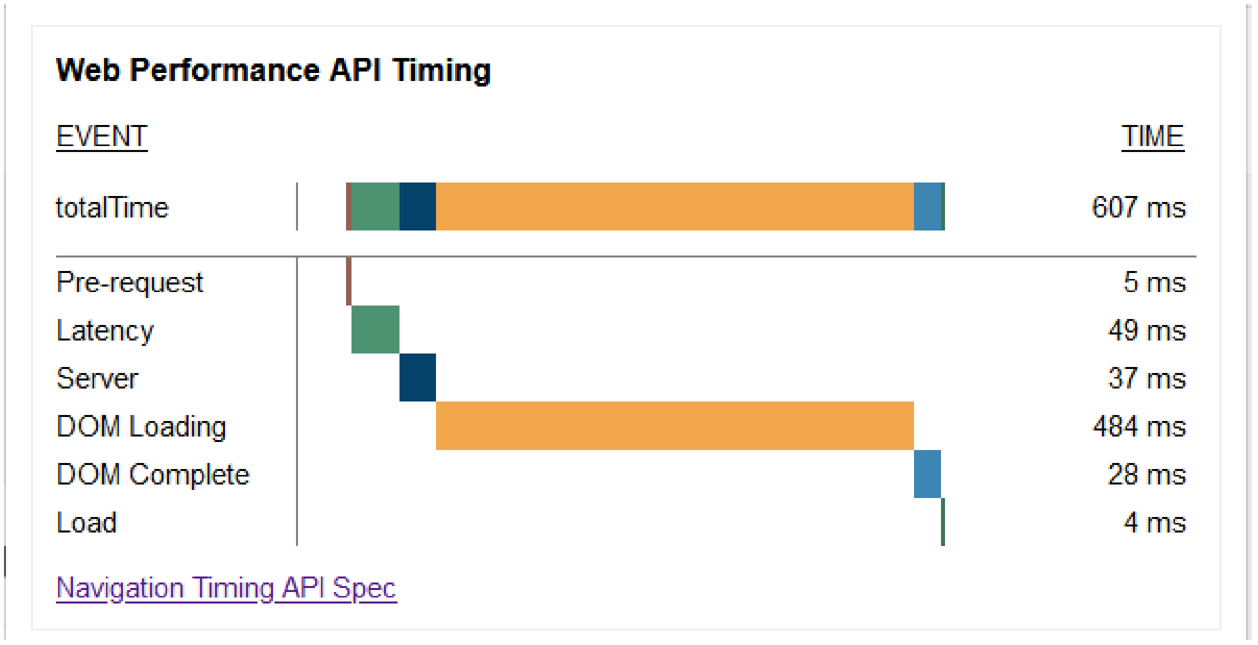
11.2 Performance test



Web Performance API Timing

EVENT	TIME
totalTime	679 ms
Pre-request	33 ms
Latency	77 ms
Server	13 ms
DOM Loading	470 ms
DOM Complete	81 ms
Load	5 ms

Navigation Timing API Spec


Web Performance API Timing

EVENT	TIME
totalTime	607 ms
Pre-request	5 ms
Latency	49 ms
Server	37 ms
DOM Loading	484 ms
DOM Complete	28 ms
Load	4 ms

Navigation Timing API Spec

11.3 Functional Test

This function test is based on the scenario testing.

Function to be tested: Add Course:

ID	Scenario	Expected Output	Preconditions	Actual Output	Test Status
1	Login to the System	Log User in and go to home Page	-	As Expected,	Pass
2	Press Add Course Button e.g." Web development"	It will appear in the Home Page	-	As Expected,	Pass

Function to be tested: Add Course Material:

ID	Scenario	Expected Output	Preconditions	Actual Output	Test Status
1	Login to the System	Log User in and go to home Page	-	As Expected,	Pass
2	Chose the Course You want to ADD Materials on, e.g. “Web development” Choose file By Browsing Your Computer, Enter File Name and Description.	Course Material Will Be ADDED and Redirect home Page	Couse has been Created	As Expected,	Pass

Function to be tested: Edit account:

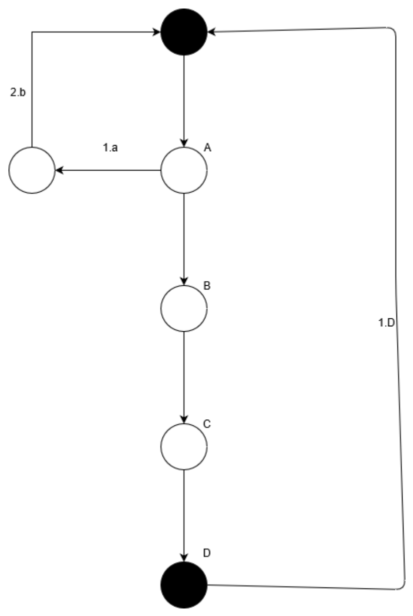
ID	Scenario	Expected Output	Preconditions	Actual Output	Test Status
1	Login to the System	Log User in and go to home Page	-	As Expected,	Pass
2	Click on account icon.	Pop list Will Appear	-	As Expected,	Pass
3	Press Edit Account, Edit one or more of the information fields and click Update.	The user’s information is updated	-	As Expected,	Pass

Branch coverage

Function to be tested: User login
Actor: User
Precondition: login
Steps:
A- System check if the user is logged in or no

1.a the user dose not log in
2.a the system will Appear a notification to the user to log in

B- The user enter log in information
C- send the information to the database
D- Save the information on the database
1.D the system will Appear "Logged in Successfully" a notification



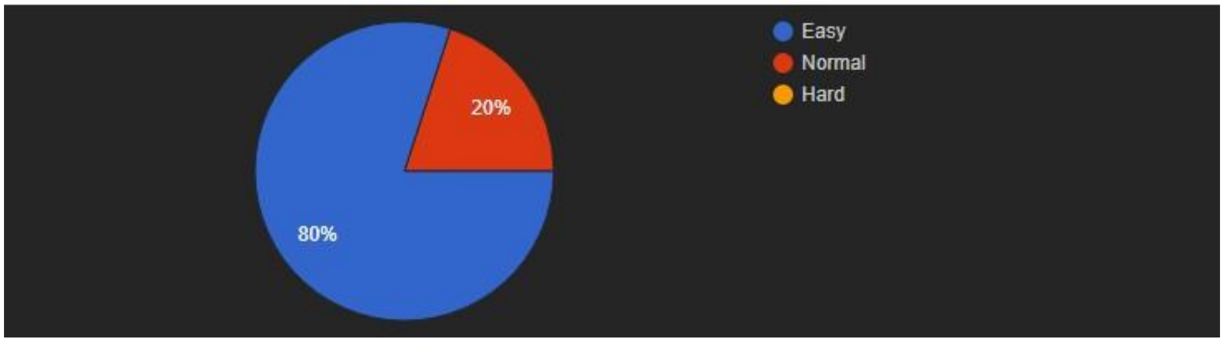
ID	Event	Description
1	A-B-D-C	This is The Basic scenario, the system will check if the user is logged in or no, then user will Fill his information, the system Will Send and Save the information in the Database
2	A – 1.a – 2.a-B-C-D-1.D	The system will display a notification to the user to let him know he logged in

11.4 Usability Test

We have made a google form and send it to our friends and colleagues to figure out how they are finding the useability of the main features.

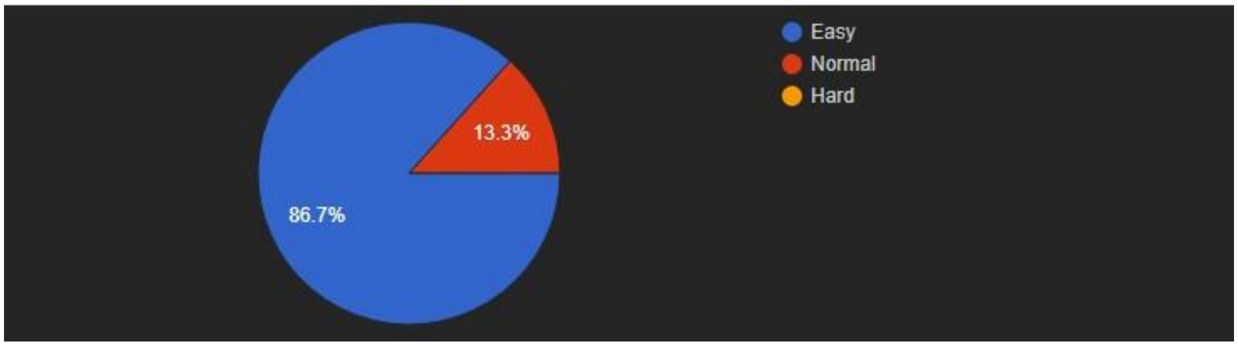
How do you find signing in in ELM

15 ردًا



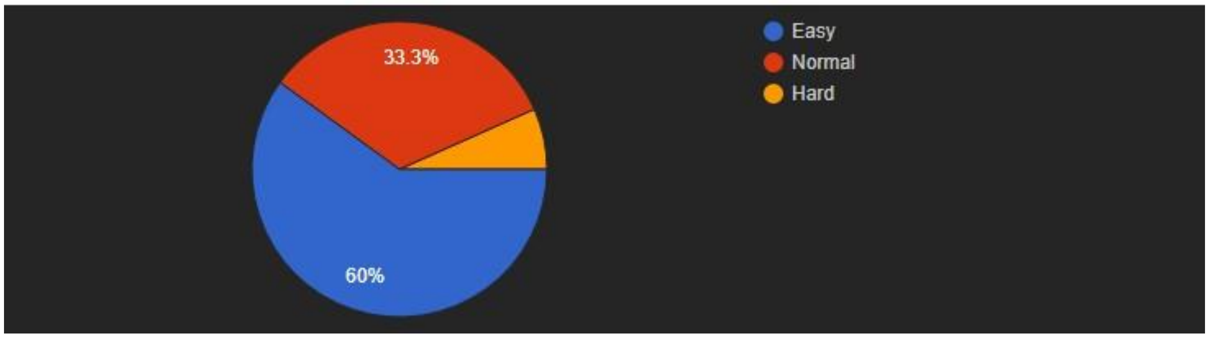
How do you find signing up in ELM

15 ردًا



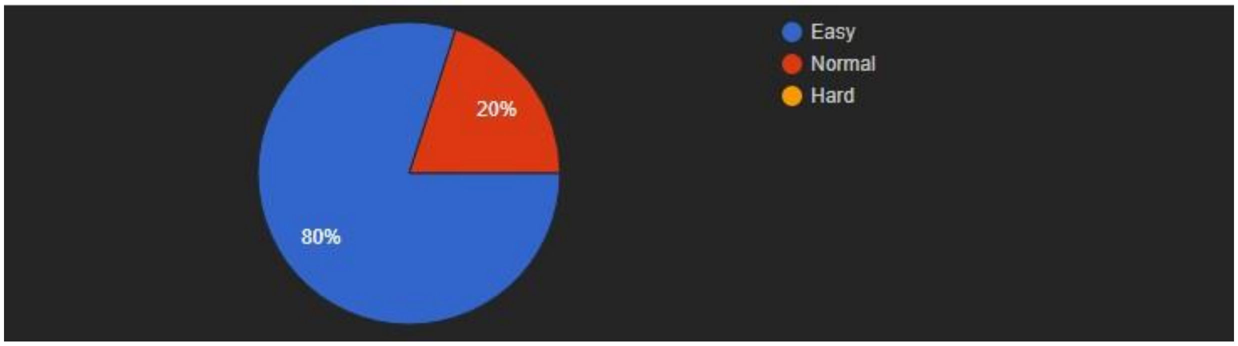
How do you find Creating a course in ELM

15 ردًا



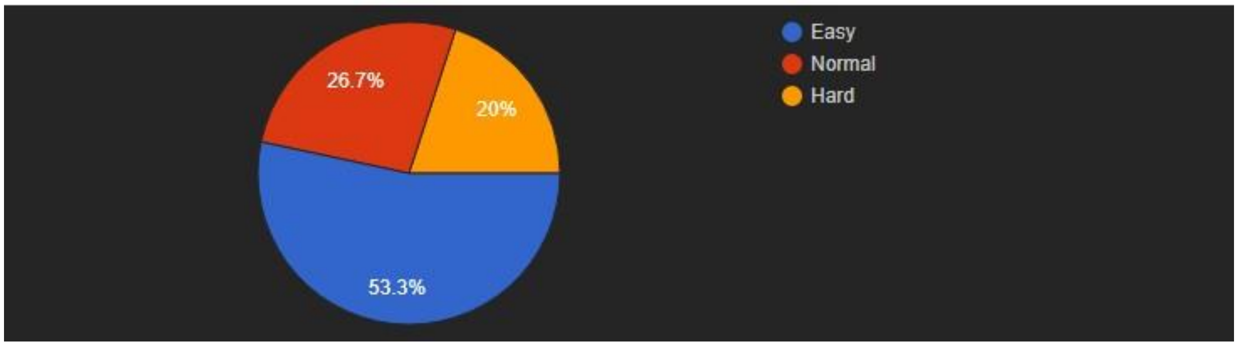
How do you find rating a course in ELM

15 ردًا



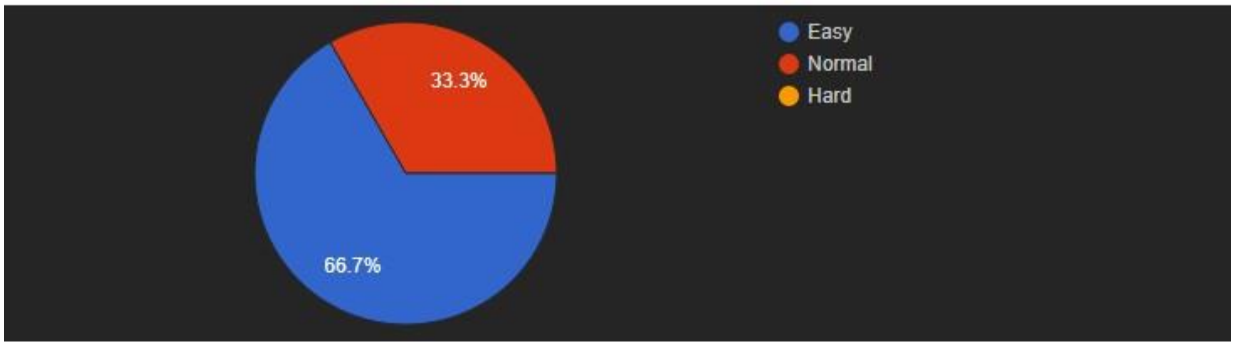
How do you find adding a material in a course in ELM

15 ردًا



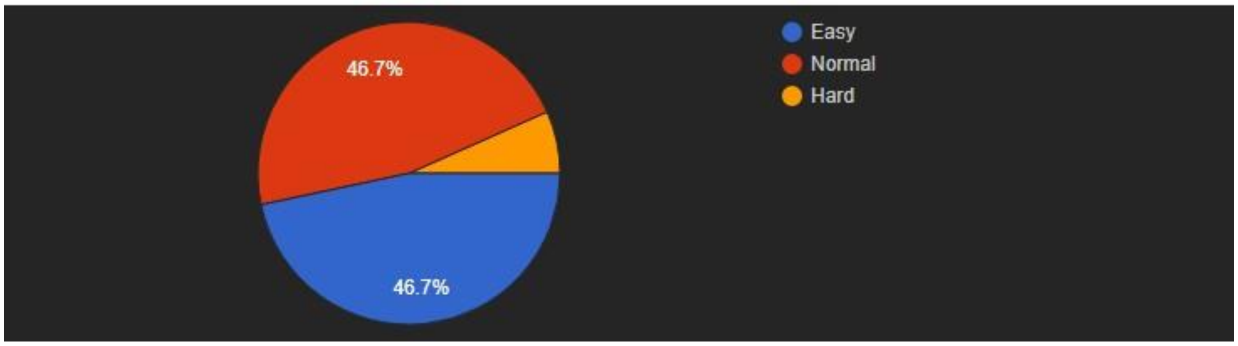
How do you find Editing your account in ELM

15 ردًا



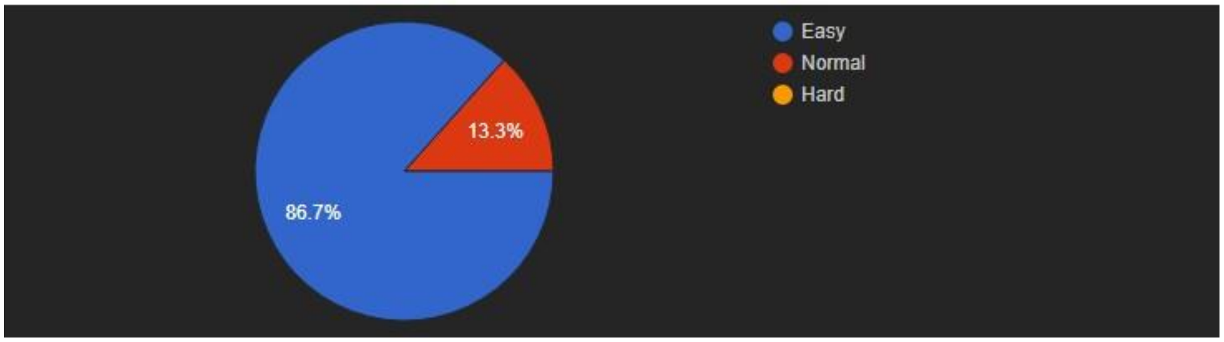
How do you find Conducting a quiz in ELM

15 ردًا



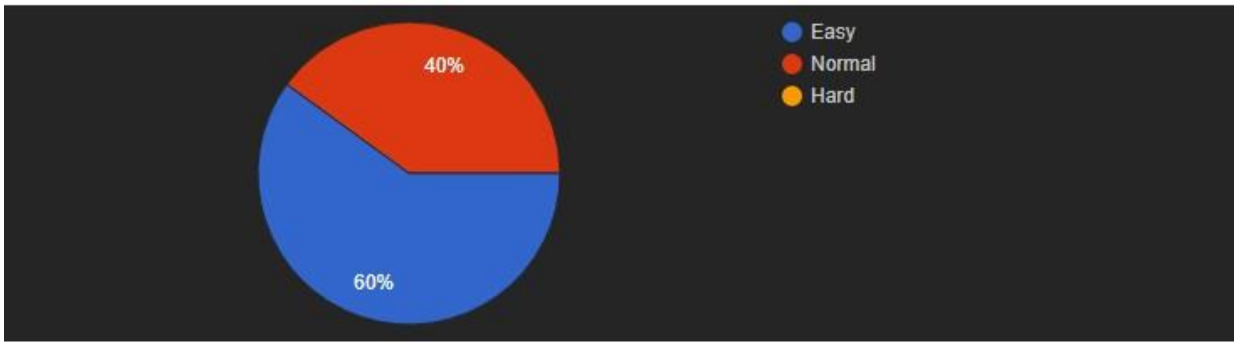
How do you find Subscribing to a course in ELM

15 ردًا



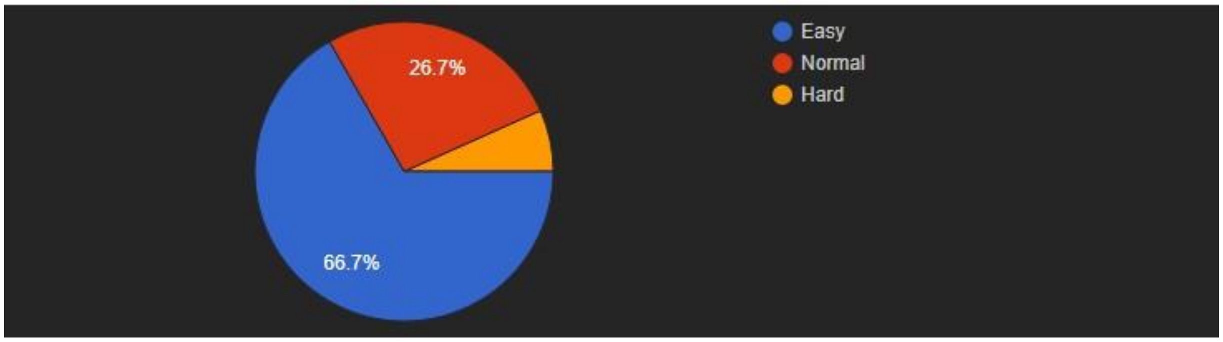
How do you find viewing your quiz result in ELM

15 ردًا



How do you find the payment in ELM

ردًا 15



12. Limitation of the System

12.1 Language issues

The website should have been translated to Arabic language but due to time limitation we have sticked to English language only.

12.2 Performance

The website is slow due to the free hosting, we did not have the enough resources to host it on a better server.

13. Conclusion and Future Work

In this report document we finalized the report 1 document, and added the actual look of the website plus some minor changes from report 1, we have conduct several testing methods , functional testing , scenario testing, unit testing, useability testing, and we also did the mapping between Requirements and the Implemented Functions , we have also listed the platforms and tools we used.

14. References

7. Reference

1. Coursera,https://www.coursera.org/, Accessed: September 2020.
2. Udemy,https://www.udemy.com/ , Accessed: September 2020.
3. Wea ,https://www.wea.org.uk/, Accessed: September 2020.
4. Shroo7,https://www.shroo7.com/, Accessed: September 2020.
5. Wikipedia,https://www.wikipedia.org/, Accessed: September 2020.
6.

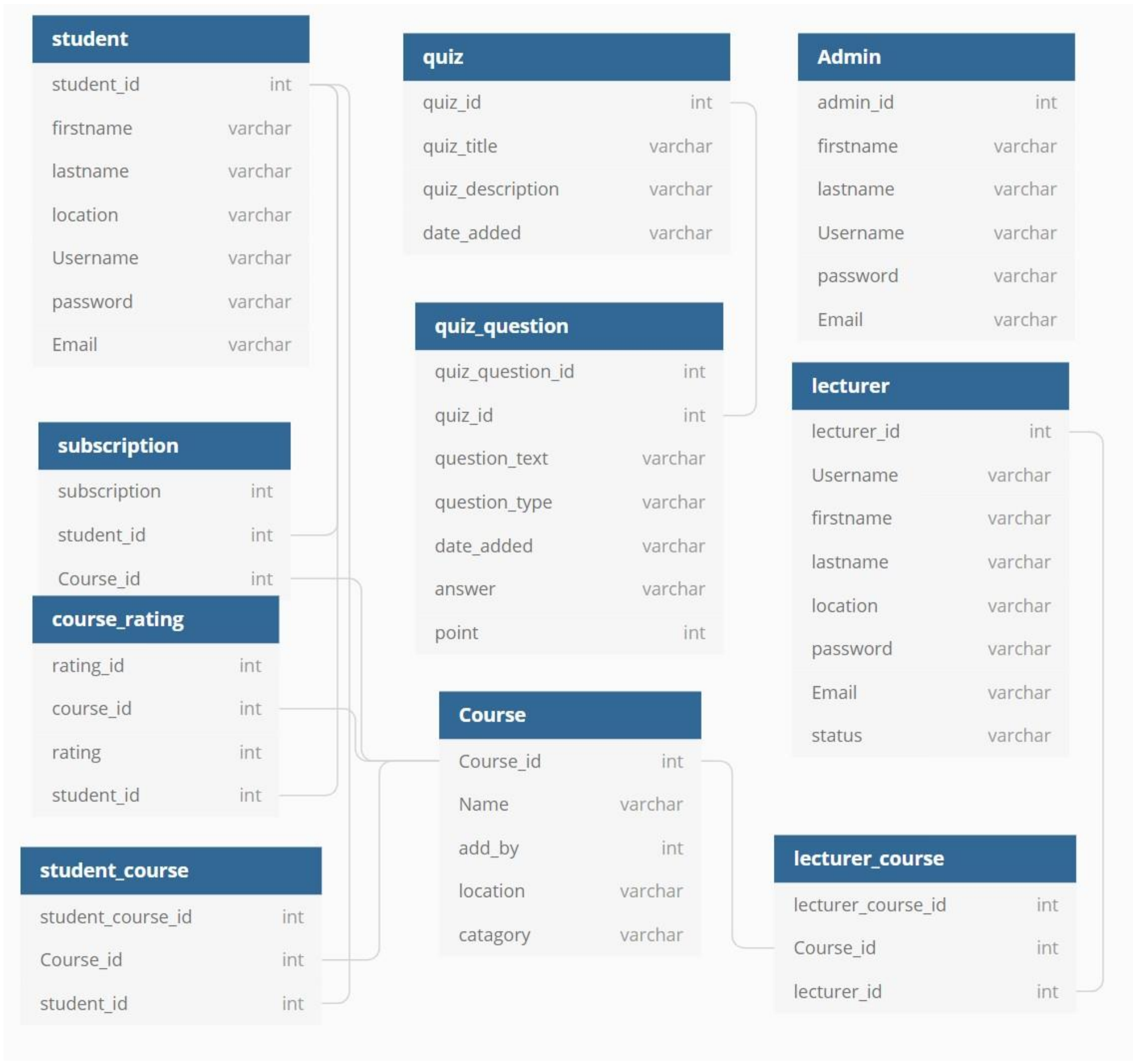
Appendix A. Functional Requirement Updates

Student shall be able to choose his payment method for paid courses. Removed.
Student shall be able to upgrade his account to premium. Added.
Student shall be able to print course certificate. Removed.
Student shall be able to report a course. Removed.

Appendix B. Non-Functional Requirement Updates

The system shall support Arabic and English languages. Not completed.

Appendix H. Database Schema Updates


student
student_id int
firstname varchar
lastname varchar
location varchar
Username varchar
password varchar
Email varchar
quiz
quiz_id int
quiz_title varchar
quiz_description varchar
date_added varchar
Admin
admin_id int
firstname varchar
lastname varchar
Username varchar
password varchar
Email varchar
quiz_question
quiz_question_id int
quiz_id int
question_text varchar
question_type varchar
date_added varchar
answer varchar
point int
lecturer
lecturer_id int
Username varchar
firstname varchar
lastname varchar
location varchar
password varchar
Email varchar
status varchar
subscription
subscription int
student_id int
Course_id int
course_rating
rating_id int
course_id int
rating int
student_id int
Course
Course_id int
Name varchar
add_by int
location varchar
catagory varchar
student_course
student_course_id int
Course_id int
student_id int
lecturer_course
lecturer_course_id int
Course_id int
lecturer_id int