The Egyptian E-Learning University

Faculty of Computers and Information Technology

Higher Educational Management System

(Graduation Project)

Prepared by

Ahmed Salah Anter	2001722
Ahmed Tamer Fathi	2001955
Fatma Ahmed Mohamed	2000821
Hossam Ahmed Ahmed	2002047
Menna Walid Fadel	2000435
Mina Magdy Helmy	2001929
Walaa Ashraf Hafez	1900984

SUPERVISED BY

Prof. Mahmoud Ashour TA. Zahraa Abdel Sattar

Higher Educational Management System (Graduation Project)

Project Outlines

1. Project overview

Introduction to the System

Name: Higher Educational Management System

Description: **The Higher Educational Management System revolutionizes education by streamlining administrative tasks and enhancing the learning experience. This online platform offers intuitive interfaces for students and teachers, enabling access to course materials, assignment submissions, and academic progress tracking. Teachers benefit from tools for lesson planning, grading, and communication. Administrators can manage user accounts, monitor analytics, and implement updates. The system promotes accessibility and features interactive forums and collaborative spaces, creating a community within the educational ecosystem. Its responsive design supports various devices, ensuring flexibility and inclusivity.**

Project Summary:

The Higher Educational Management System project introduces a groundbreaking approach to student learning and academic progress. By leveraging advanced analytics, the system provides detailed insights into students' performance, identifying specific areas of strength and weakness. This enables students to understand their mistakes and focus on improving their skills. Through personalized feedback and tailored learning paths, students are empowered to take control of their educational journey, leading to significant improvements in their academic achievements. This innovative system not only enhances administrative efficiency but also fosters a more responsive and adaptive learning environment, helping students continually improve based on their grades and feedback.

References:

Books on HTML and CSS

1. **HTML and CSS: Design and Build Websites by Jon Duckett**
2. **Head First HTML and CSS by Elisabeth Robson and Eric Freeman**
3. **HTML and CSS: The Good Parts by Ben Henick**

4. **Pro HTML5 Programming by Peter Lubbers, Brian Albers, and Frank Salim**

5. **Introducing HTML5 by Bruce Lawson and Remy Sharp**

6. **HTML5 for Web Designers by Jeremy Keith**

7. **HTML, CSS, and JavaScript All in One by Julie C. Meloni and Jennifer Kyrnin**

Books on PHP and MySQL

8. **Learning PHP, MySQL & CSS by Robin Nixon**

9. **PHP and MySQL for Dynamic Web Sites by Larry Ullman**

10. **Modern PHP: New Features and Good Practices by Josh Lockhart**

11. **PHP and MySQL Novice to Ninja by Kevin Yank**

12. **Murach's PHP and MySQL by Joel Murach and Ray Harris**

13. **PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide by Larry Ullman**

Official Flutter Documentation

14. **Flutter Documentation: Comprehensive guides, tutorials, and API references. Flutter Documentation**

Online Sources for Web and Flutter Development

15. **Flutter Bloc Package: Flutter Bloc on pub.dev**

16. **Flutter Slider Drawer Package: Flutter Slider Drawer on pub.dev**

17. **MVVM Architecture Search Results: Google Search Results**

18. **MVVM Clean Architecture Pattern in Android with Use Cases: Medium Article by Ami Patel**

19. **Stack Overflow: A valuable resource for problem-solving and learning from the community in web development. Stack Overflow**

Problems and Solutions

Problem 1: Inefficient Course and Resource Management

Description: Traditionally, managing course schedules, enrollment, and allocation of resources (like lab equipment or software) can be cumbersome and prone to errors.

Solution: The system automates and centralizes the management of course schedules, enrollment processes, and resource allocation. Features like real-time schedule updates and inventory tracking reduce administrative overhead and minimize scheduling conflicts or resource shortages.

Problem 2: Difficulty in Tracking Academic Progress and Performance

Description: Faculty members often struggle to effectively monitor and support individual student progress, especially in large classes.

Solution: The system includes advanced analytics tools that track student performance in real-time, allowing faculty to identify students who may need additional support. It also enables personalized feedback and academic advising, enhancing student learning outcomes.

Problem 3: Limited Personalized Learning Opportunities

Description: Computer science students often require personalized learning paths, especially in areas like programming languages, algorithms, or specific technology stacks.

Solution: The system offers adaptive learning modules and recommends resources and courses based on individual student performance and preferences. This feature aids students in focusing on areas where they need improvement and exploring topics of interest more deeply.

Problem 4: Challenges in Research and Project Management

Description: Managing and tracking research projects, especially collaborative or interdisciplinary ones, can be complex and disorganized.

Solution: The system includes a research management module that helps in organizing, tracking, and reporting research activities. It facilitates collaboration with internal and external partners, manages deadlines, and keeps track of research outputs.

System Features and Capabilities

Core Functionalities

Course Management:

Automated Scheduling: Facilitates the creation and modification of class schedules, avoiding conflicts and optimizing room and resource usage.

Enrollment and Waitlist Processing: Manages course enrollments, including capacity limits and waitlist handling.

Student Information System:

Student Records: Maintains comprehensive records of student demographics, academic history, and progress.

Performance Analytics: Offers insights into student grades, participation, and overall academic trends.

Faculty Management:

Faculty Profiles: Keeps track of faculty qualifications, courses taught, and research interests.

Workload and Performance Tracking: Monitors teaching loads, research activities, and performance evaluations.

Advanced Features

Personalized Learning and Skill Analysis:

Adaptive Learning Paths: Recommends courses and resources based on individual student performance and preferences.

Skill Gap Analysis: Identifies areas where students need improvement, especially in programming skills, algorithmic thinking, and other core computer science competencies.

Research and Project Management:

Project Tracking Tools: Manages timelines, milestones, and collaboration for student and faculty research projects.

Publication and Grant Management: Assists in tracking publications, submissions, and grant applications.

Resource Allocation:

Inventory Management: Tracks and manages lab equipment, software licenses, and other educational resources.

Budget Management Tools: Provides real-time insights into budget allocation, expenditure, and financial planning.

Interactive and Collaborative Tools

Communication Platforms:

Integrated Messaging and Forums: Facilitates communication between students, faculty, and administrators.

Collaboration Spaces: Offers virtual environments for group work, project collaboration, and peer-to-peer learning.

E-Learning and Virtual Labs:

Online Course Materials: Provides access to lectures, readings, and other learning resources.

Virtual Lab Environments: Enables students to practice programming and technical skills in a controlled, simulated environment.

Administrative and Reporting Capabilities

Data Reporting and Analytics:

Custom Reports: Generates reports for academic performance, resource utilization, and administrative efficiency.

Data Visualization: Presents complex data in an easy-to-understand format, aiding in decision-making.

Compliance and Accreditation Support:

Regulatory Compliance Monitoring: Ensures the department adheres to educational standards and regulations.

Accreditation Documentation: Assists in the preparation of documents and reports for accreditation purposes.

User Experience and Accessibility

User-Centric Design:

Intuitive Interface: Offers a user-friendly interface that is easy to navigate for all user groups.

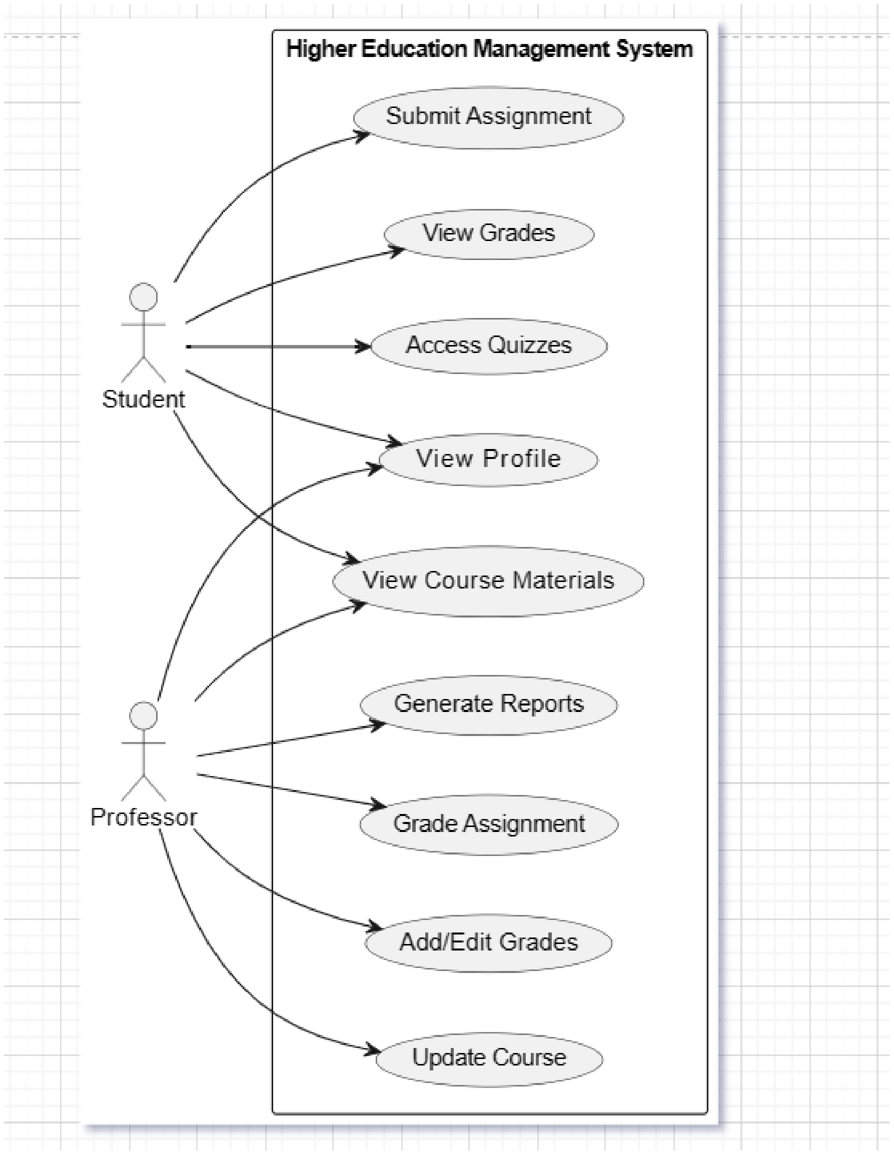
Accessibility Features: Ensures the system is accessible to users with disabilities, complying with accessibility standards.

Mobile Compatibility:

Mobile Access: Provides a mobile version or app for accessing the system on-the-go, enhancing flexibility and convenience.

Here's the Use case diagram of the project:

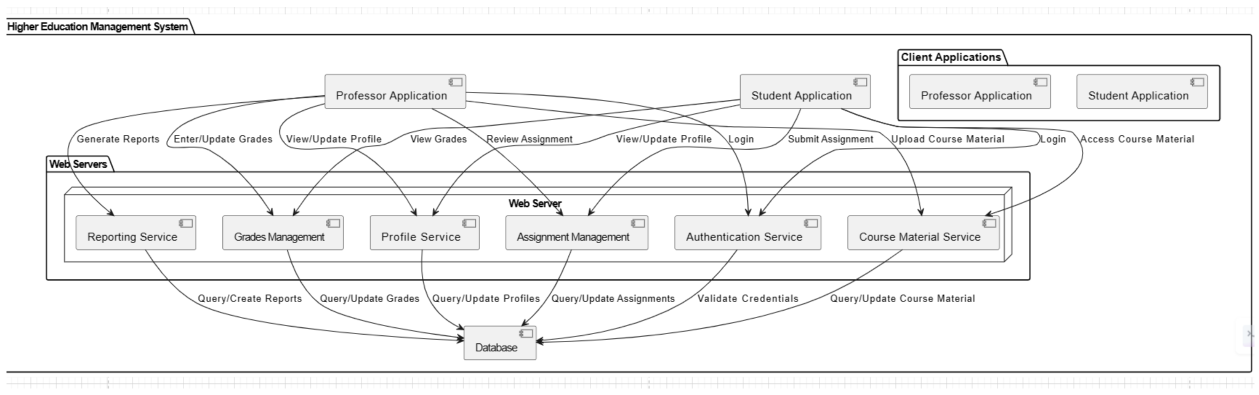
as it will help illustrate how users interact with the system and the various functionalities it offers.



Technical Aspects

1. Technical Architecture

And this is the System Architecture Diagram of the project:



This diagram will provide an overview of the system's structure, helping to understand the overall architecture before diving into details.

Overall Structure

The system has three main parts: the frontend, backend, and database.

Frontend: Created with HTML, CSS, and JavaScript, this is the user interface that is easy to use and works on different devices.

Backend: Built with PHP and Laravel using MVC (Model-View-Controller) architecture, this handles the main functions, user login, data processing, and connects the frontend to the database.

Mobile App: Developed with Flutter, it works on both iOS and Android, offering the same functions as the web version.

Database

Relational Database: MySQL is used for its reliability and ability to handle complex data.

NoSQL Database: MongoDB might be used for flexible data storage needs, like unstructured data or logs.

Cloud-based Environment

Hosting the system on cloud platforms like AWS, Google Cloud, or Azure provides scalability, reliability, and easy maintenance. Cloud services support database hosting, file storage (like Amazon S3), and backup, ensuring the system is always available and protected against data loss.

Integration with Existing Systems

Compatibility: The system works well with existing technologies in schools, including old systems, third-party tools, and common educational software.

Data Exchange and APIs: APIs allow the system to exchange data with other software like student information systems and learning management systems using standard methods like REST and data formats like JSON or XML.

User Management: The system can use existing login systems (like LDAP or Active Directory) so users can log in with their current credentials. Single Sign-On (SSO) allows users to access multiple systems with one login.

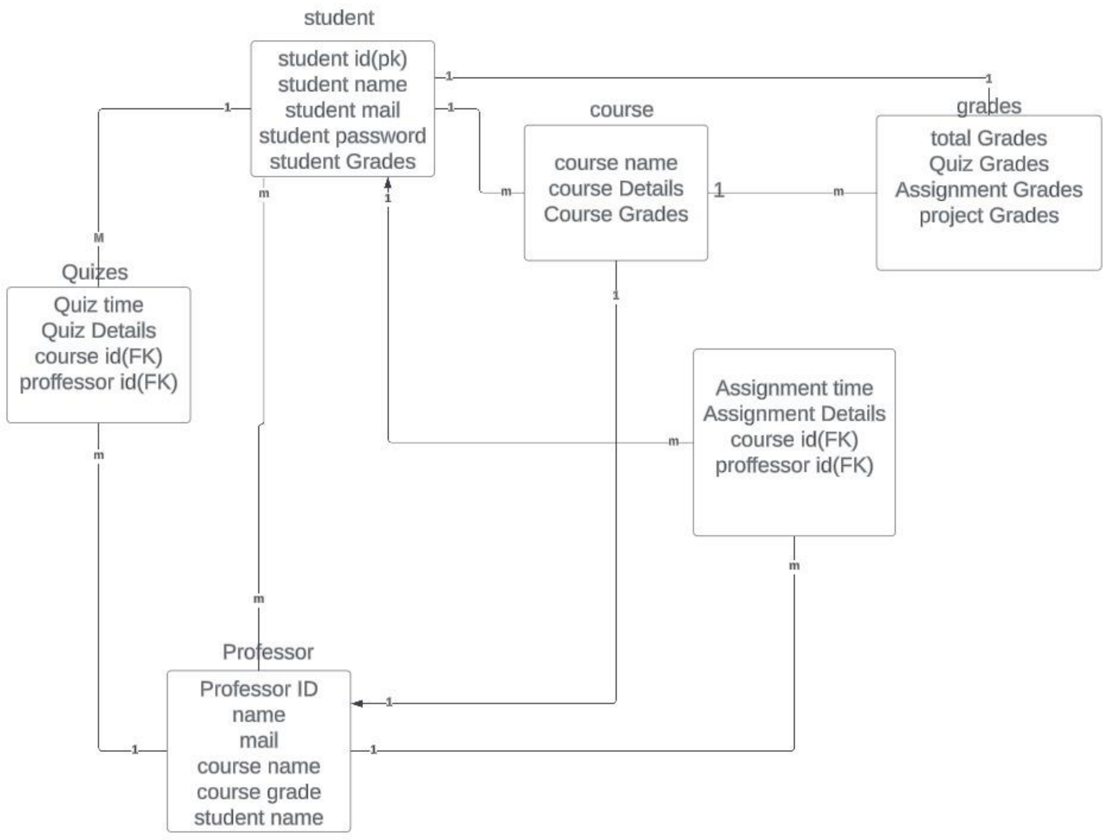
Reporting and Analytics: The system can integrate with existing reporting tools and analytics platforms to help schools use data effectively for decision-making.

Scalability and Modular Design

The system is designed to easily add or change features without disrupting current functions. It can grow or shrink based on the needs of the institution, ensuring it remains useful and adaptable over time.

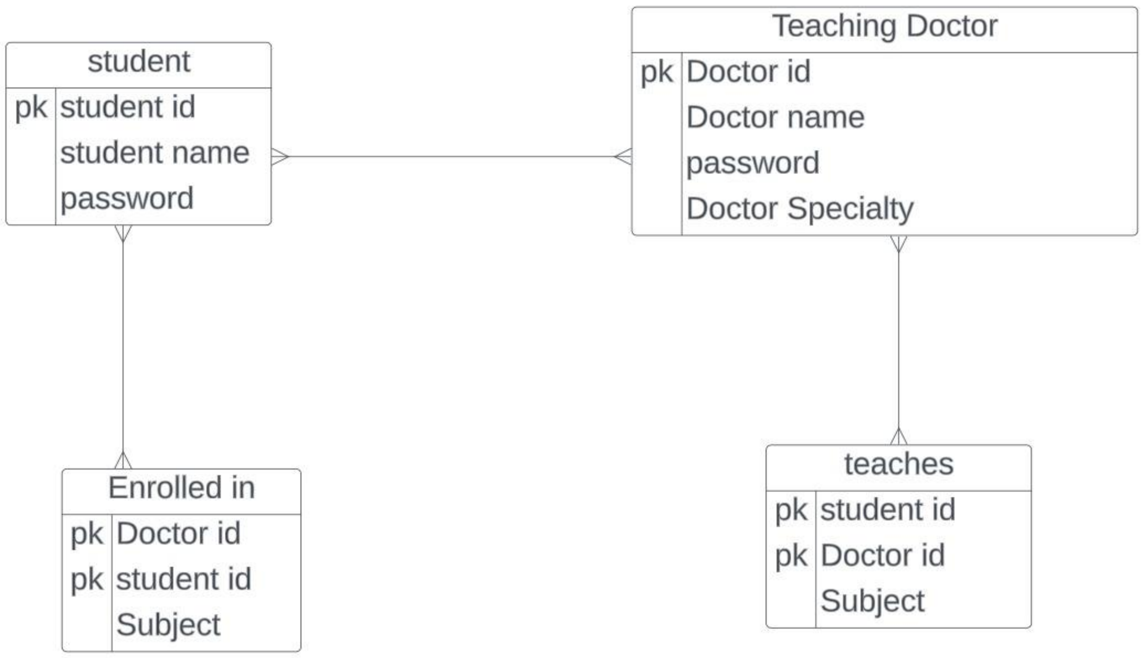
Here's the Database schema diagram:

Specifying focus on illustrating the database structure, including tables, relationships, and key fields.



ER diagram:

This is dedicated to illustrating the entities, attributes, and relationships in our database.


student
pk student id
student name
password
Teaching Doctor
pk Doctor id
Doctor name
password
Doctor Specialty
Enrolled in
pk Doctor id
pk student id
Subject
teaches
pk student id
pk Doctor id
Subject

2. Technical Architecture

A. System Architecture

1. Overview

This multi-layered system is designed for efficiency and scalability, encompassing frontend, backend, and database layers. An included diagram visually represents the system's architecture, highlighting the interaction between these layers.

2. Frontend Architecture

Built with HTML, CSS, and JavaScript, the frontend focuses on user interface and experience. HTML structures the content, CSS styles it, and JavaScript adds interactive elements, ensuring a dynamic and responsive design.

3. Backend Architecture

Utilizing PHP with the Laravel framework, the backend follows the MVC pattern. This setup allows for efficient handling of business logic, data processing, and communication with the frontend and database.

Laravel enhances functionality with its middleware, ORM, and templating capabilities, contributing to robust security and efficient data management.

4. Database Architecture

A MySQL database is employed, chosen for its robustness in handling complex data relationships. The schema is designed for optimal data storage, integrity, and efficient retrieval.

5. Mobile Application Architecture

The mobile app is developed in Flutter, adhering to the Model-View-ViewModel (MVVM) architecture. This pattern separates business logic

from UI elements, facilitating a more organized and maintainable codebase.

State management is handled using BLoC (Business Logic Component), specifically the Cubit library. This approach aids in managing the app's state reactively and efficiently, ensuring a smooth and responsive user experience.

Flutter's cross-platform capabilities guarantee a consistent and engaging experience across both Android and iOS.

The mobile app communicates with the backend through RESTful APIs, ensuring real-time data synchronization and a seamless integration with the web platform.

B. Technology Stack and used programming languages

1. Frontend Technologies

HTML, CSS, and JavaScript are the core technologies, providing structure, style, and functionality to the user interface.

2. Backend Technologies

PHP and Laravel form the backbone of the backend. Laravel's MVC architecture streamlines development and enhances functionality.

3. Database Technologies

MySQL is chosen for its reliability and support for complex data structures.

4. Mobile App Development

Flutter: Used for its cross-platform capabilities, allowing a uniform application on different mobile platforms.

MVVM Pattern: Adopted in Flutter to separate business logic (ViewModel) from UI components (View), with Model handling data.

BLoC (Cubit): Chosen for state management, facilitating efficient and understandable state changes within the app.

5. Additional Tools and Libraries

Various supporting tools and libraries are integrated to aid development, testing, and deployment, chosen for their effectiveness and compatibility with the overall system.

6. Rationale Behind Technology Choices

Each technology and architecture pattern is selected to optimize performance, development efficiency, and future scalability. This thoughtful integration ensures a robust and user-friendly system.

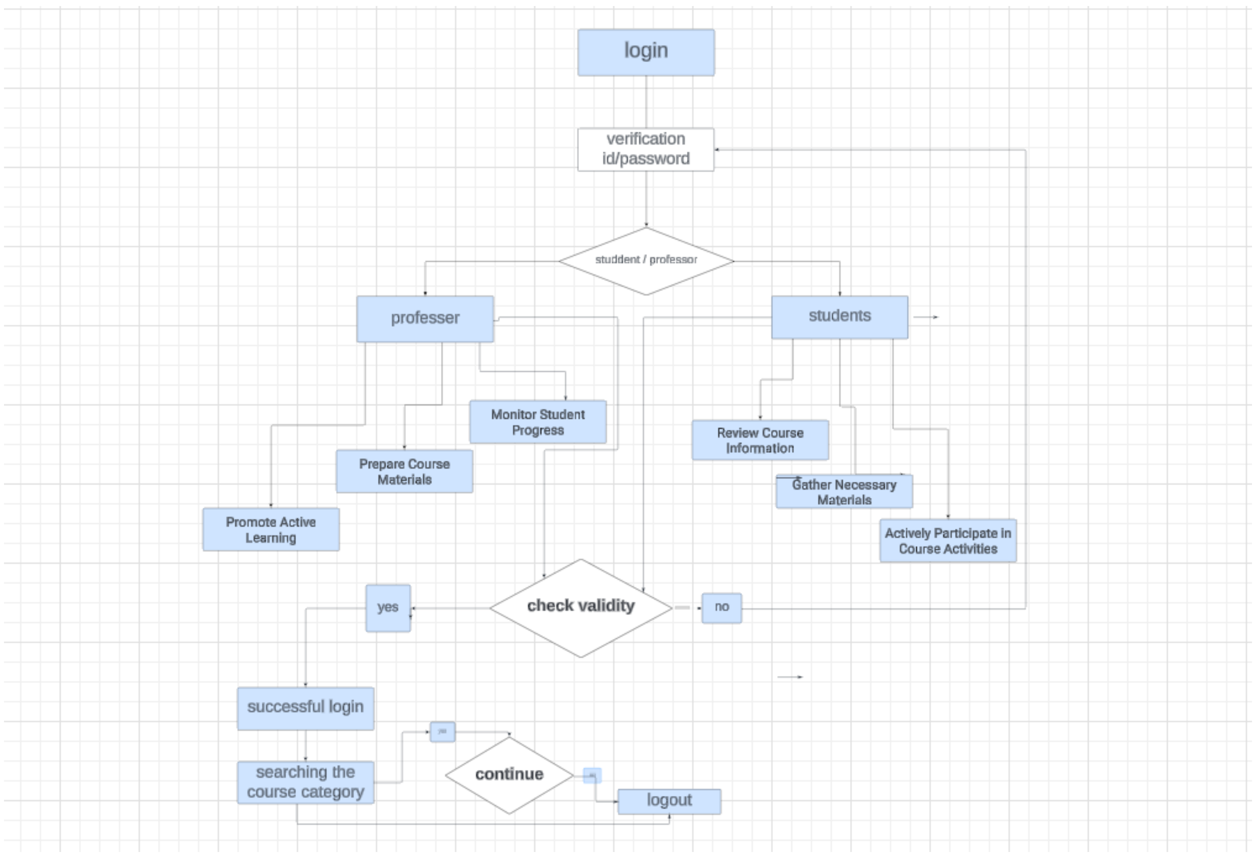
7. Future-Proofing

The system is designed with adaptability in mind, allowing for easy incorporation of future technological advancements and features.

3. System Design

- **System workflow:**

Here's the project's simple flowchart:

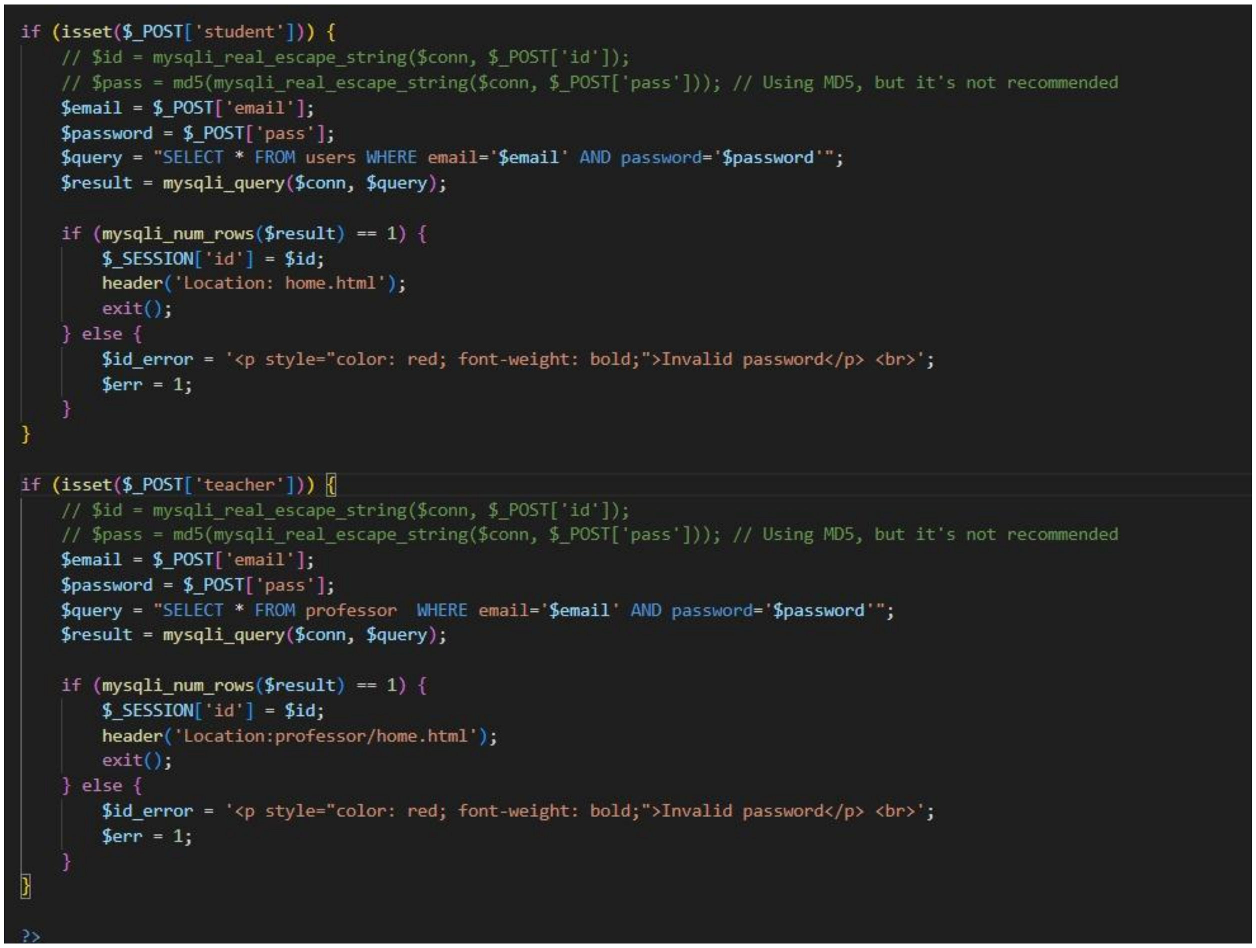


- **Web design**

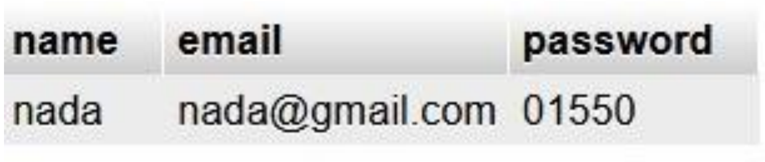
LOGIN PAGE

- A login page: is a screen in an application that prompts users to enter their credentials (such as username and password) in order to access the system or platform. The purpose of a login page is to authenticate users and verify

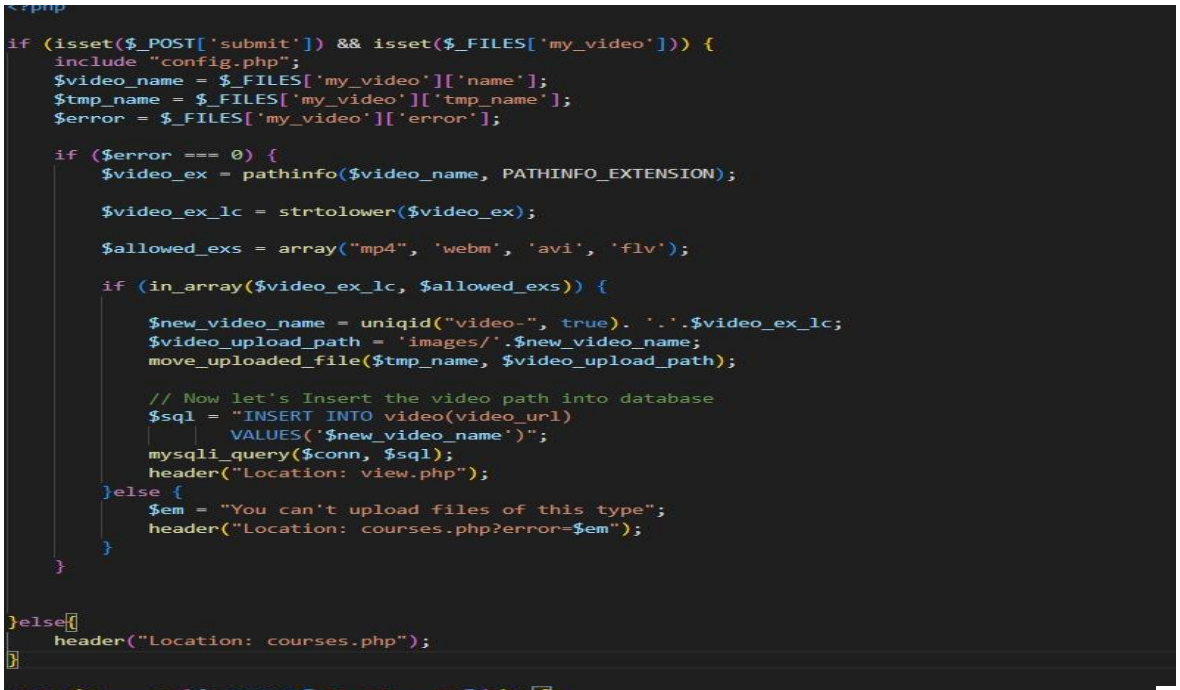
their identity before granting them access to the protected resources or features.



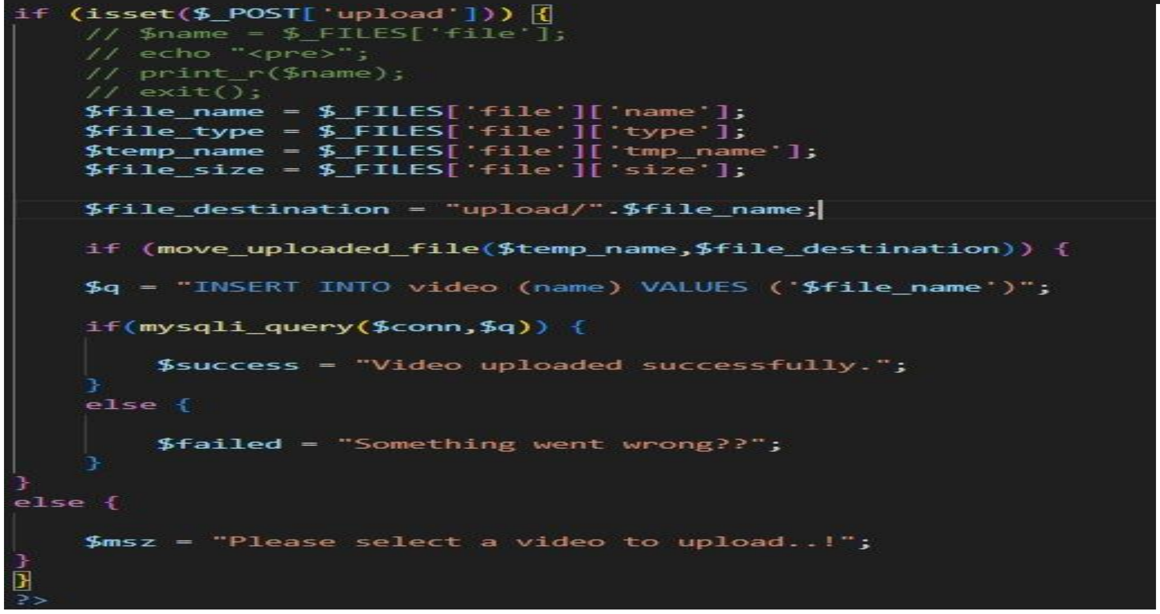
```
if (isset($_POST['student'])) {
    // $id = mysqli_real_escape_string($conn, $_POST['id']);
    // $pass = md5(mysqli_real_escape_string($conn, $_POST['pass'])); // Using MD5, but it's not recommended
    $email = $_POST['email'];
    $password = $_POST['pass'];
    $query = "SELECT * FROM users WHERE email='$email' AND password='$password'";
    $result = mysqli_query($conn, $query);

    if (mysqli_num_rows($result) == 1) {
        $_SESSION['id'] = $id;
        header('Location: home.html');
        exit();
    } else {
        $id_error = '<p style="color: red; font-weight: bold;">Invalid password</p> <br>';
        $err = 1;
    }
}

if (isset($_POST['teacher'])) {
    // $id = mysqli_real_escape_string($conn, $_POST['id']);
    // $pass = md5(mysqli_real_escape_string($conn, $_POST['pass'])); // Using MD5, but it's not recommended
    $email = $_POST['email'];
    $password = $_POST['pass'];
    $query = "SELECT * FROM professor  WHERE email='$email' AND password='$password'";
    $result = mysqli_query($conn, $query);

    if (mysqli_num_rows($result) == 1) {
        $_SESSION['id'] = $id;
        header('Location:professor/home.html');
        exit();
    } else {
        $id_error = '<p style="color: red; font-weight: bold;">Invalid password</p> <br>';
        $err = 1;
    }
}

?>
```



name	email	password
nada	nada@gmail.com	01550



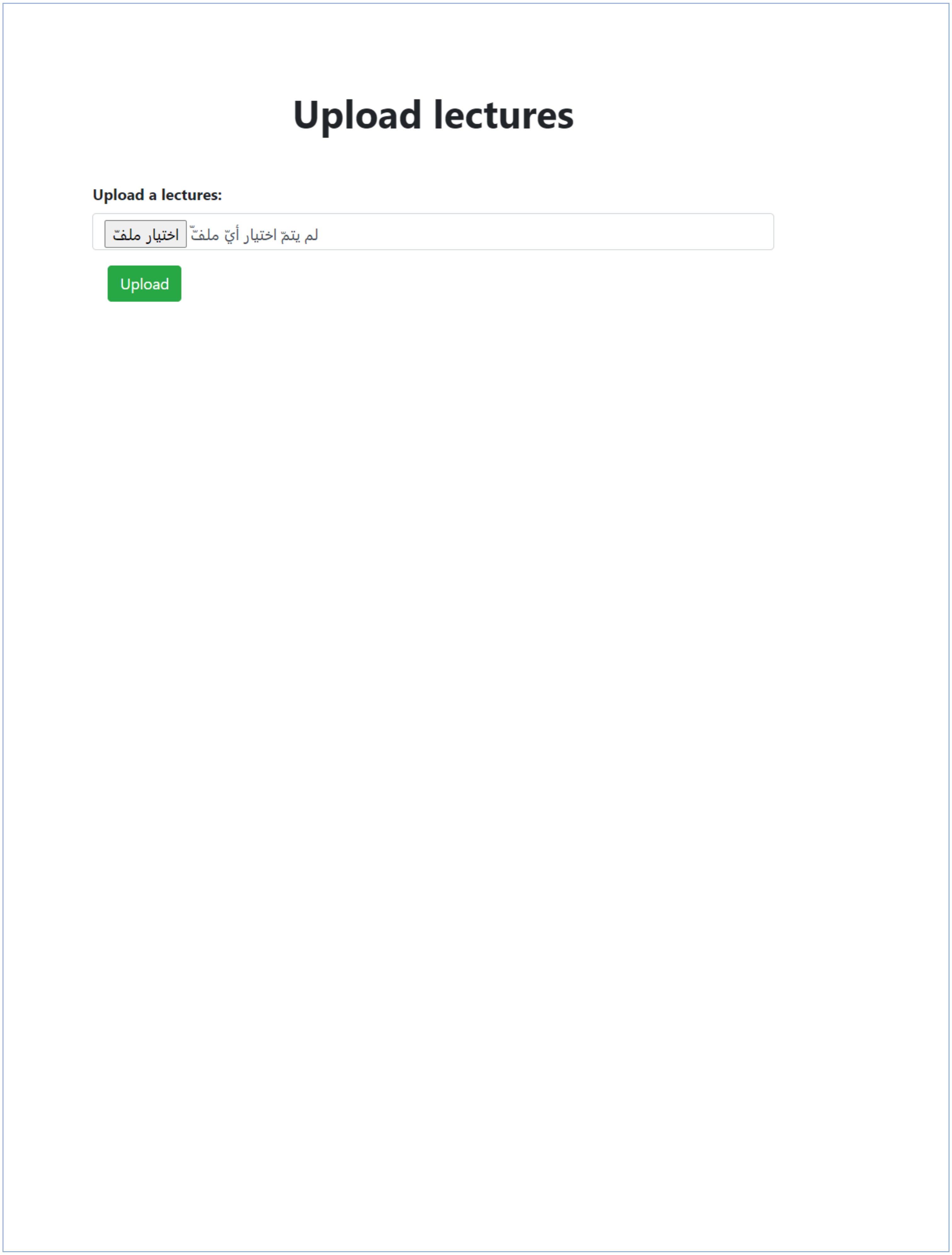
```
<?php

if (isset($_POST['submit']) && isset($_FILES['my_video'])) {
    include "config.php";
    $video_name = $_FILES['my_video']['name'];
    $tmp_name = $_FILES['my_video']['tmp_name'];
    $error = $_FILES['my_video']['error'];

    if ($error === 0) {
        $video_ex = pathinfo($video_name, PATHINFO_EXTENSION);

        $video_ex_lc = strtolower($video_ex);

        $allowed_exs = array("mp4", 'webm', 'avi', 'flv');

        if (in_array($video_ex_lc, $allowed_exs)) {

            $new_video_name = uniqid("video-", true). '.'.$video_ex_lc;
            $video_upload_path = 'images/'.$new_video_name;
            move_uploaded_file($tmp_name, $video_upload_path);

            // Now let's Insert the video path into database
            $sql = "INSERT INTO video(video_url)
                   VALUES('$new_video_name')";
            mysqli_query($conn, $sql);
            header("Location: view.php");
        }else {
            $em = "You can't upload files of this type";
            header("Location: courses.php?error=$em");
        }
    }


}else{
    header("Location: courses.php");
}
```



```
if (isset($_POST['upload'])) {
    // $name = $_FILES['file'];
    // echo "<pre>";
    // print_r($name);
    // exit();
    $file_name = $_FILES['file']['name'];
    $file_type = $_FILES['file']['type'];
    $temp_name = $_FILES['file']['tmp_name'];
    $file_size = $_FILES['file']['size'];

    $file_destination = "upload/".$file_name;

    if (move_uploaded_file($temp_name,$file_destination)) {

    $q = "INSERT INTO video (name) VALUES ('$file_name')";

    if(mysqli_query($conn,$q)) {

        $success = "Video uploaded successfully.";
    }
    else {

        $failed = "Something went wrong??";
    }
}
else {

    $msz = "Please select a video to upload..!";
}
}
?>
```

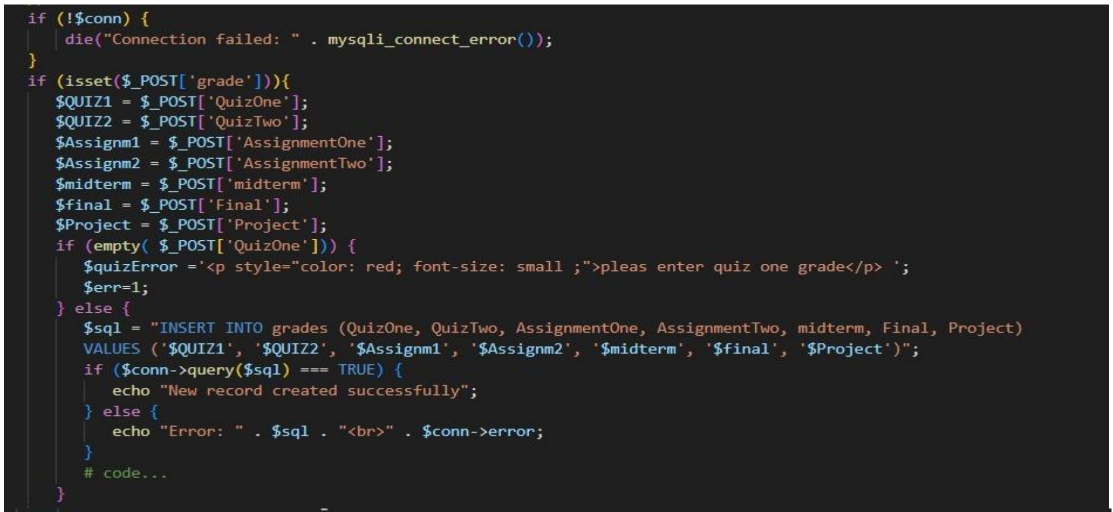
Edit Page

-An edit page: is a screen within an application that allows doctor to modify or update data, such as profile information, documents, or courses. The purpose of an edit page is to provide users with a user-friendly interface

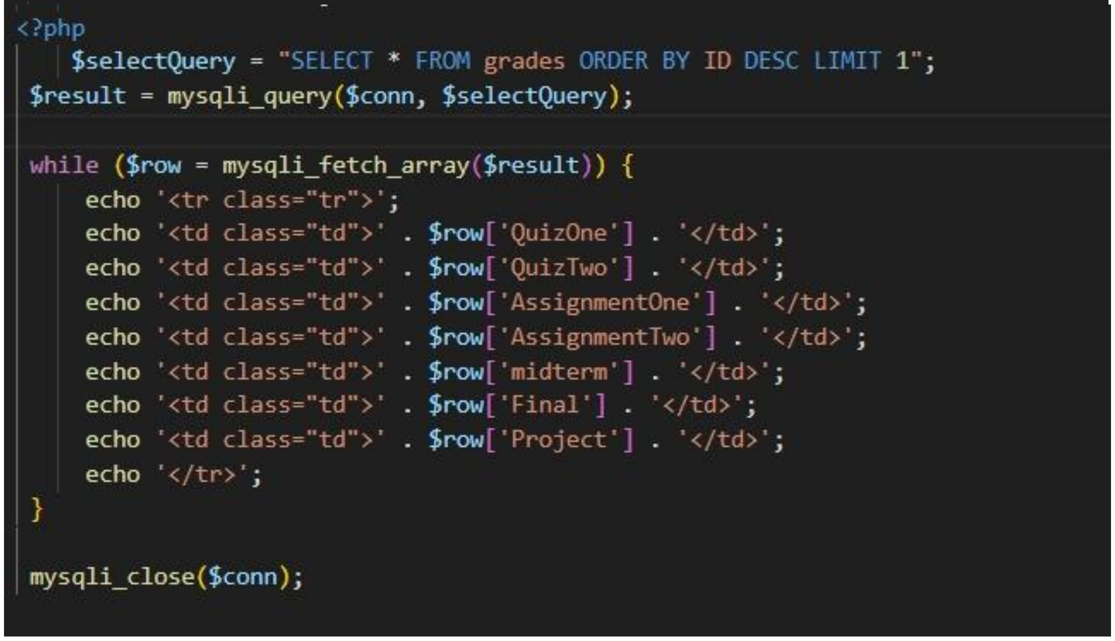

Upload lectures
Upload a lectures:
اختيار ملفّ
لم يتمّ اختيار أيّ ملفّ
Upload

Courses page

-In courses page: this screen allows the doctor to add grades for the student, then transfer them to Database, and then display the grade to the student

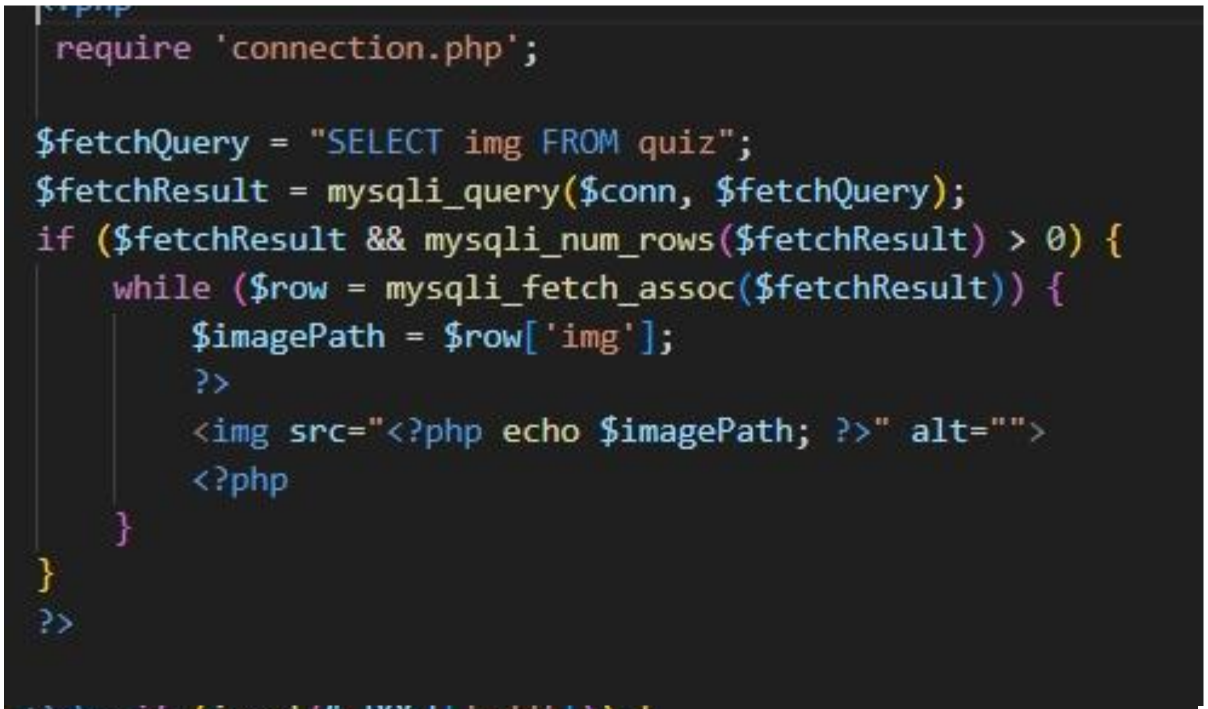


```php
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
if (isset($_POST['grade'])){
   $QUIZ1 = $_POST['QuizOne'];
   $QUIZ2 = $_POST['QuizTwo'];
   $Assignm1 = $_POST['AssignmentOne'];
   $Assignm2 = $_POST['AssignmentTwo'];
   $midterm = $_POST['midterm'];
   $final = $_POST['Final'];
   $Project = $_POST['Project'];
   if (empty( $_POST['QuizOne'])) {
      $quizError ='<p style="color: red; font-size: small ;">pleas enter quiz one grade</p> ';
      $err=1;
   } else {
      $sql = "INSERT INTO grades (QuizOne, QuizTwo, AssignmentOne, AssignmentTwo, midterm, Final, Project)
      VALUES ('$QUIZ1', '$QUIZ2', '$Assignm1', '$Assignm2', '$midterm', '$final', '$Project')";
      if ($conn->query($sql) === TRUE) {
         echo "New record created successfully";
      } else {
         echo "Error: " . $sql . "<br>" . $conn->error;
      }
      # code...
   }
```

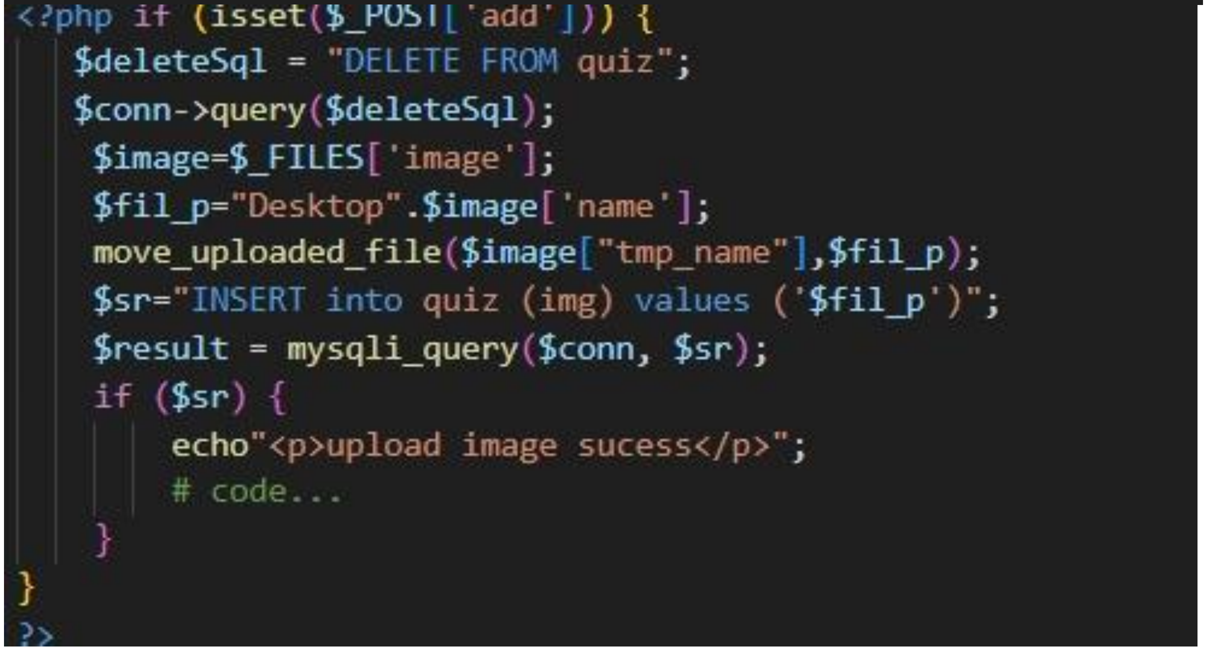


```php
<?php
    $selectQuery = "SELECT * FROM grades ORDER BY ID DESC LIMIT 1";
 $result = mysqli_query($conn, $selectQuery);

 while ($row = mysqli_fetch_array($result)) {
     echo '<tr class="tr">';
     echo '<td class="td">' . $row['QuizOne'] . '</td>';
     echo '<td class="td">' . $row['QuizTwo'] . '</td>';
     echo '<td class="td">' . $row['AssignmentOne'] . '</td>';
     echo '<td class="td">' . $row['AssignmentTwo'] . '</td>';
     echo '<td class="td">' . $row['midterm'] . '</td>';
     echo '<td class="td">' . $row['Final'] . '</td>';
     echo '<td class="td">' . $row['Project'] . '</td>';
     echo '</tr>';
 }

 mysqli_close($conn);
```

- in this part the student finishes the quiz and submit it, then the doctor sees the result of the quiz in database

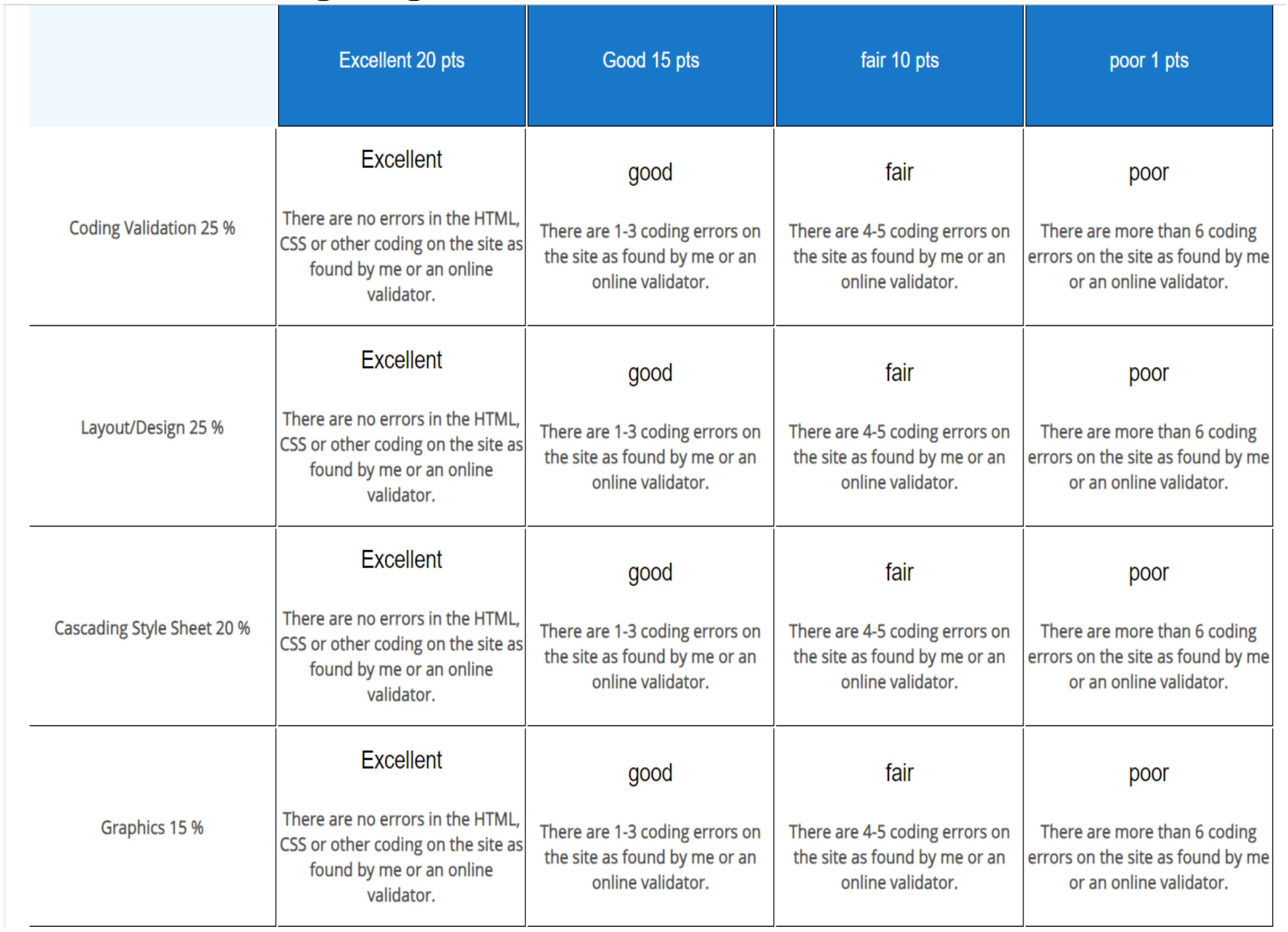


```php
 require 'connection.php';

$fetchQuery = "SELECT img FROM quiz";
$fetchResult = mysqli_query($conn, $fetchQuery);
if ($fetchResult && mysqli_num_rows($fetchResult) > 0) {
    while ($row = mysqli_fetch_assoc($fetchResult)) {
        $imagePath = $row['img'];
        ?>
        <img src="<?php echo $imagePath; ?>" alt="">
        <?php
    }
}
?>
```

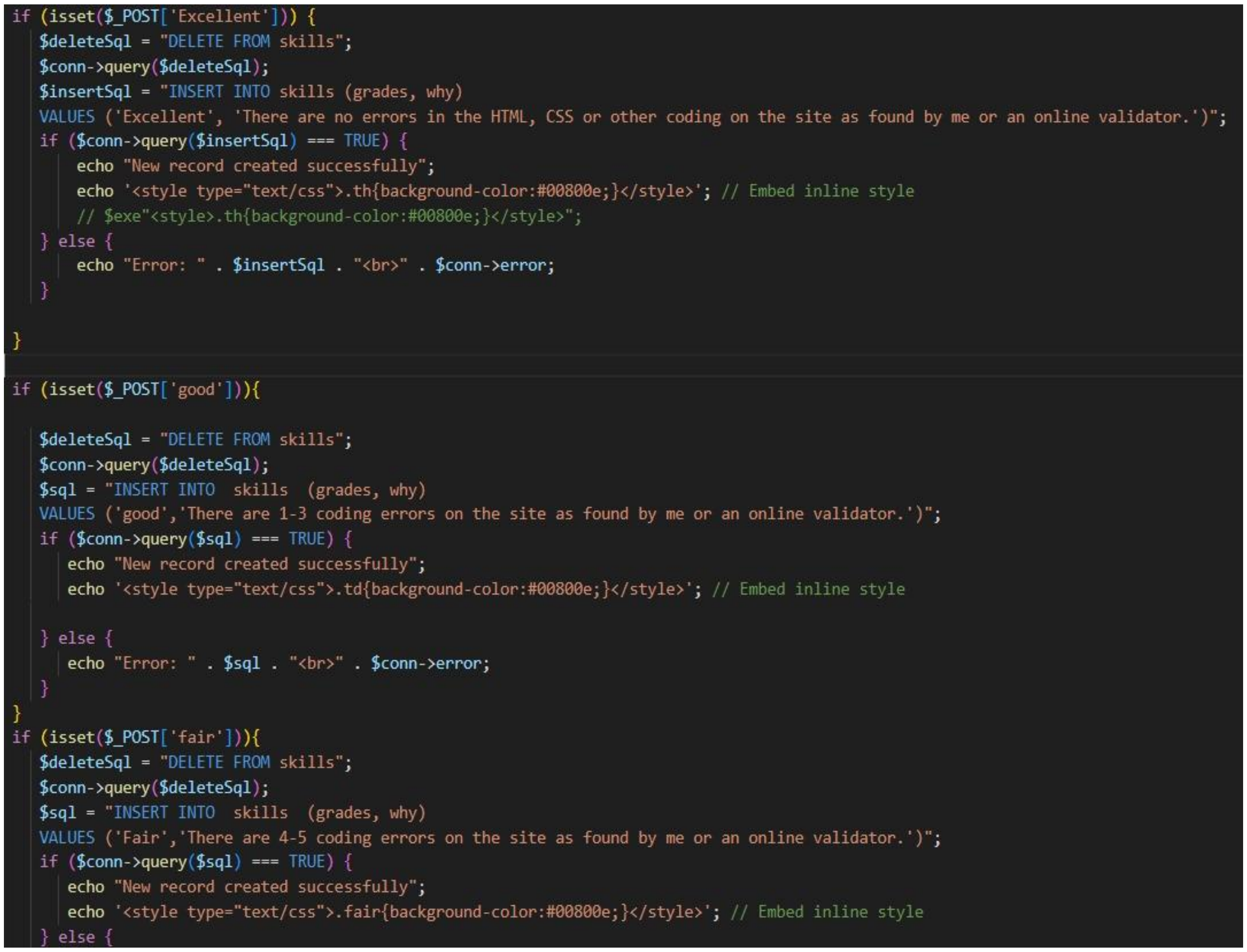


```php
<?php if (isset($_POST['add'])) {
   $deleteSql = "DELETE FROM quiz";
   $conn->query($deleteSql);
    $image=$_FILES['image'];
    $fil_p="Desktop".$image['name'];
    move_uploaded_file($image["tmp_name"],$fil_p);
    $sr="INSERT into quiz (img) values ('$fil_p')";
    $result = mysqli_query($conn, $sr);
    if ($sr) {
        echo"<p>upload image sucess</p>";
        # code...
    }
}
?>
```

-Here we make it possible that the doctor evaluates the student's quiz based on the student's solution and divides it into four needs. The doctor evaluates it: Coding Validation 25% , Layout/Design 25% , Cascading Style Sheet 20% , Graphics 15% , Each of these items is divided into Excellent 20 pts, Good 15 pts, fair 10 pts, poor 1 pts, The doctor let the student knows the reason for having this grade.

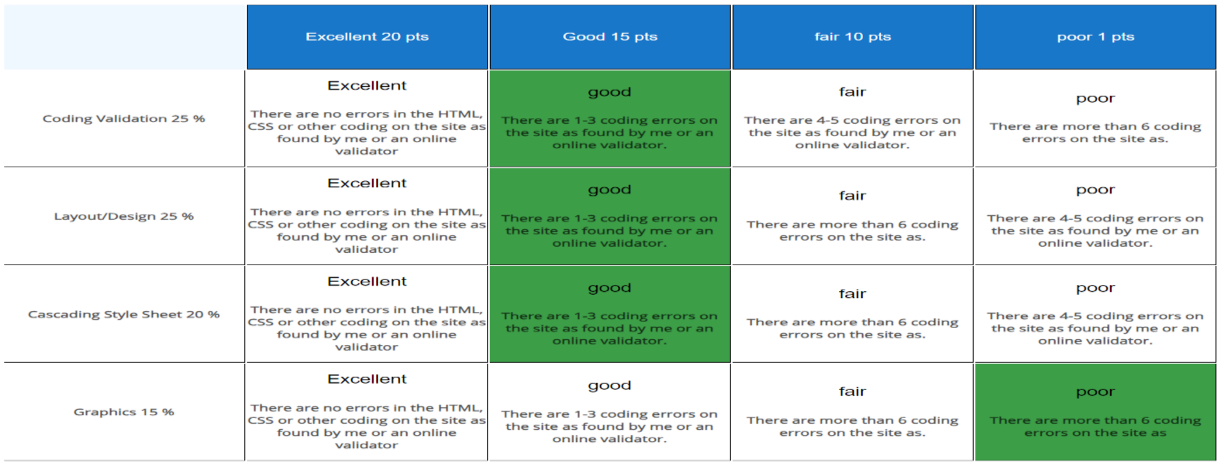


	Excellent 20 pts	Good 15 pts	fair 10 pts	poor 1 pts
Coding Validation 25 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator.	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are 4-5 coding errors on the site as found by me or an online validator.	poor There are more than 6 coding errors on the site as found by me or an online validator.
Layout/Design 25 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator.	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are 4-5 coding errors on the site as found by me or an online validator.	poor There are more than 6 coding errors on the site as found by me or an online validator.
Cascading Style Sheet 20 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator.	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are 4-5 coding errors on the site as found by me or an online validator.	poor There are more than 6 coding errors on the site as found by me or an online validator.
Graphics 15 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator.	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are 4-5 coding errors on the site as found by me or an online validator.	poor There are more than 6 coding errors on the site as found by me or an online validator.

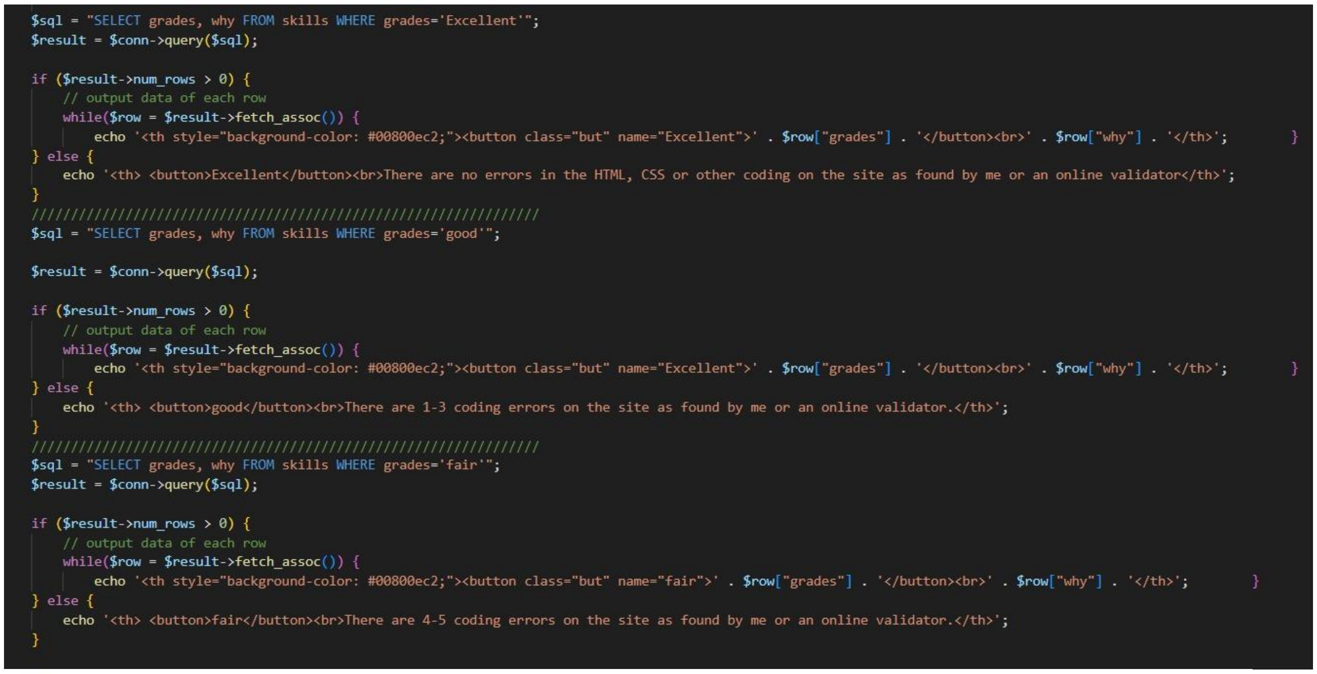


```
if (isset($_POST['Excellent'])) {
    $deleteSql = "DELETE FROM skills";
    $conn->query($deleteSql);
    $insertSql = "INSERT INTO skills (grades, why)
    VALUES ('Excellent', 'There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator.')";
    if ($conn->query($insertSql) === TRUE) {
        echo "New record created successfully";
        echo '<style type="text/css">.th{background-color:#00800e;}</style>'; // Embed inline style
        // $exe"<style>.th{background-color:#00800e;}</style>";
    } else {
        echo "Error: " . $insertSql . "<br>" . $conn->error;
    }

}

if (isset($_POST['good'])){

    $deleteSql = "DELETE FROM skills";
    $conn->query($deleteSql);
    $sql = "INSERT INTO  skills  (grades, why)
    VALUES ('good','There are 1-3 coding errors on the site as found by me or an online validator.')";
    if ($conn->query($sql) === TRUE) {
       echo "New record created successfully";
       echo '<style type="text/css">.td{background-color:#00800e;}</style>'; // Embed inline style

    } else {
       echo "Error: " . $sql . "<br>" . $conn->error;
    }
}
if (isset($_POST['fair'])){
    $deleteSql = "DELETE FROM skills";
    $conn->query($deleteSql);
    $sql = "INSERT INTO  skills  (grades, why)
    VALUES ('Fair','There are 4-5 coding errors on the site as found by me or an online validator.')";
    if ($conn->query($sql) === TRUE) {
       echo "New record created successfully";
       echo '<style type="text/css">.fair{background-color:#00800e;}</style>'; // Embed inline style
    } else {
```

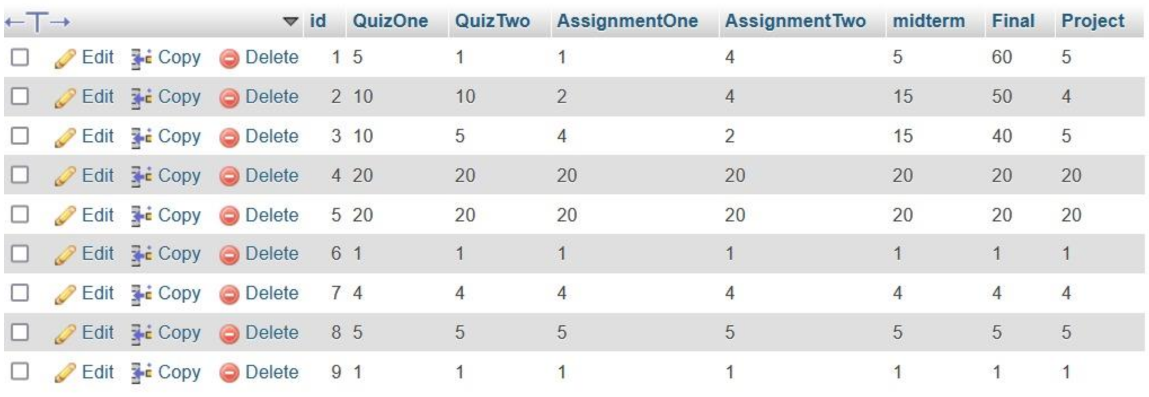
-in this page the student see his grade and faults, exactly where is his mistakes, and the database of courses



	Excellent 20 pts	Good 15 pts	fair 10 pts	poor 1 pts
Coding Validation 25 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are 4-5 coding errors on the site as found by me or an online validator.	poor There are more than 6 coding errors on the site as.
Layout/Design 25 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are more than 6 coding errors on the site as.	poor There are 4-5 coding errors on the site as found by me or an online validator.
Cascading Style Sheet 20 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are more than 6 coding errors on the site as.	poor There are 4-5 coding errors on the site as found by me or an online validator.
Graphics 15 %	Excellent There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator	good There are 1-3 coding errors on the site as found by me or an online validator.	fair There are more than 6 coding errors on the site as.	poor There are more than 6 coding errors on the site as



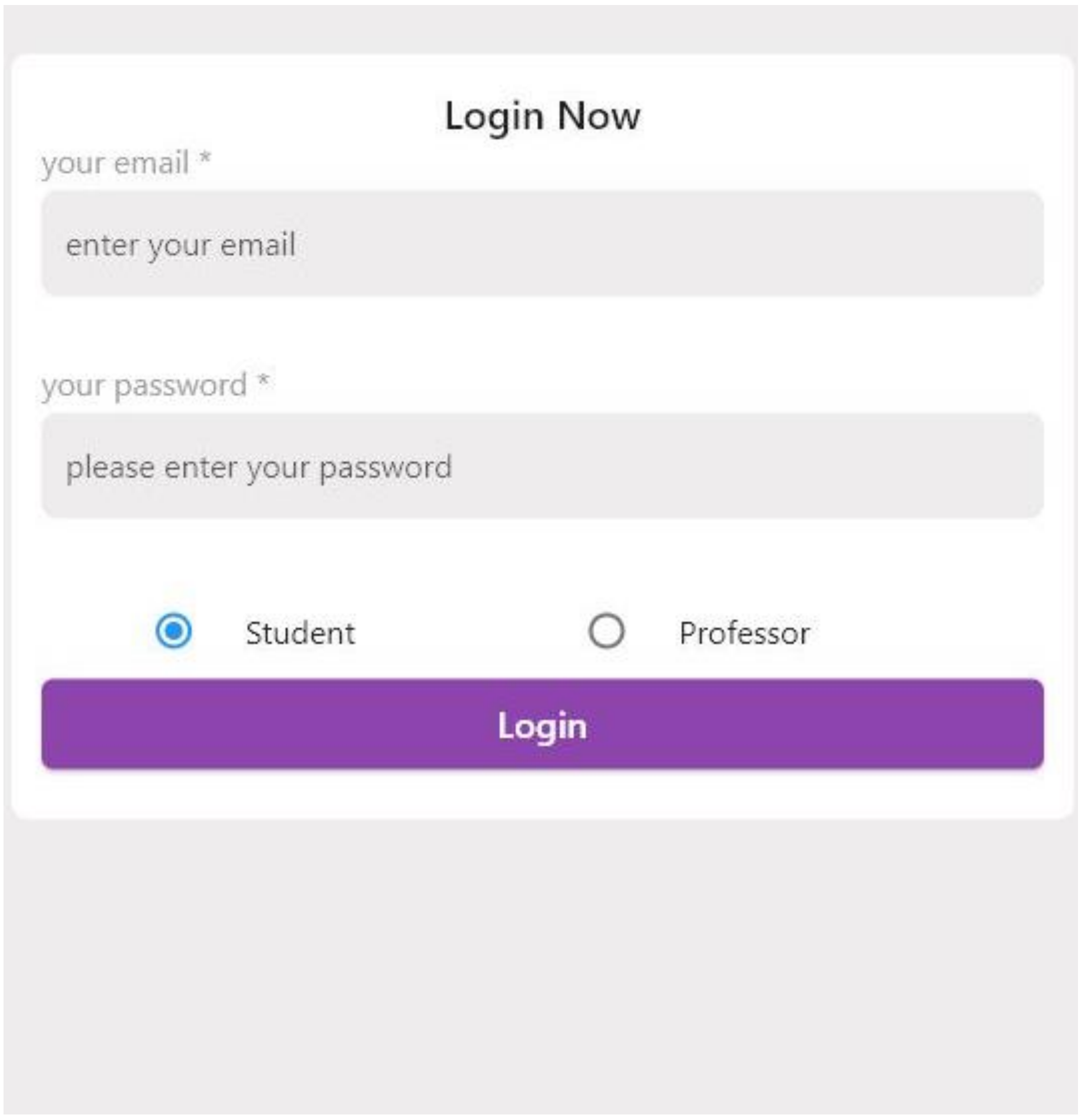
```
$sql = "SELECT grades, why FROM skills WHERE grades='Excellent'";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo '<th style="background-color: #00800ec2;"><button class="but" name="Excellent">' . $row["grades"] . '</button><br>' . $row["why"] . '</th>';        }
} else {
    echo '<th> <button>Excellent</button><br>There are no errors in the HTML, CSS or other coding on the site as found by me or an online validator</th>';
}
/////////////////////////////////////////////////////////////////
$sql = "SELECT grades, why FROM skills WHERE grades='good'";

$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo '<th style="background-color: #00800ec2;"><button class="but" name="Excellent">' . $row["grades"] . '</button><br>' . $row["why"] . '</th>';        }
} else {
    echo '<th> <button>good</button><br>There are 1-3 coding errors on the site as found by me or an online validator.</th>';
}
/////////////////////////////////////////////////////////////////
$sql = "SELECT grades, why FROM skills WHERE grades='fair'";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo '<th style="background-color: #00800ec2;"><button class="but" name="fair">' . $row["grades"] . '</button><br>' . $row["why"] . '</th>';        }
} else {
    echo '<th> <button>fair</button><br>There are 4-5 coding errors on the site as found by me or an online validator.</th>';
}
```



	id	QuizOne	QuizTwo	AssignmentOne	AssignmentTwo	midterm	Final	Project
Edit Copy Delete	1	5	1	1	4	5	60	5
Edit Copy Delete	2	10	10	2	4	15	50	4
Edit Copy Delete	3	10	5	4	2	15	40	5
Edit Copy Delete	4	20	20	20	20	20	20	20
Edit Copy Delete	5	20	20	20	20	20	20	20
Edit Copy Delete	6	1	1	1	1	1	1	1
Edit Copy Delete	7	4	4	4	4	4	4	4
Edit Copy Delete	8	5	5	5	5	5	5	5
Edit Copy Delete	9	1	1	1	1	1	1	1

- **Mobile Application**

Login Screen


Login Now

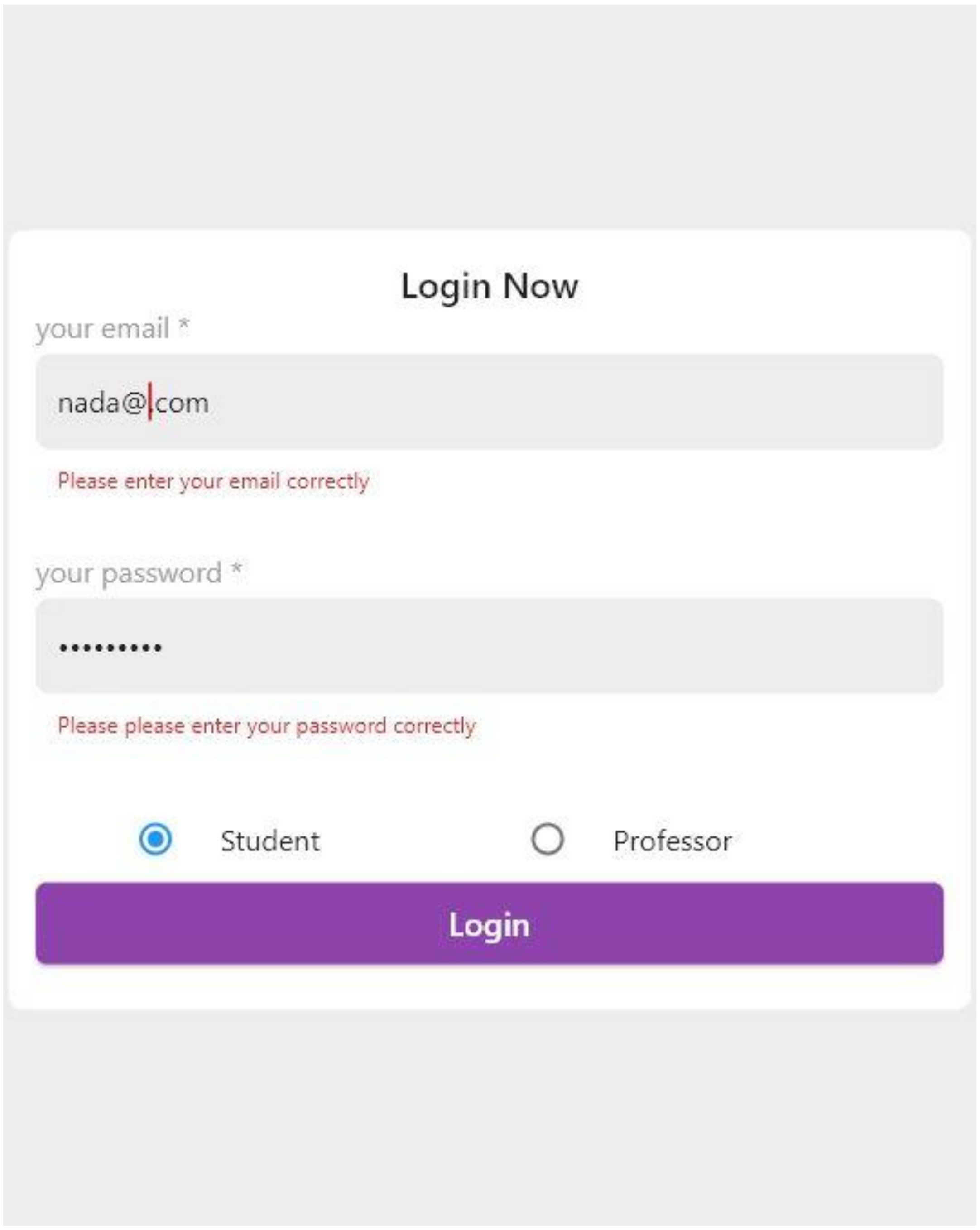
your email *

enter your email

your password *

please enter your password

[✓ Student] Professor



Login Now

your email *

nada@com

Please enter your email correctly

your password *

•••••••••

Please please enter your password correctly

(●) Student ( ) Professor

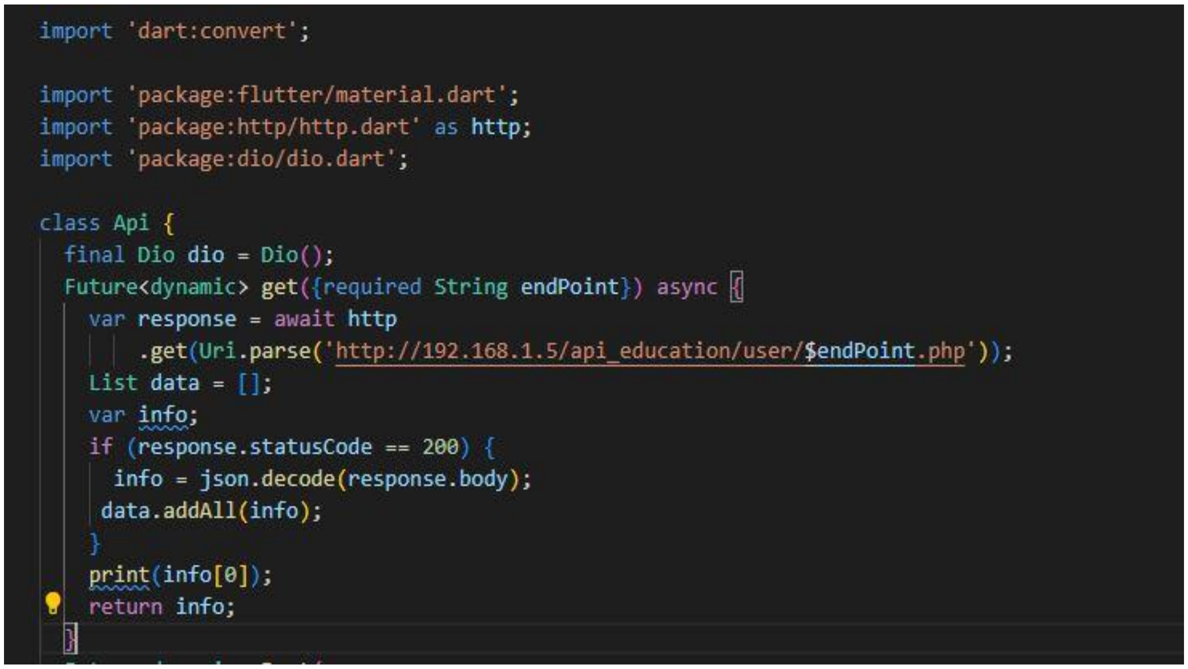
***In case email or password are written wrong**

- **The Login Page serves as the gateway to the Education Management System, providing a secure and personalized entry point for users. Users are prompted to enter their unique email and password to access the system. Additionally, a radio button selection allows users**

to specify their role, either as a student or a professor, ensuring a tailored experience based on their account type.

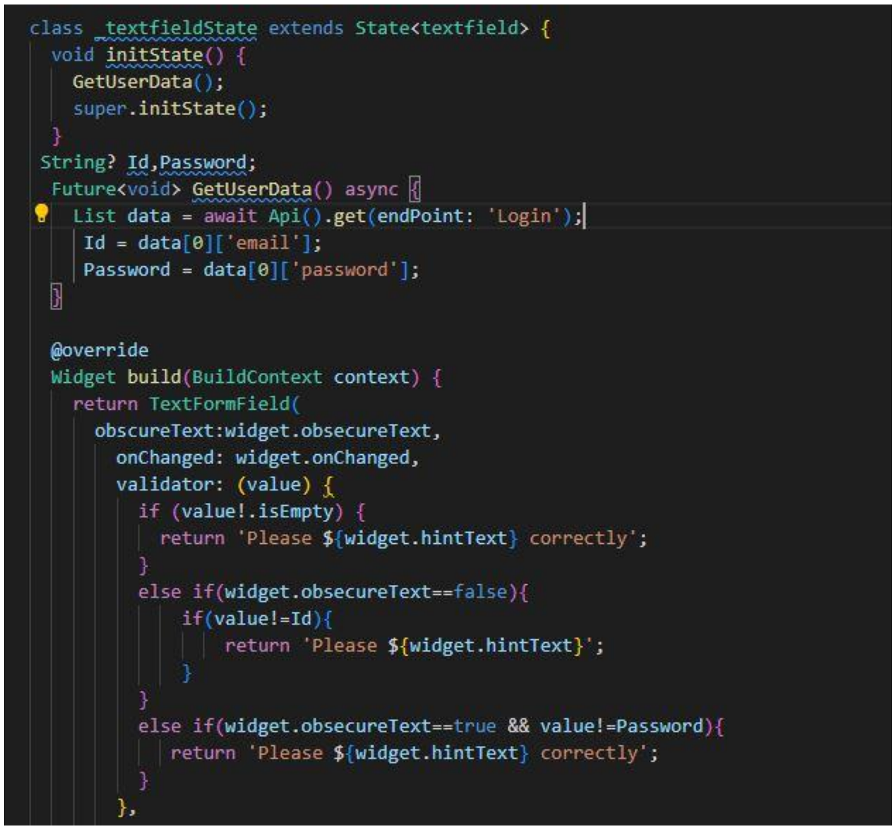
Key Features:

- **User Authentication: Securely verify user identity through a combination of unique email and password.**
- **Role Selection: Choose between 'Student' or 'Professor' via radio button to customize the system experience.**
- **Accessibility: User-friendly design for a seamless and intuitive login process.**



```dart
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'package:dio/dio.dart';

class Api {
  final Dio dio = Dio();
  Future<dynamic> get({required String endPoint}) async {
    var response = await http
        .get(Uri.parse('http://192.168.1.5/api_education/user/$endPoint.php'));
    List data = [];
    var info;
    if (response.statusCode == 200) {
      info = json.decode(response.body);
     data.addAll(info);
    }
    print(info[0]);
    return info;
  }
```

apiGetMethod(to Get the data from the database)



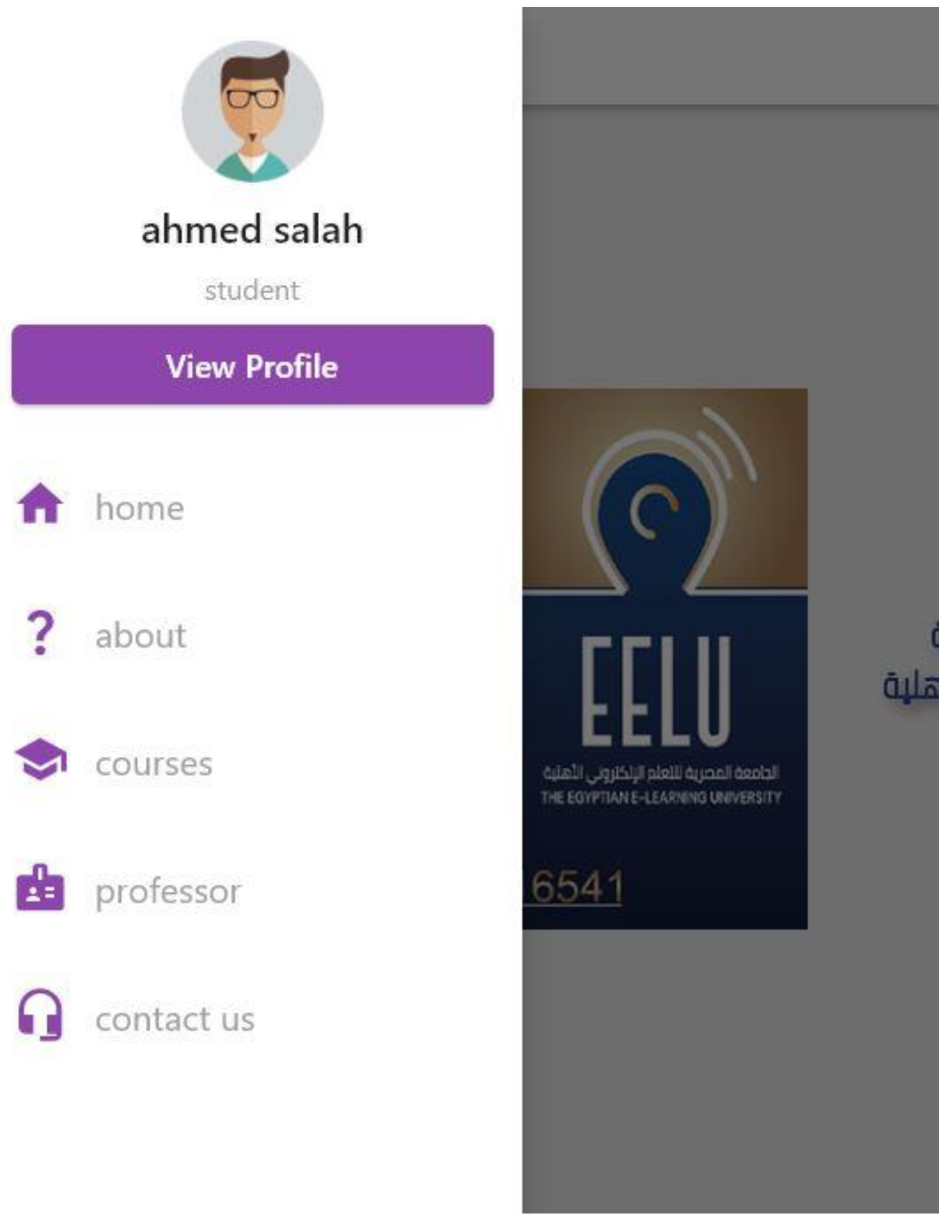
```
class _textfieldState extends State<textfield> {
  void initState() {
    GetUserData();
    super.initState();
  }
 String? Id,Password;
  Future<void> GetUserData() async {
    List data = await Api().get(endPoint: 'Login');
     Id = data[0]['email'];
     Password = data[0]['password'];
  }

  @override
  Widget build(BuildContext context) {
    return TextFormField(
      obscureText:widget.obsecureText,
        onChanged: widget.onChanged,
        validator: (value) {
          if (value!.isEmpty) {
            return 'Please ${widget.hintText} correctly';
          }
          else if(widget.obsecureText==false){
              if(value!=Id){
                  return 'Please ${widget.hintText}';
              }
          }
          else if(widget.obsecureText==true && value!=Password){
             return 'Please ${widget.hintText} correctly';
          }
        },
```

Login Page Code

The IdentifierRadioButton widget:

is a Flutter component designed to display a set of radio buttons for users to choose their role—either 'Student' or 'Professor'—within an education management system. This widget is implemented as a stateful widget, allowing dynamic updates to the user interface based on the selected role.

Home Screen : Student UI

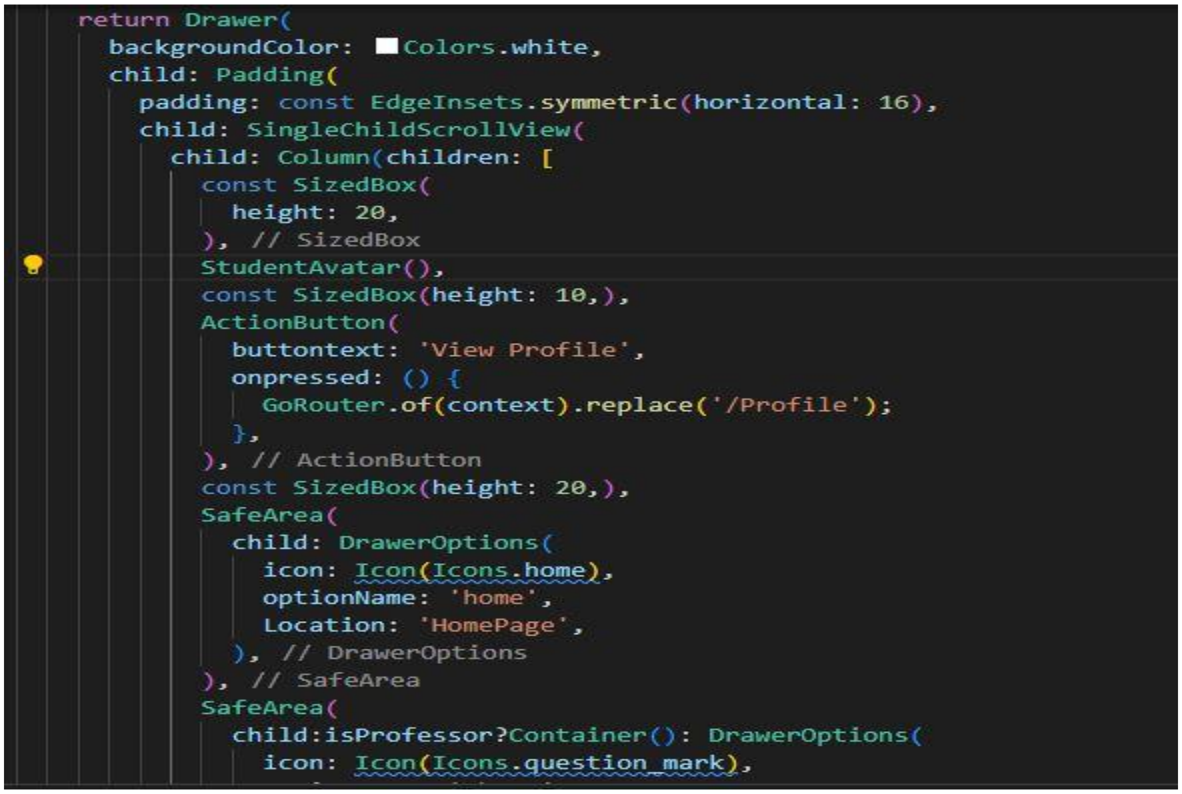

ahmed salah
student
View Profile
home
about
courses
professor
contact us
EELU
الجامعة المصرية للتعلم الإلكتروني الأهلية
THE EGYPTIAN E-LEARNING UNIVERSITY
6541

- **The Home Page serves as the central hub of the Education Management System, offering a user-friendly interface for students. The key feature of this page is the Drawer, providing quick access to various sections of the system. Users can navigate seamlessly to different pages, including 'Home', 'About', 'Courses', 'Professors', and 'Contact Us', by selecting the corresponding option from the Drawer menu.**

Key Features:

- **Drawer Navigation: A responsive and intuitive Drawer component allowing students to access different sections of the system effortlessly.**
- **Page Options: The Drawer menu includes options such as 'Home', providing an overview; 'About', offering system details; 'Courses', displaying available courses; 'Professors', showcasing teaching staff; and 'Contact Us', facilitating communication.**
- **Efficient Navigation: Students can click on any desired option in the Drawer to swiftly navigate to the chosen page.**

Drawer Code:

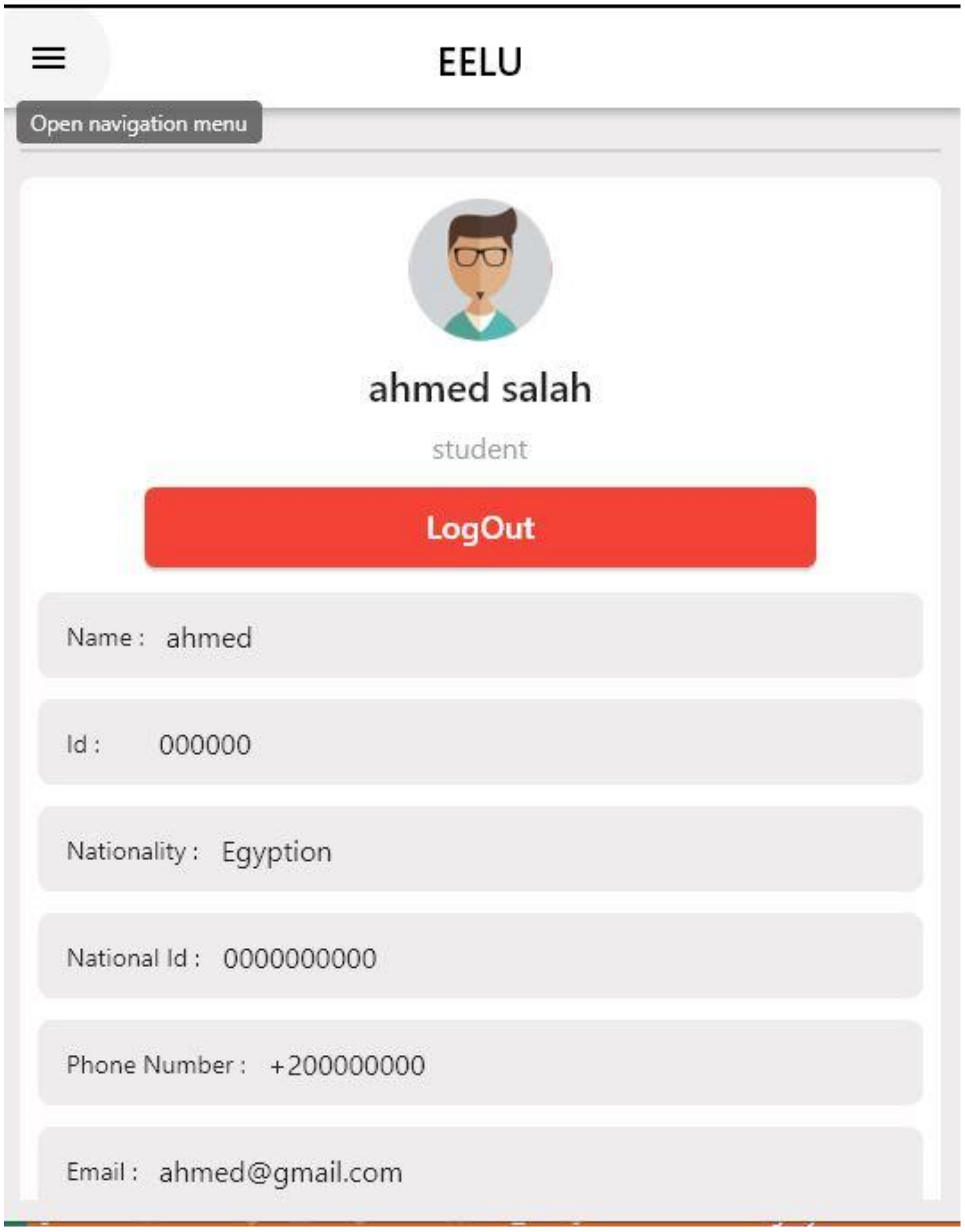


```
return Drawer(
  backgroundColor: Colors.white,
  child: Padding(
    padding: const EdgeInsets.symmetric(horizontal: 16),
    child: SingleChildScrollView(
      child: Column(children: [
        const SizedBox(
          height: 20,
        ), // SizedBox
        StudentAvatar(),
        const SizedBox(height: 10,),
        ActionButton(
          buttontext: 'View Profile',
          onpressed: () {
            GoRouter.of(context).replace('/Profile');
          },
        ), // ActionButton
        const SizedBox(height: 20,),
        SafeArea(
          child: DrawerOptions(
            icon: Icon(Icons.home),
            optionName: 'home',
            Location: 'HomePage',
          ), // DrawerOptions
        ), // SafeArea
        SafeArea(
          child:isProfessor?Container(): DrawerOptions(
            icon: Icon(Icons.question_mark),
```

- **The NavigationDrawerWidget is a Flutter component that encapsulates the navigation drawer functionality within the Education Management System. This drawer provides users with quick access to essential features, such as viewing the user's profile, navigating to the home page, exploring courses, accessing professor-specific information, and contacting the system administrators.**

Key Features:

- **Modular Components: Utilizes custom components like StudentAvatar, ActionButton, and DrawerOptions to create a modular and extensible design.**
- **Dynamic Navigation: Depending on the user's role (student or professor), certain options are conditionally displayed in the drawer, enhancing the user experience.**
- **Routing Integration: Leverages the GoRouter package for efficient navigation between different pages within the Education Management System.**
- **Stylish Design: Incorporates consistent padding, spacing, and visual elements for an aesthetically pleasing and user-friendly interface.**


EELU

ahmed salah

student

Name : ahmed

Id : 000000

Nationality : Egyption

National Id : 0000000000

Phone Number : +200000000

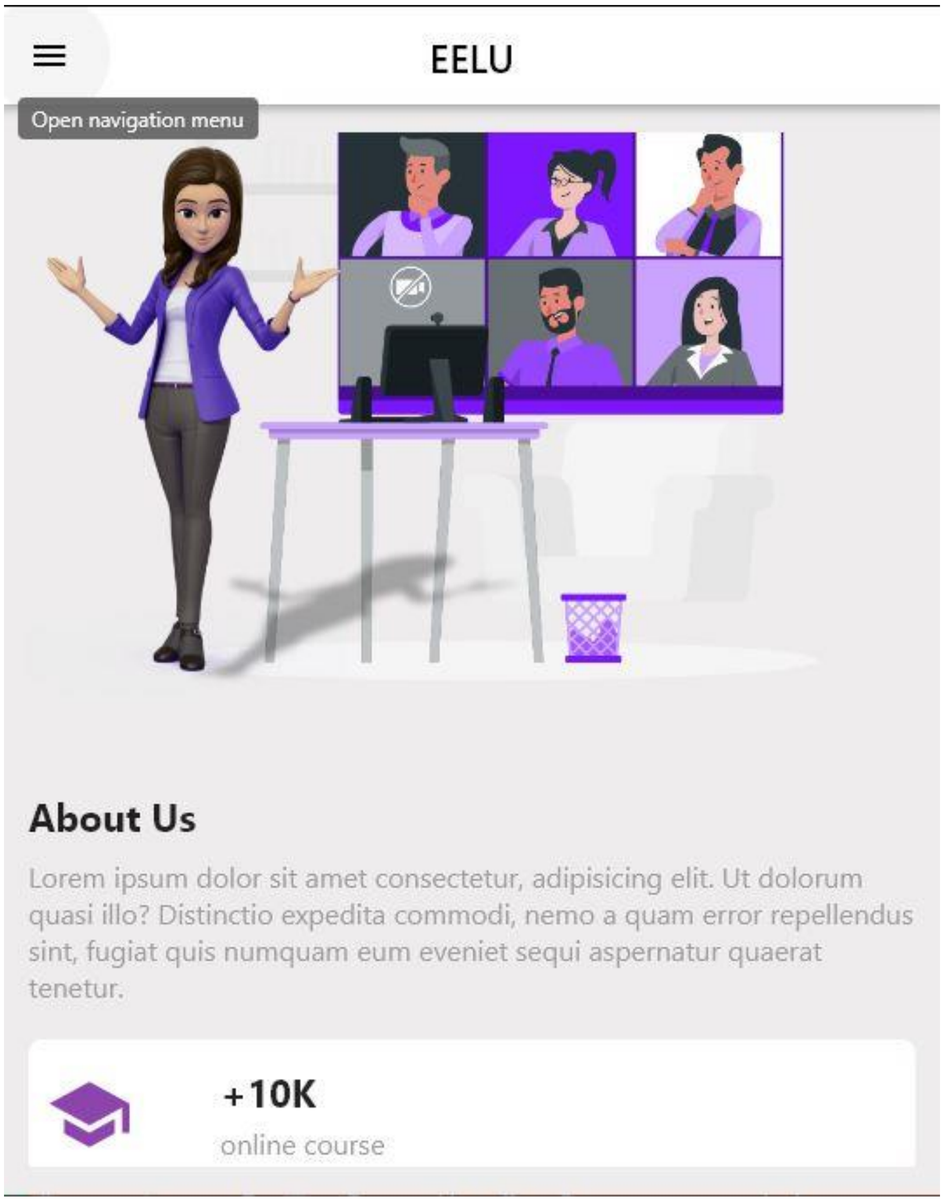
Email : ahmed@gmail.com

Student Profile

- **The Profile Page within the Education Management System offers users a comprehensive overview of their personal information. Students or professors can access key details such as their ID, email, national ID, and phone number on this page. The page is thoughtfully designed to present information in a clear and organized manner, enhancing user awareness and engagement. Additionally, users have the convenience of logging out directly from this page.**

Key Features:

- **Personal Information Display: Provides a user-friendly display of essential details, including ID, email, national ID, and phone number.**
- **Logout Functionality: Enables users to log out seamlessly, promoting secure and convenient access control.**
- **Clean and Intuitive Design: Organizes information in a visually appealing and easily digestible format for an enhanced user experience.**
- **Accessibility: Ensures users can quickly access and review their personal details.**


EELU

Open navigation menu

About Us

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ut dolorum quasi illo? Distinctio expedita commodi, nemo a quam error repellendus sint, fugiat quis numquam eum eveniet sequi aspernatur quaerat tenetur.

+10K

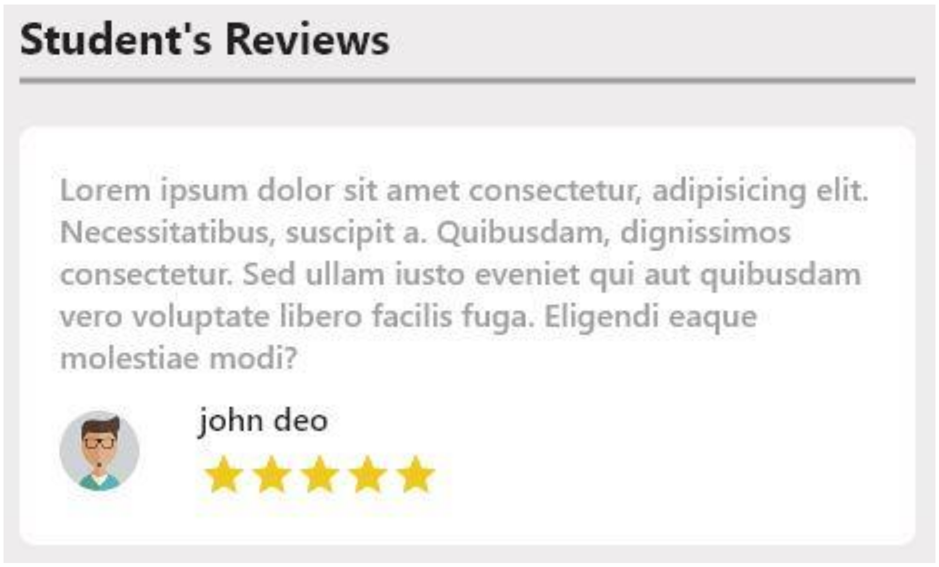
online course

About Page

- **The About Page provides users within the Education Management System with valuable information about the university. Users can access essential details, such as the university's mission, vision, and core values. Additionally, this page may include information about the institution's history, notable achievements, and key personnel. The About Page is designed to offer users insight into the university's identity and values, fostering a sense of connection and understanding.**

- ***Key Features:***
- **Mission and Vision: Presents the university's mission and vision statements, outlining its overarching goals and values.**
- **Institutional History: Offers a brief overview of the university's history, highlighting key milestones and achievements.**

Student Review:



Student's Reviews

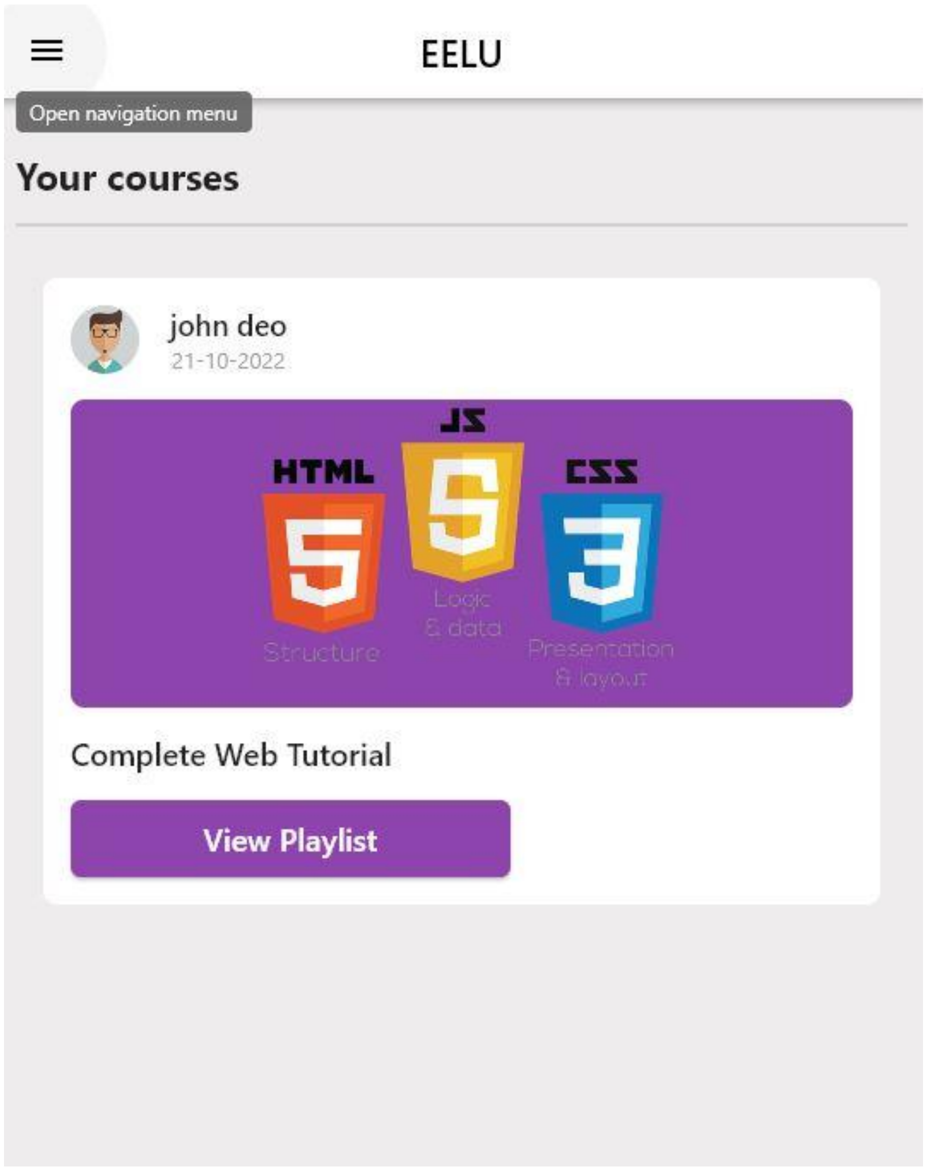
Lorem ipsum dolor sit amet consectetur, adipisicing elit. Necessitatibus, suscipit a. Quibusdam, dignissimos consectetur. Sed ullam iusto eveniet qui aut quibusdam vero voluptate libero facilis fuga. Eligendi eaque molestiae modi?

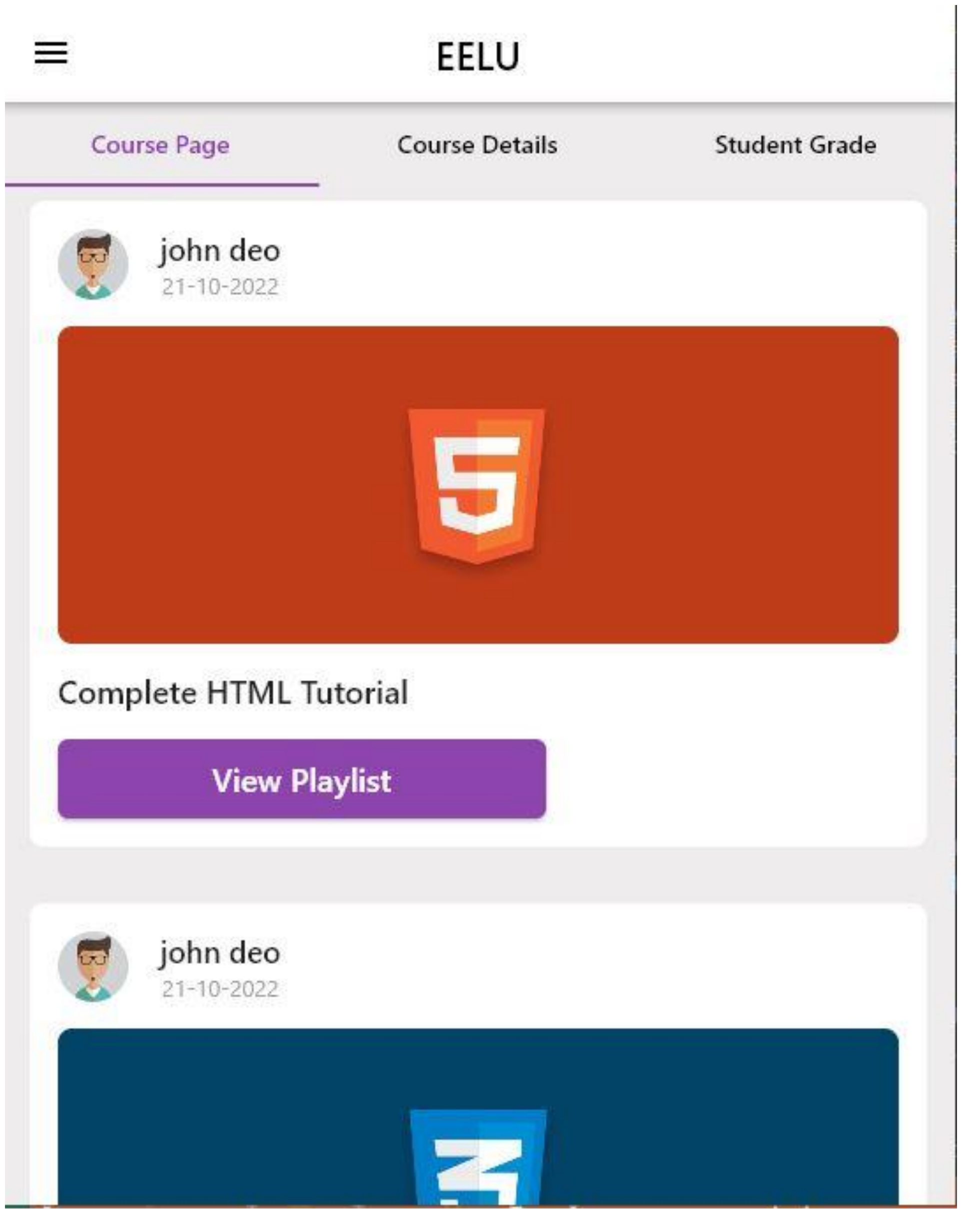
john deo

★★★★★

- ***Key Features:***
- **Review Submission: Enables students to write and submit reviews, sharing their thoughts and overall experiences.**
- **Read Reviews: Allows students to access and read reviews from other students.**
- **Rating System: Optionally includes a rating system, allowing students to assign scores to different aspects of their experience.**
- **Commenting: Facilitates interaction through commenting, allowing students to engage in discussions or seek additional information.**

Course Page:

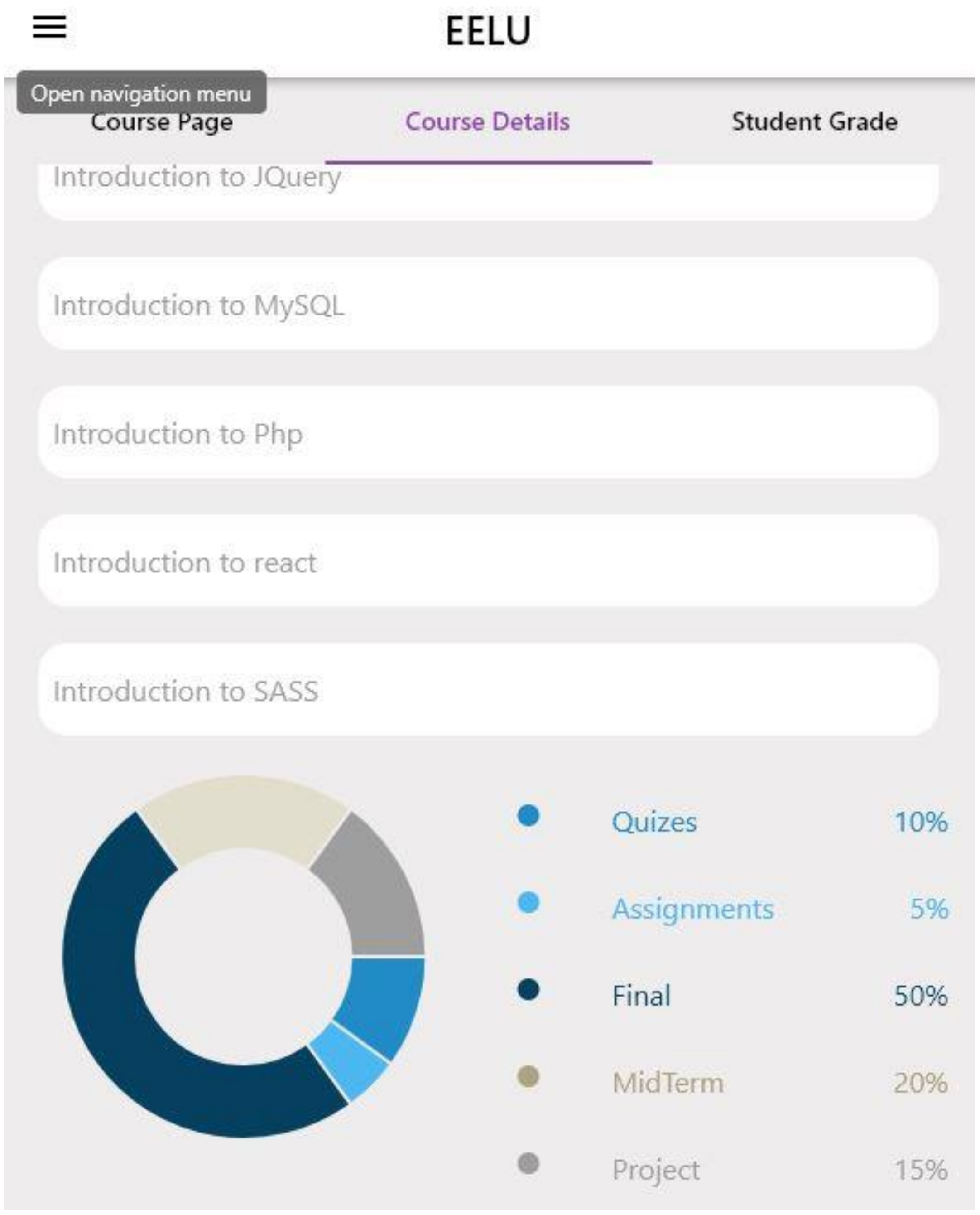

EELU
Open navigation menu
Your courses
john deo
21-10-2022
JS
HTML
CSS
Structure
Logic & data
Presentation & layout
Complete Web Tutorial
View Playlist



- **The Course Page within the Education Management System serves as a gateway to explore and access the main courses offered by the university. Students can navigate through a list of courses, each represented with essential details. The page features a prominent "View Playlist" button, providing a direct link to a second page that displays a comprehensive list of all available courses. This approach streamlines the user experience, allowing quick access to detailed course information when needed.**

Key Features:

- **Main Course Overview: Presents a concise overview of the main courses available within the university, showcasing essential details.**
- **"View Playlist" Button: A prominent button that allows users to navigate to a dedicated page with a comprehensive list of all courses.**
- **Efficient Navigation: Streamlines the exploration process by providing quick access to specific course details.**
- **User-Friendly Interface: Designed with clarity and simplicity, ensuring an intuitive experience for users**



Introduction to JQuery

Introduction to MySQL

Introduction to Php

Introduction to react

Introduction to SASS

●	Quizes	10%
●	Assignments	5%
●	Final	50%
●	MidTerm	20%
●	Project	15%

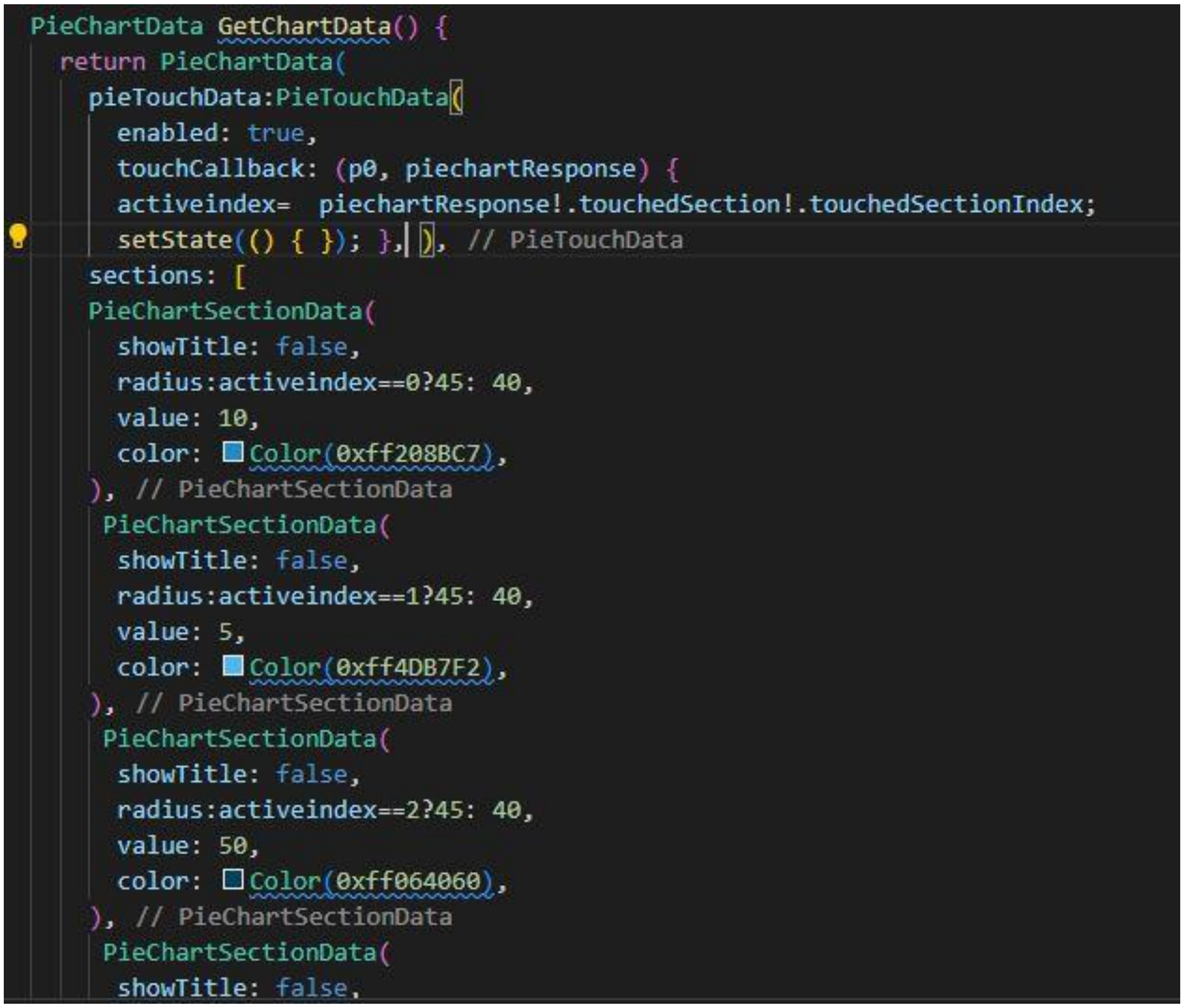
Course Details

- **The Course Detail Page provides students with in-depth information about a specific course, offering a comprehensive overview of the curriculum and assessment structure. Students can explore the topics covered in the course, as well as gain insights into the various** components such as quizzes, assignments, mid-term exams, finals, and projects. Additionally, the page outlines the percentage distribution of each component, allowing students to understand the weightage of assessments within the course. This detailed presentation empowers students to make informed decisions and approach their studies strategically.

Key Features:

- Curriculum Overview: Delivers a detailed breakdown of the topics and subjects covered in the course.
- Assessment Components: Specifies the different assessment components, including quizzes, assignments, mid-term exams, finals, and projects.
- Percentage Distribution: Outlines the percentage weightage of each assessment component, aiding students in understanding the contribution of each to the overall grade.
- Clear Presentation: Presents information in a clear and organized manner, enhancing readability and comprehension.

Chart code:



```
PieChartData GetChartData() {
  return PieChartData(
    pieTouchData:PieTouchData(
      enabled: true,
      touchCallback: (p0, piechartResponse) {
      activeindex=  piechartResponse!.touchedSection!.touchedSectionIndex;
      setState(() { }); }, ), // PieTouchData
    sections: [
    PieChartSectionData(
      showTitle: false,
      radius:activeindex==0?45: 40,
      value: 10,
      color: Color(0xff208BC7),
    ), // PieChartSectionData
     PieChartSectionData(
      showTitle: false,
      radius:activeindex==1?45: 40,
      value: 5,
      color: Color(0xff4DB7F2),
    ), // PieChartSectionData
     PieChartSectionData(
      showTitle: false,
      radius:activeindex==2?45: 40,
      value: 50,
      color: Color(0xff064060),
    ), // PieChartSectionData
     PieChartSectionData(
      showTitle: false,
```

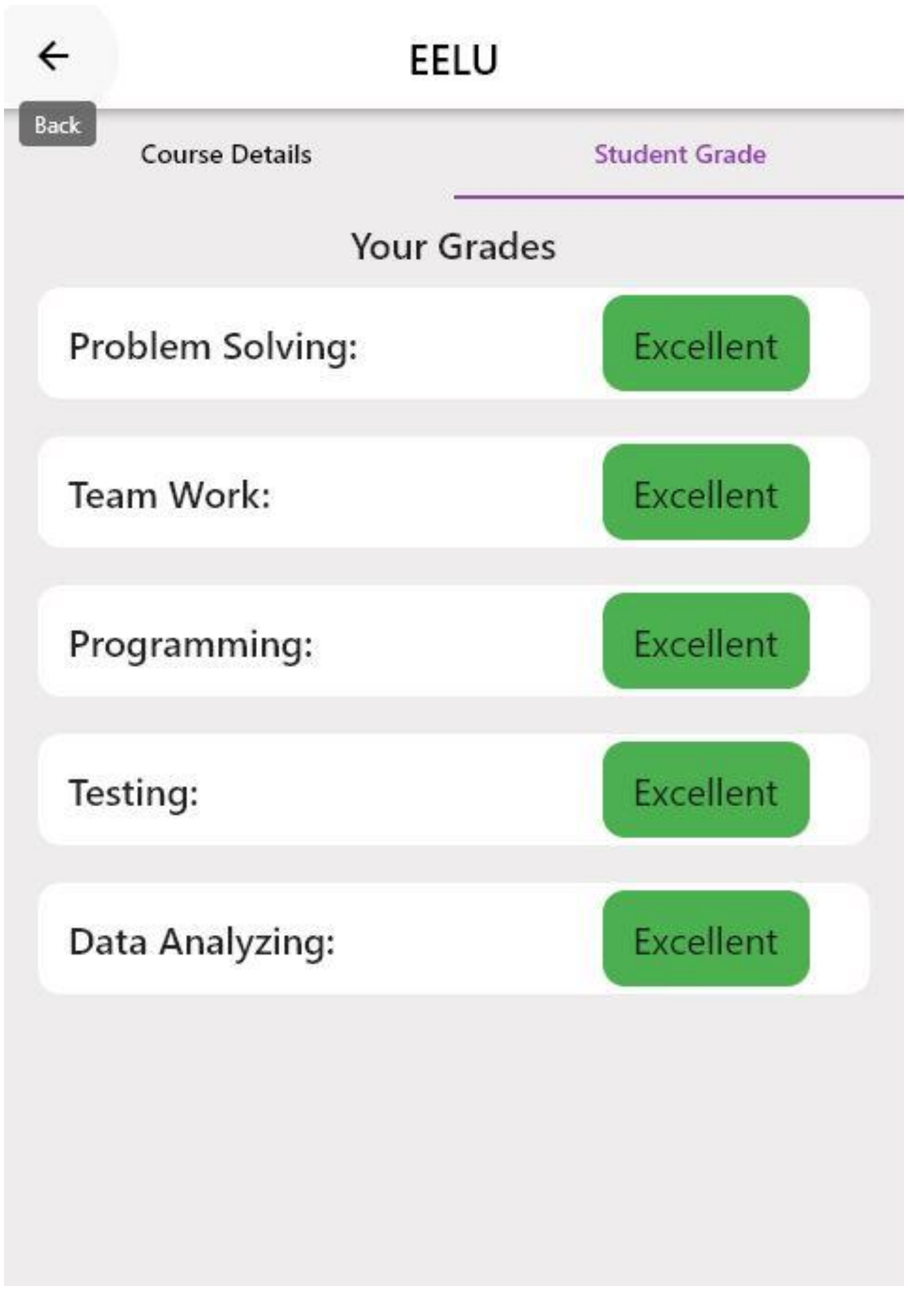
- **This Flutter code defines the configuration for a Pie Chart, utilizing the PieChartData class to specify the appearance and behavior of each section in the chart. The chart is designed to respond to touch interactions, allowing users to interactively explore different sections. The activeindex variable is used to highlight the touched section and trigger a state update to reflect the interaction visually.**

Key Features:

- **Interactive Touch: Enables touch interactions on the Pie Chart, providing a dynamic and engaging user experience.**
- **Dynamic Radius: Adjusts the radius of each chart section based on whether it is touched, enhancing visual feedback.**

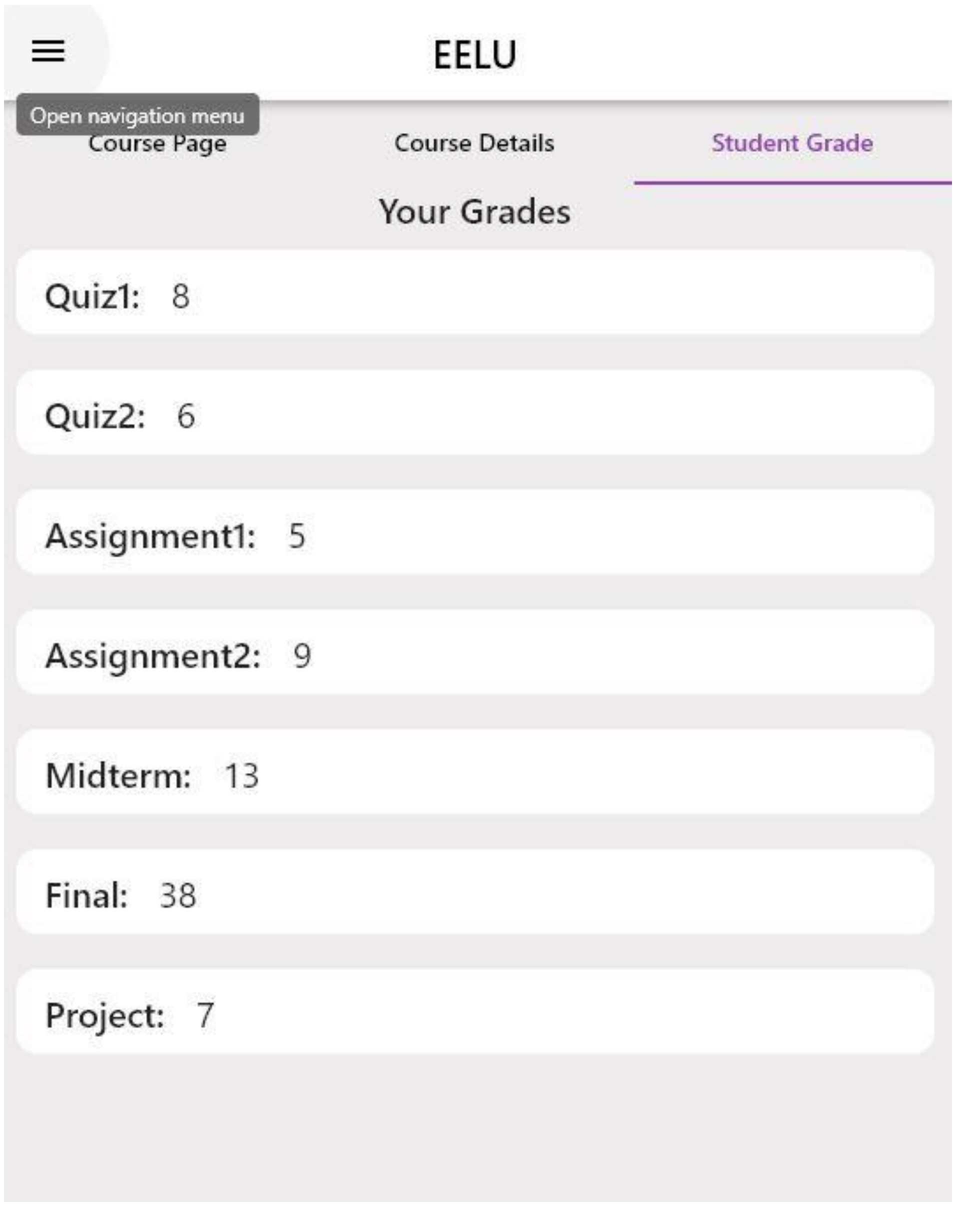
- **Colorful Sections: Specifies different colors for each chart section to enhance visual distinction.**
- **Section Values: Assigns values to each section to represent the proportional data it represents in the overall dataset.**

Student Grades:



Your Grades

Problem Solving:	Excellent
Team Work:	Excellent
Programming:	Excellent
Testing:	Excellent
Data Analyzing:	Excellent



Quiz1: 8

Quiz2: 6

Assignment1: 5

Assignment2: 9

Midterm: 13

Final: 38

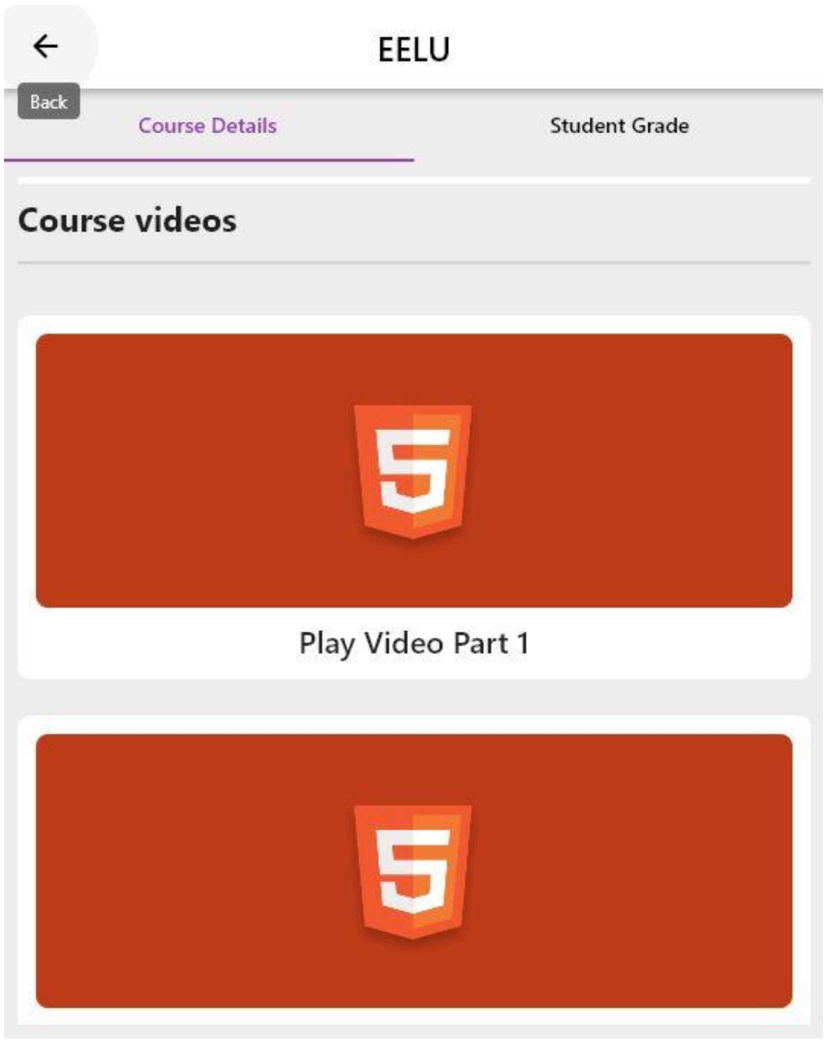
Project: 7

- **The Student's Grade and Skills Page offers a comprehensive overview of a student's academic performance and skills within the Education Management System. Students can easily access their grades in quizzes, assignments, finals, mid-term exams, and projects. Additionally, this page introduces a novel feature that provides insights into a student's skills, including programming, testing, teamwork, data analyzing, and problem-solving. By identifying areas of strength and weakness, this feature empowers students to take targeted actions to enhance their skills and improve their overall academic performance.**

Key Features:

- **Grade Display: Presents a breakdown of grades in quizzes, assignments, finals, mid-term exams, and projects, giving students a clear understanding of their academic performance.**
- **Skills Assessment: Introduces a new feature that evaluates skills in programming, testing, teamwork, data analyzing, and problem-solving.**
- **Visual Feedback: Utilizes visual elements such as charts or graphs to present grades and skills in an easily digestible format.**
- **Actionable Insights: Identifies areas for improvement in skills, empowering students to focus on specific areas for growth.**

Course Videos

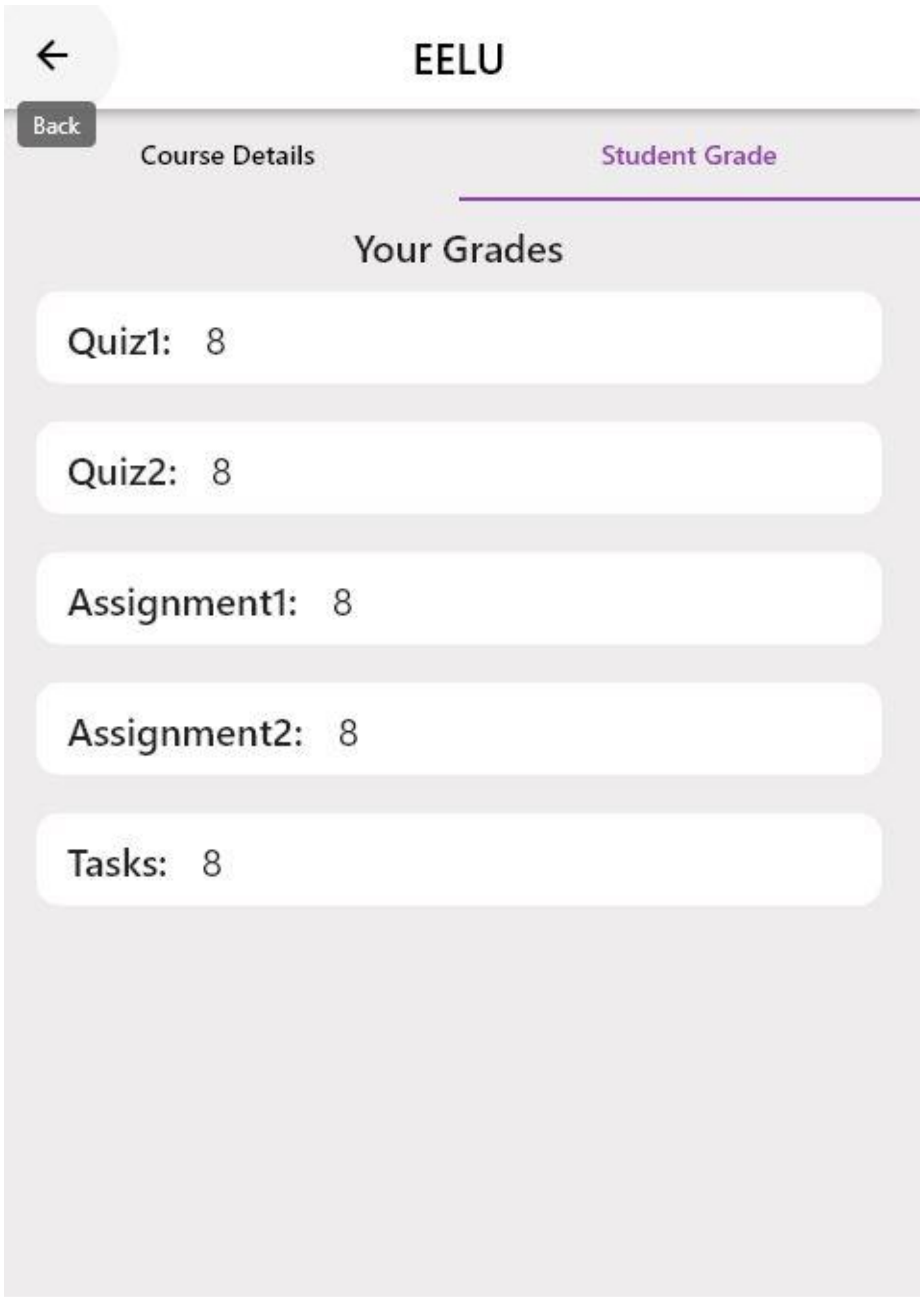


- **The Lectures and Study Page is an integral component of the Education Management System, allowing students to delve into the course content in a structured manner. Accessed by clicking "View Playlist" on a specific course, this page provides a curated list of lectures associated with that course. Students can seamlessly navigate through the lectures, accessing study materials, video content, and additional resources. This page is designed to enhance the learning experience, offering a centralized location for students to study, review, and engage with the course material at their own pace.**

Key Features:

- **Structured Lecture List: Presents a curated and organized list of lectures associated with the selected course.**
- **Study Materials: Provides access to study materials, including video content, presentations, documents, and any additional resources.**
- **Navigation Controls: Enables students to navigate through lectures, offering the flexibility to revisit previous sessions or explore upcoming content.**
- **User-Friendly Interface: Designed with clarity and simplicity, ensuring an intuitive experience for efficient learning.**

Each Course Grades Details:



Your Grades

Quiz1: 8

Quiz2: 8

Assignment1: 8

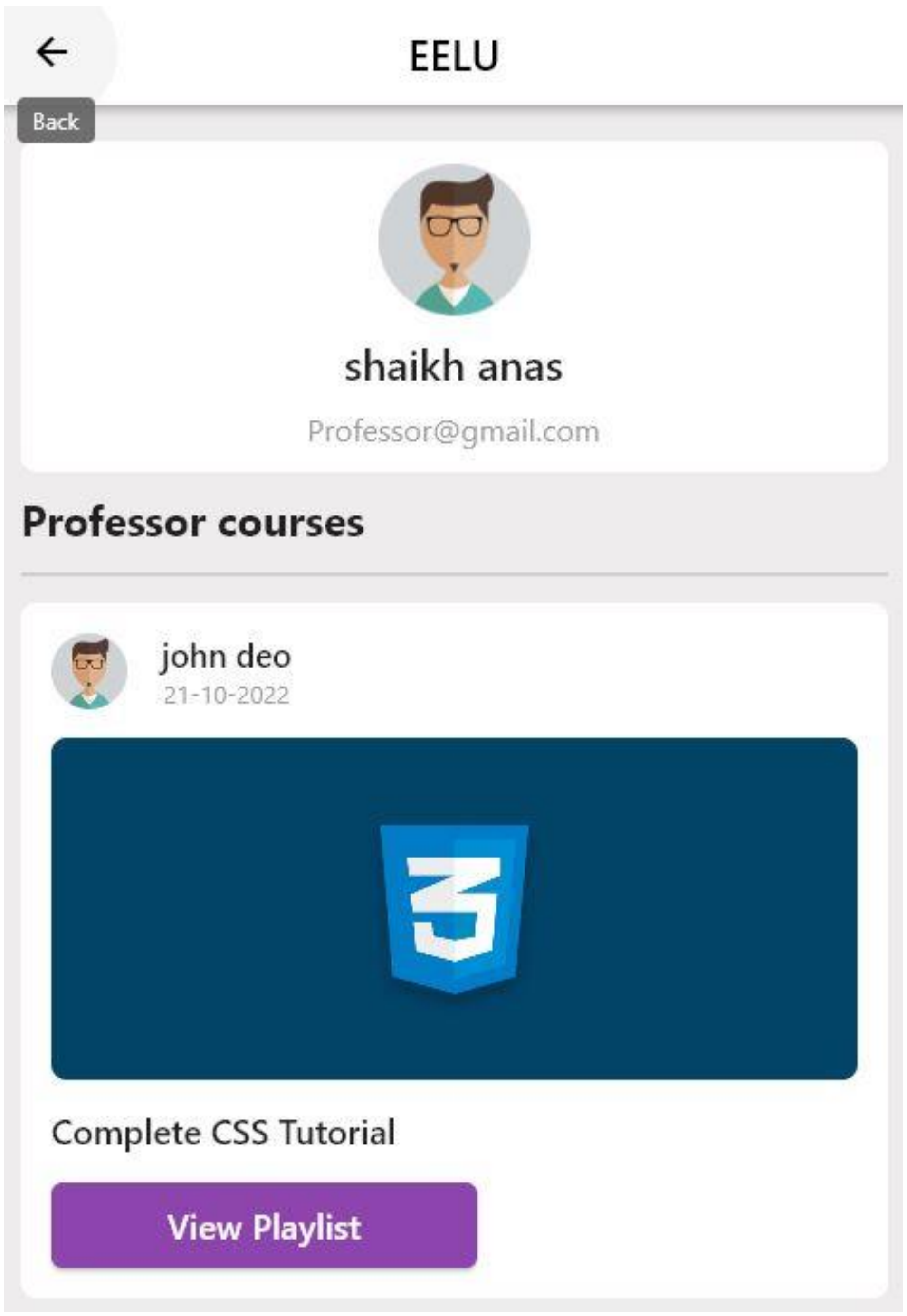
Assignment2: 8

Tasks: 8

- **The Course Grades Page, accessible through the tab bar navigation, provides students with a consolidated view of their academic performance within a specific course. By selecting the "Course Grades" tab, students can seamlessly access their grades in quizzes, assignments, and other tasks associated with the course. This page offers a detailed breakdown of grades, allowing students to monitor their progress, identify areas for improvement, and stay informed about their overall standing in the course.**

Key Features:

- **Consolidated View: Presents a consolidated and organized display of grades in quizzes, assignments, and other relevant tasks for the selected course.**
- **Detailed Breakdown: Provides a detailed breakdown of individual assessments, allowing students to understand their performance in specific areas.**
- **Progress Monitoring: Facilitates ongoing monitoring of academic progress, enabling students to track their achievements and identify areas for improvement.**
- **User-Friendly Interface: Incorporates a user-friendly design within the tab bar navigation, ensuring easy access to essential academic information.**



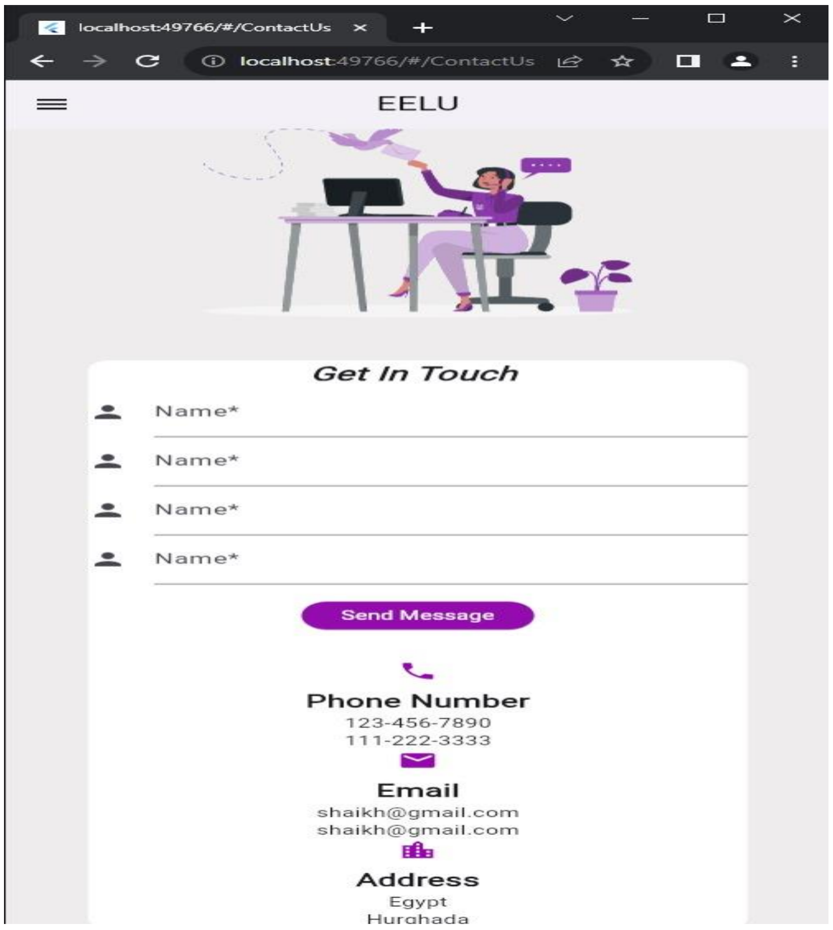
Professor Page

- **The Professor Page serves as a hub for students within the Education Management System to explore information about all the professors associated with the institution. Students can access a comprehensive** list of professors, view individual profiles, and explore the courses each professor is instructing. This page enhances the student experience by providing valuable insights into the academic staff, fostering a better understanding of the expertise and courses offered by each professor.

Key Features:

- Professor Directory: Presents a directory of all professors affiliated with the institution.
- Individual Profiles: Allows students to click on each professor to view detailed profiles, providing information about academic background, expertise, and contact details.
- Course Listings: Displays a list of courses associated with each professor, facilitating course selection and enrollment.
- User-Friendly Navigation: Incorporates an intuitive interface, enabling students to easily explore and access information about professors and their courses

Contact US:


localhost:49766/#/ContactUs

EELU

Get In Touch

Name*

Name*

Name*

Name*

Send Message

Phone Number
123-456-7890
111-222-3333

Email
shaikh@gmail.com
shaikh@gmail.com

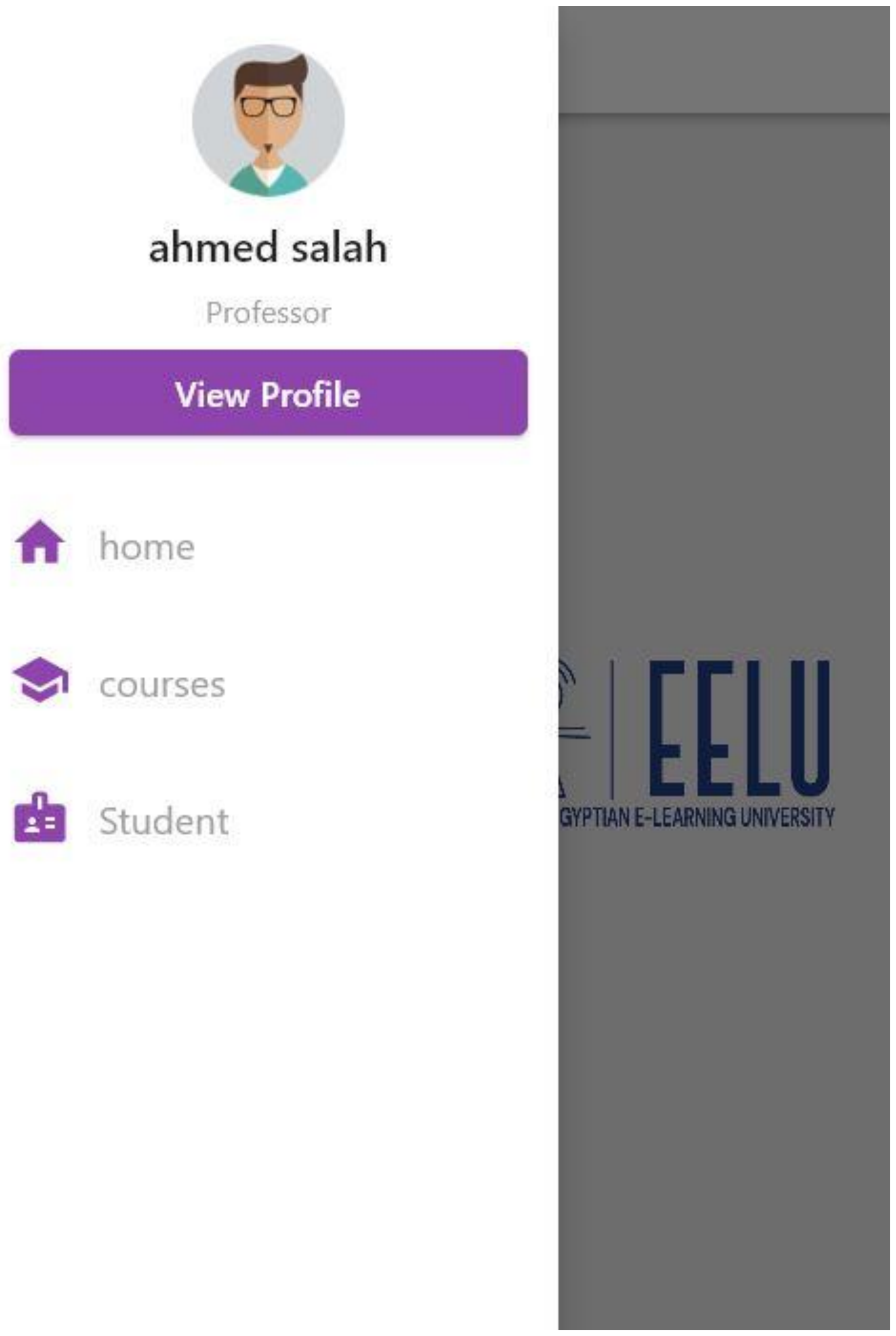
Address
Egypt
Hurghada

- **The Contact Us Page is a vital resource within the Education Management System, offering students a direct means of communication with the university. This page provides essential contact information, including the university's address, phone number, and email. Students can use this information to reach out for inquiries, feedback, or compliments, fostering a transparent and responsive communication channel between students and the university administration.**

Key Features:

- **Contact Information: Displays the university's physical address, phone number, and email for easy access.**
- **Communication Channel: Serves as a centralized point for students to initiate contact with the university administration or relevant departments.**

Home Page: Professor UI

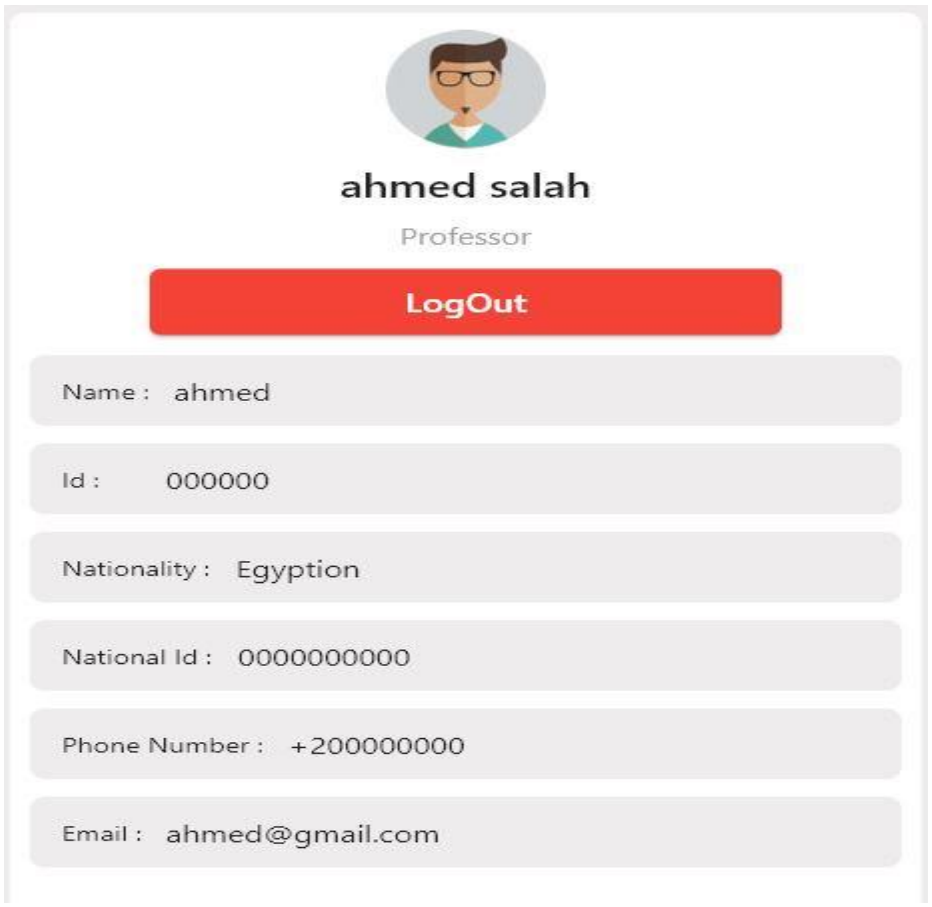


- The Professor's Home Page serves as the central hub for academic professionals within the Education Management System. A key component of this page is the navigation drawer, providing professors with seamless access to various sections of the application. The drawer includes essential options such as "Home," "Courses," "Students," and "Profile," offering quick navigation to different functionalities. This design ensures an efficient and user-friendly experience, allowing professors to manage courses, interact with students, and access personal information effortlessly.

Key Features:

- **Navigation Drawer: Features a drawer accessible from the home page, offering quick links to essential sections of the application.**
- **Home Section: Allows professors to return to the home page for a centralized view of information.**
- **Courses Section: Navigates professors to a dedicated page for managing and overseeing their courses.**
- **Students Section: Provides access to information about enrolled students, facilitating communication and academic management.**
- **Profile Section: Allows professors to view and update their personal profiles, ensuring accurate and up-to-date information.**

Professor Profile


ahmed salah

Professor

Name : ahmed

Id : 000000

Nationality : Egyption

National Id : 0000000000

Phone Number : +200000000

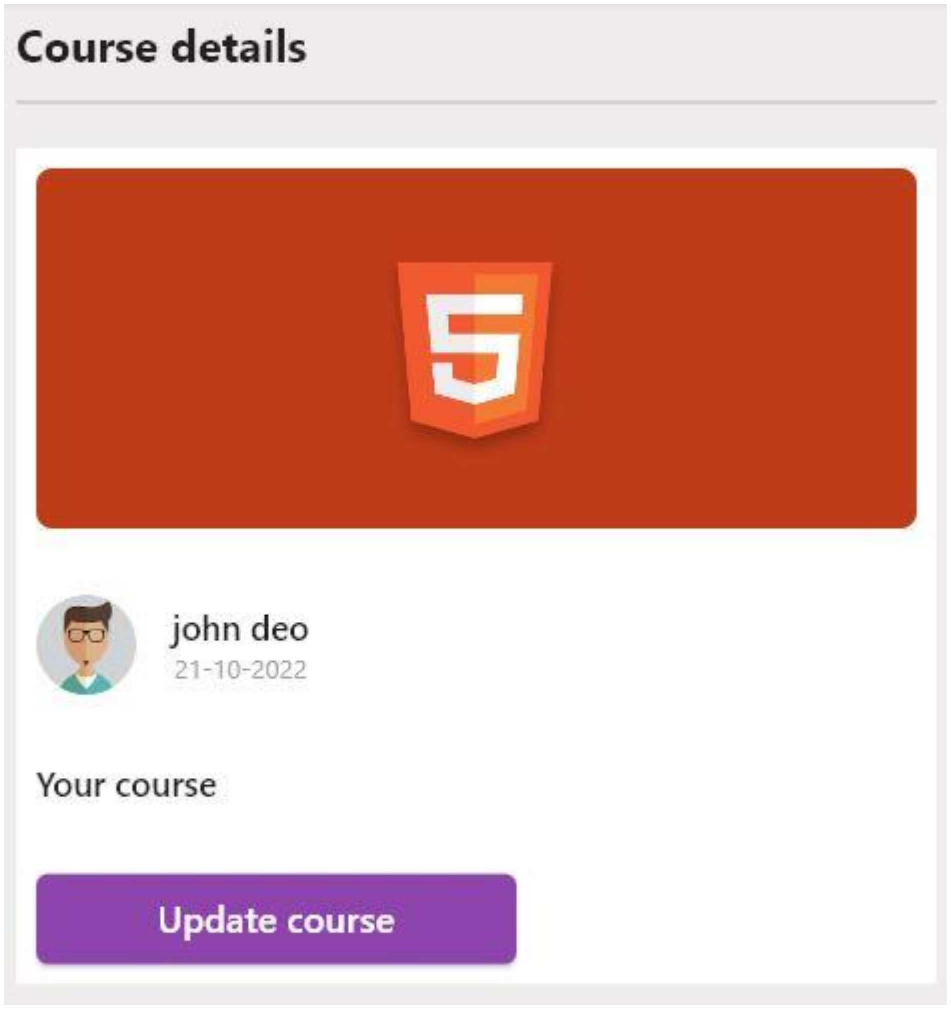
Email : ahmed@gmail.com

- **The Professor Profile Page is a dedicated space within the Education Management System, providing professors with comprehensive information about their professional identity.**

Key Features:

- **Personal Information: Displays essential personal details such as professor ID, national ID, and phone number.**
- **Profile Management: Offers the ability to view and potentially update profile information for accuracy.**
- **Logout Option: Provides a secure logout button for professors to end their session when necessary.**

Course Page: Professor UI

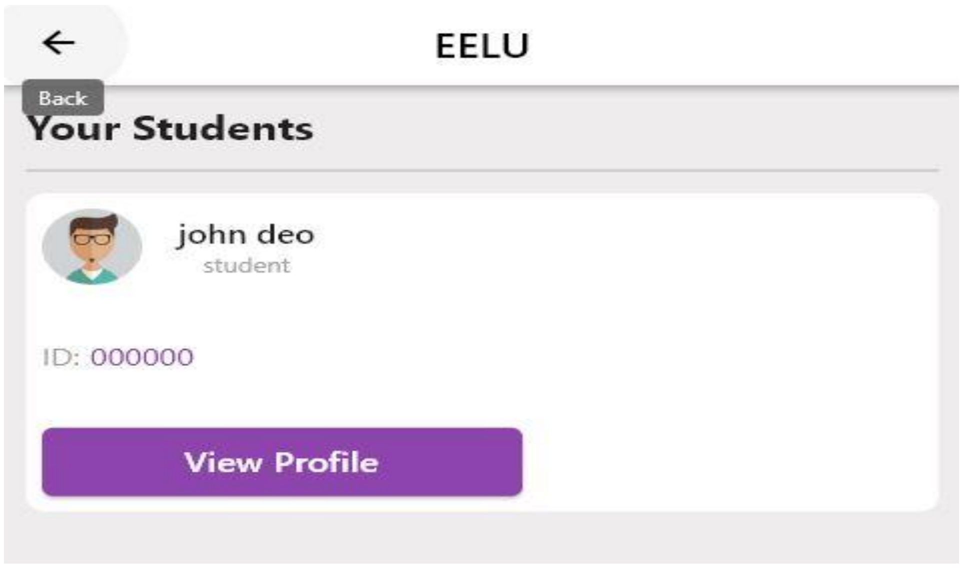


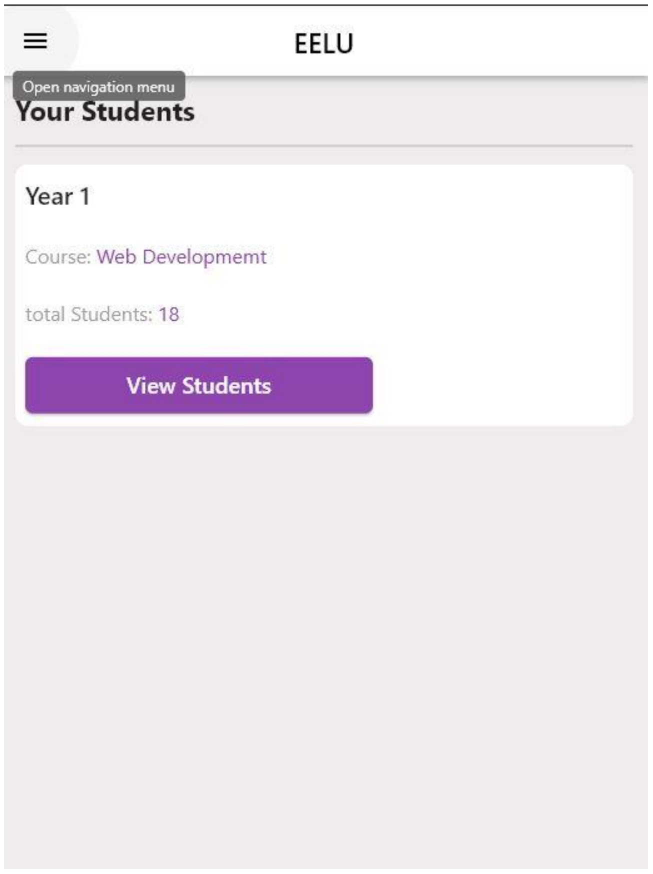
- **The Professor Courses Page serves as a centralized platform for academic professionals to manage their courses efficiently within the Education Management System. Professors can seamlessly view a list of their assigned courses, add new lectures to enhance course content, remove outdated lectures, and update course details as needed**

Key Features:

- **Course Listing: Displays a comprehensive list of courses assigned to the professor.**
- **Update Course Details: Provides the ability to update course information, ensuring accuracy and relevance.**
- **User-Friendly Interface: Offers an intuitive interface for seamless navigation and efficient course management**

Students Page: Professor UI





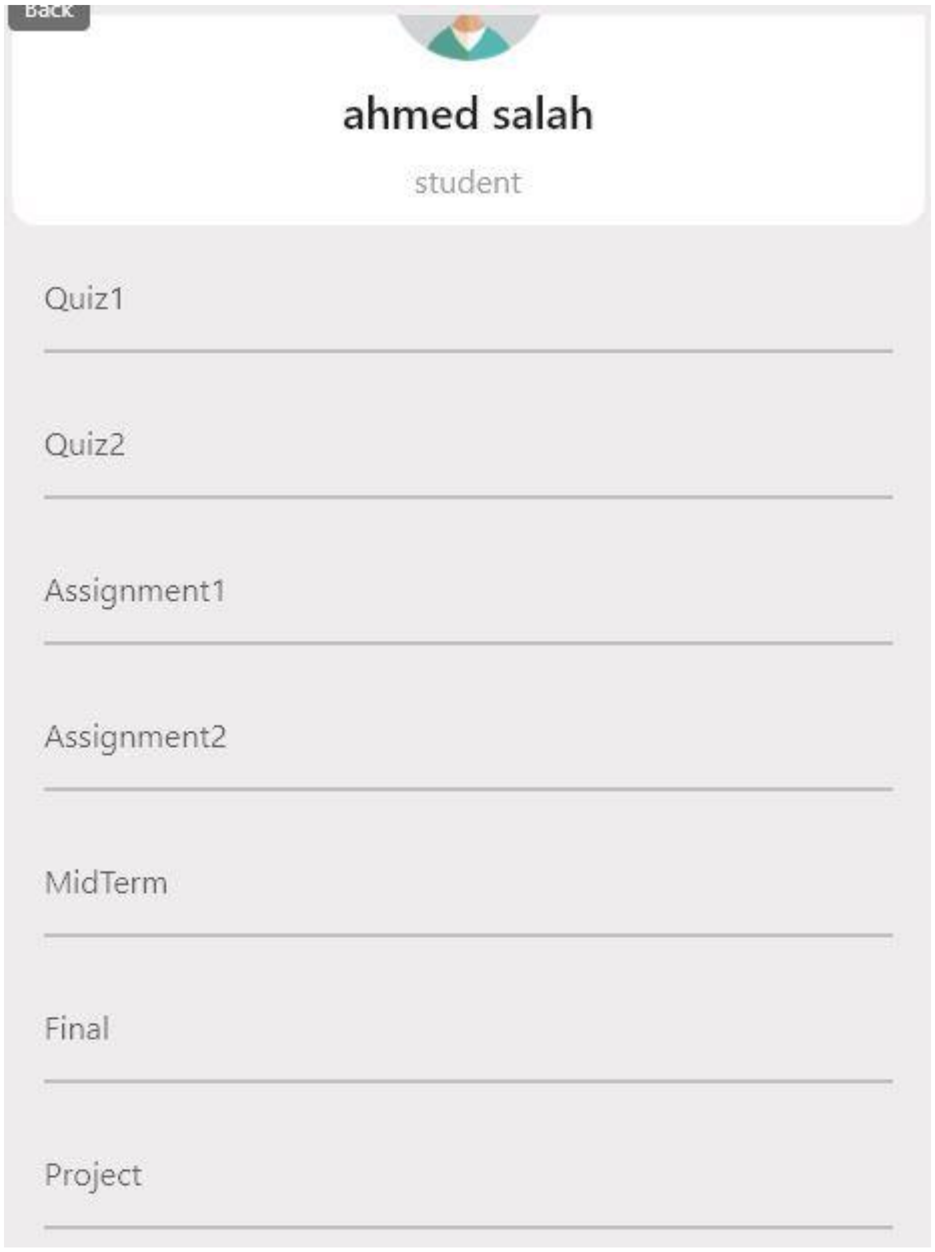
- **The Professor Students Page empowers academic professionals within the Education Management System to efficiently manage and interact with students enrolled in their courses. Professors can access a comprehensive list of classes they are teaching and, within each class, view detailed information about students, categorized by their academic year. This page enhances communication and engagement, allowing professors to have a clear overview of their students and access details specific to each academic year.**

Key Features:

- **Class Listing: Displays a list of classes taught by the professor, providing an organized view of course assignments.**
- **Student Categorization: Categorizes students within each class based on their academic year, facilitating targeted communication.**
- **Detailed Student Information: Allows professors to click on specific academic years to view detailed information about all students within that year.**

- **User-Friendly Interface: Offers an intuitive interface for seamless navigation and efficient student management.**

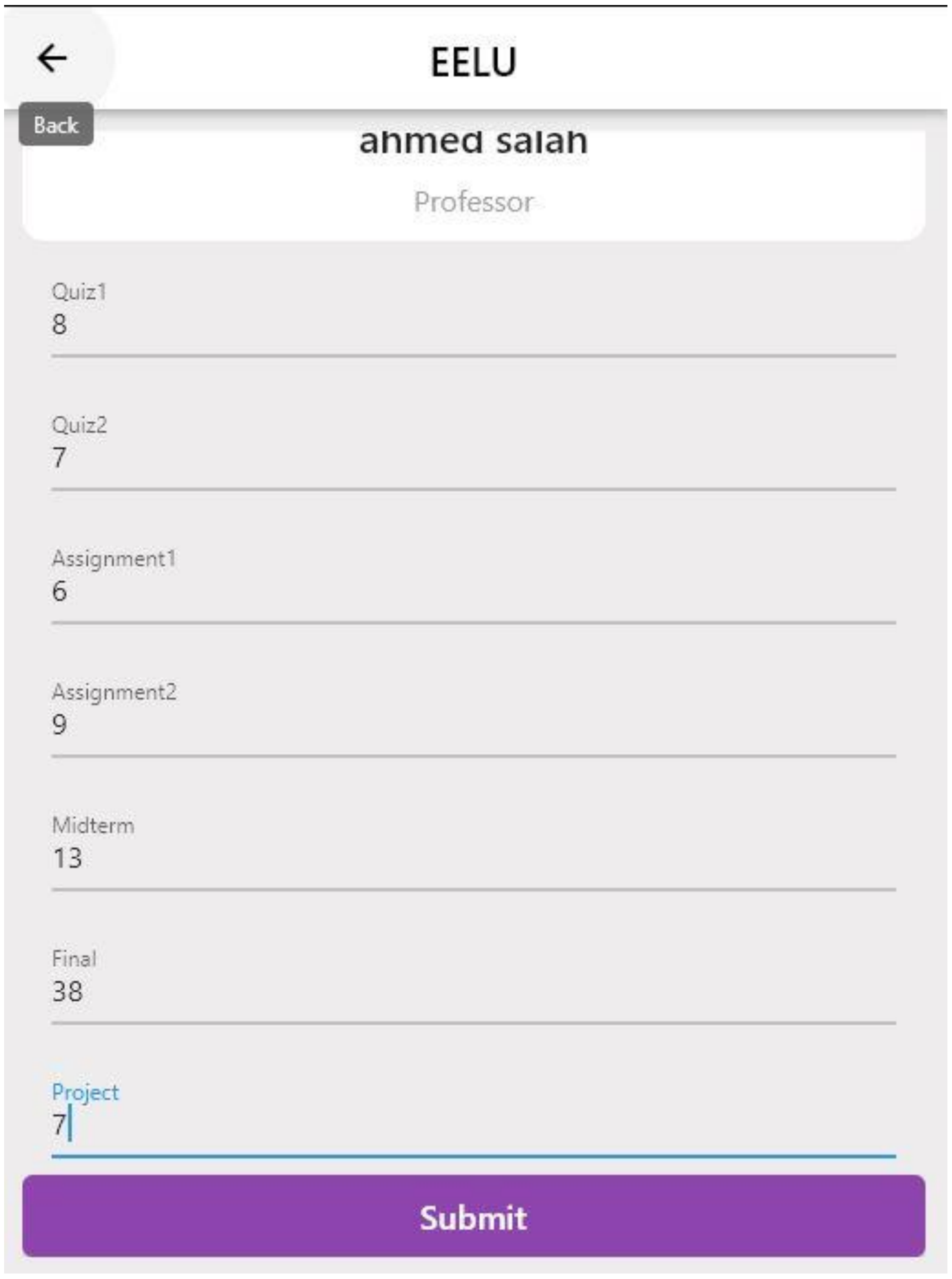
Students Grade: Professor UI



- **The Professor Students Grades Page is a vital tool for academic professionals within the Education Management System, offering a streamlined approach to manage and update grades for quizzes, assignments, finals, midterms, and projects. Professors can input and modify grades for individual students, providing accurate and timely feedback on their academic performance. Additionally, this page allows students to access**

and view their grades, fostering transparency and communication between professors and students.

- ***Key Features:***
- **Grade Entry: Enables professors to input and update grades for quizzes, assignments, finals, midterms, and projects.**
- **Student-Specific Grades: Allows professors to assign grades individually for each student in the class.**
- **Transparent Feedback: Fosters transparency by providing students with access to their grades for self-assessment.**
- **User-Friendly Interface: Offers an intuitive interface for efficient grade management and communication.**



ahmed salah

Professor

Quiz1
8

Quiz2
7

Assignment1
6

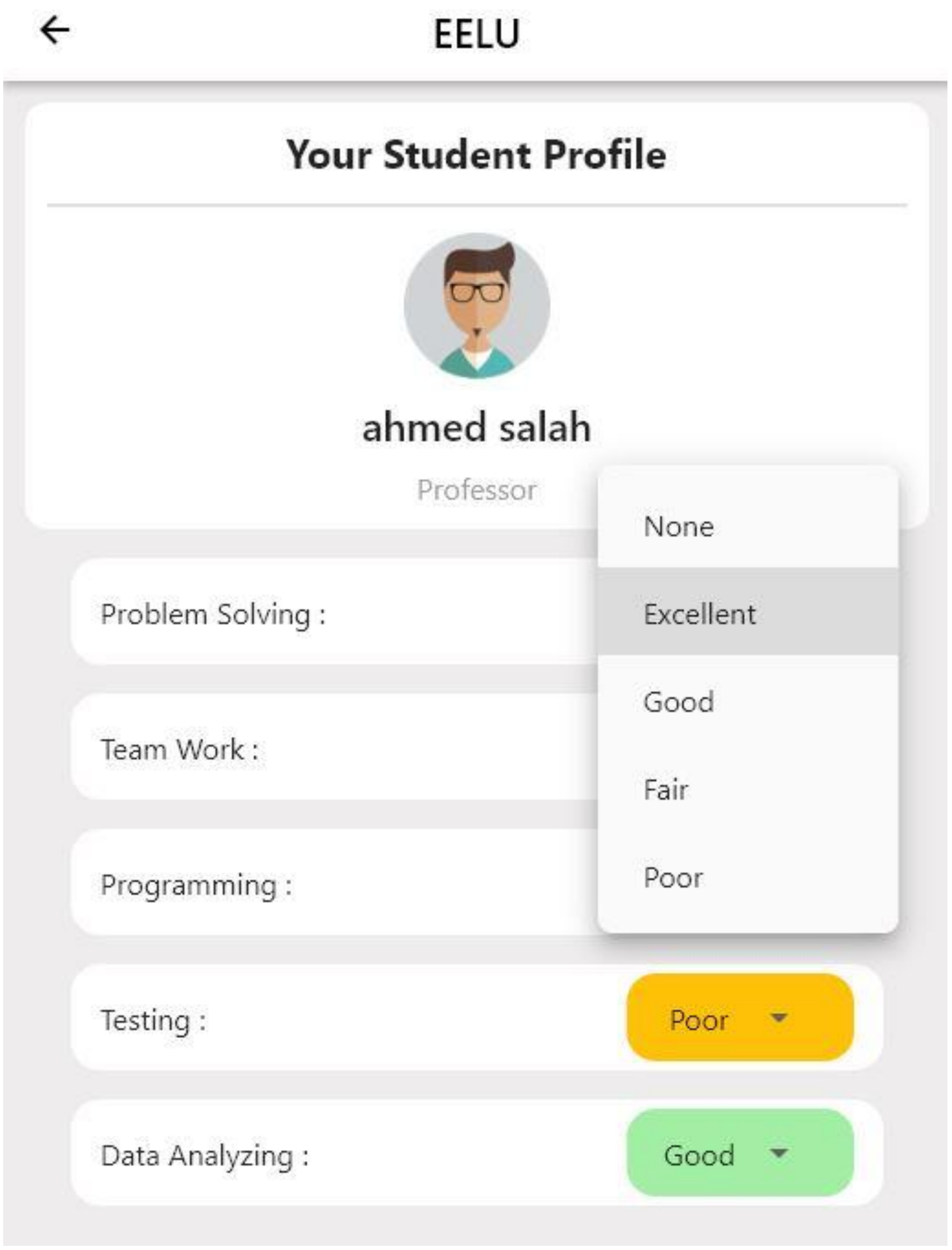
Assignment2
9

Midterm
13

Final
38

Project
7

Professor Grading Students (quizzes, assignments, final, midterm, projects)


EELU

Your Student Profile

ahmed salah

Professor

Problem Solving :

Team Work :

Programming :

Testing : Poor

Data Analyzing : Good

Professor Grading Student Skills

Evaluation Process Overview

Throughout the academic term, students were subjected to a comprehensive evaluation protocol to measure their understanding and

retention of course material. This assessment was segmented into various components to capture the multifaceted nature of learning.

Quizzes: Short, focused quizzes were administered to ensure ongoing engagement with the course content. Students were evaluated on their ability to quickly recall and apply key concepts.

- **Quiz 1: Focused on foundational knowledge, scored 8 out of a possible score.**
- **Quiz 2: Covered intermediate topics, scored 7, reflecting a consistent grasp of the material.**

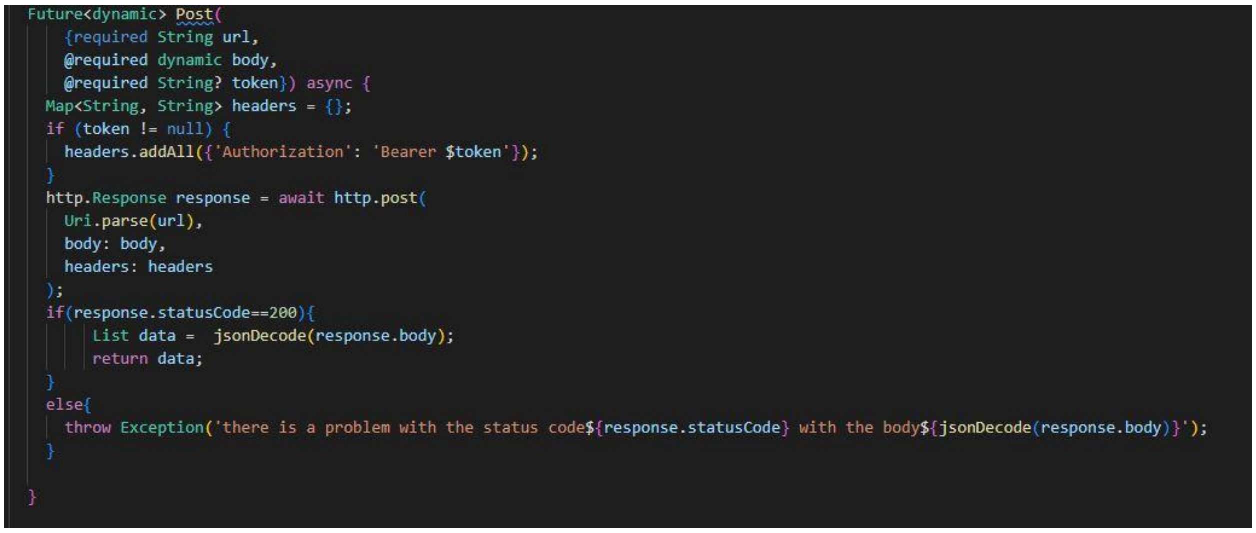
Assignments: The assignments served as a practical application of theoretical concepts, allowing students to demonstrate their problem-solving skills.

- **Assignment 1: Gauged the basic application of course principles, scored 6, indicating room for improvement.**
- **Assignment 2: Demanded a deeper understanding and creative application, scored 9, showcasing notable improvement.**

Midterm and Final Exams: These cumulative assessments measured the students' ability to synthesize and articulate their knowledge comprehensively.

- **Midterm: A rigorous assessment covering all topics to that point, scored 13.**
- **Final: The conclusive exam, with a score of 38, tested the students on the entirety of the course content.**

Project: The capstone project was a synthesis of the semester's learning, scored 7, reflecting the student’s capability to undertake complex tasks and produce a substantive piece of work.



```
Future<dynamic> Post(
    {required String url,
    @required dynamic body,
    @required String? token}) async {
  Map<String, String> headers = {};
  if (token != null) {
    headers.addAll({'Authorization': 'Bearer $token'});
  }
  http.Response response = await http.post(
    Uri.parse(url),
    body: body,
    headers: headers
  );
  if(response.statusCode==200){
      List data =  jsonDecode(response.body);
      return data;
  }
  else{
    throw Exception('there is a problem with the status code${response.statusCode} with the body${jsonDecode(response.body)}');
  }

}
```

Function Overview

Function Name: Post

Description:

This function is designed to facilitate communication with a web service by sending HTTP POST requests. It is an asynchronous function that returns dynamic data, possibly in the form of JSON, from the server. It accepts a URL endpoint, a body for the request, and optionally a bearer token for authentication.

Parameters

url (String): The endpoint to which the POST request will be sent.

body (dynamic): The data to be sent to the server. This must be a dynamic object that can be serialized into a JSON format.

token (String, optional): A bearer token for authentication. If provided, it is included in the headers of the request.

Return Type

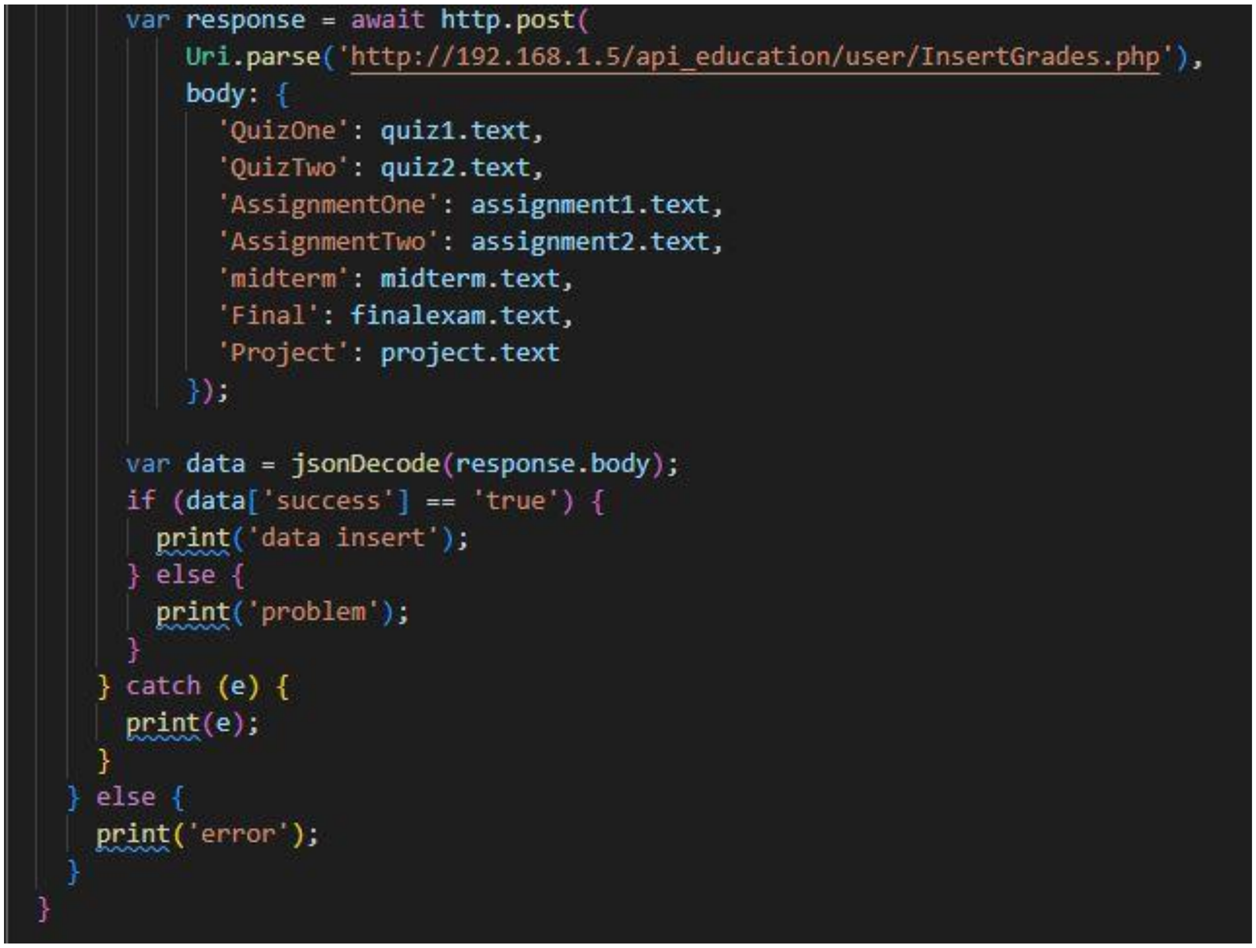
Future<dynamic>: Returns a Future that resolves to dynamic data type. The response from the server is expected to be JSON-formatted data.

Error Handling

The function includes error handling to manage non-200 HTTP responses. If the server responds with a status code other than 200, an exception is thrown with details about the status code and the response body.

Considerations

Mention any special considerations, like handling large amounts of data, ensuring the security of the token, or compatibility with different types of server responses.



```
        var response = await http.post(
          Uri.parse('http://192.168.1.5/api_education/user/InsertGrades.php'),
          body: {
            'QuizOne': quiz1.text,
            'QuizTwo': quiz2.text,
            'AssignmentOne': assignment1.text,
            'AssignmentTwo': assignment2.text,
            'midterm': midterm.text,
            'Final': finalexam.text,
            'Project': project.text
          });

        var data = jsonDecode(response.body);
        if (data['success'] == 'true') {
          print('data insert');
        } else {
          print('problem');
        }
      } catch (e) {
        print(e);
      }
    } else {
      print('error');
    }
  }
```

Function Implementation Overview

Functionality Description:

This code snippet is part of a Dart-based application that posts academic grades data to a remote server. The grades include quizzes, assignments,

midterms, the final exam, and a project. It uses an HTTP POST request to send the data to a PHP script hosted on the server.

Code Description

URL: The function posts data to a specific URL (http://192.168.1.5/api/education/user/InsertGrades.php). This endpoint is likely designed to receive grade data and store it in a database.

Body: The request's body includes keys for two quizzes, two assignments, a midterm, a final exam, and a project. Each of these is expected to have an associated text value which is presumably input by the user or fetched from a user interface.

Error Handling

HTTP Response Handling: The code checks if the HTTP request was successful by evaluating the success field in the JSON response. If success is true, it prints "data insert", indicating the grades were successfully stored. Otherwise, it prints "problem".

Exception Handling: The code includes a catch block to handle any exceptions that might occur during the HTTP request, logging the exception details.

HTTP Error Handling: If the response status code is not successful (i.e., not caught by the earlier logic), it prints "error".

Security Considerations

Highlight the importance of securing the endpoint and consider discussing measures like HTTPS, authentication, and input validation to protect against common vulnerabilities.

Skills Assessment

The student profile evaluation is a qualitative measure of a student's skills and attributes, essential for both academic success and professional competency.

- **Problem Solving: Rated as 'Excellent', demonstrating outstanding analytical abilities and innovative thinking.**
- **Team Work: Judged as 'Good', the student showed a commendable ability to work collaboratively, contributing positively to group dynamics.**
- **Programming: Evaluated as 'Poor', this area was identified as a growth opportunity for the student, suggesting a need for further practice and learning.**
- **Testing: Also rated as 'Poor', indicating that the student may benefit from additional training in systematic evaluation and quality assurance.**
- **Data Analyzing: Assessed as 'Good', the student has shown a solid ability to interpret and analyze data effectively.**

These assessments are integral to the academic process, providing students with a clear indication of their strengths and areas for improvement, as well as informing instructional strategies for educators.

4. Implementation

Development Process

1. Methodology

- project was developed using an Agile Development methodology, emphasizing iterative progress, flexibility, and stakeholder feedback. This approach facilitated rapid response to changes and continuous improvement.
- Development was divided into sprints, each lasting two weeks, with specific goals and deliverables. These sprints allowed for regular reassessment of the project direction and priorities.
- Team Collaboration was integral, utilizing tools like Jira for task tracking and Git for version control. Daily stand-up meetings ensured alignment and effective communication.

2. Planning and Analysis

- Requirement Gathering: Initial stages involved thorough discussions with stakeholders and surveys to define system requirements accurately.
- Feasibility Study: A feasibility study was conducted, assessing the technical, economic, and operational aspects to ensure the project's viability.

3. Design

- System Design: The design phase involved creating detailed architecture diagrams and adopting design patterns, notably MVVM for the Flutter-based mobile app.
- UI/UX Design: The user interface and experience were crafted using tools like Figma, focusing on intuitive navigation and aesthetic appeal.

B. Key Functionalities

1. User Authentication

- **A robust authentication system was implemented for user login and registration, with secure session management to maintain security and user state.**

2. Course Management

- **The system allows administrators and instructors to create, modify, and remove course offerings, with users able to view and interact with these courses.**

3. Student Enrollment and Grade Tracking

- **The platform facilitates student enrollment in courses and features a system for tracking and displaying grades.**

4. Mobile App Functionalities

- **The mobile app, developed in Flutter, includes features like push notifications, course browsing, and tailored interactive elements for mobile users.**

C. Testing and Deployment

1. Testing

- **The project underwent thorough testing, including unit testing for components and integration testing to ensure system cohesion.**
- **User acceptance testing was conducted to validate the functionality against user requirements.**

2. Deployment

- **Both the web application and mobile app were deployed using cloud services to ensure scalability and reliability.**
- **Pre-deployment strategies, such as load testing, were utilized to ensure a smooth rollout.**

5.Project Summary

Title: Higher Education Management System

Objective: The project aimed to develop a robust, user-friendly management system specifically designed for higher education institutions. This system was intended to streamline administrative processes, enhance the educational experience, and provide cohesive access across both web and mobile platforms.

Key Features and Functionalities:

Course Management: A comprehensive module for managing course details, schedules, and materials, enabling easy access for both faculty and students.

Student Enrollment and Tracking: Facilitates student enrollment in courses with features for tracking academic progress, attendance, and grades.

Secure User Authentication: Implements a robust authentication mechanism, ensuring secure access for students, faculty, and administrators.

Mobile Application: A cross-platform mobile app developed in Flutter, offering students and faculty convenient access to system features on-the-go.

API Integration: RESTful APIs designed for seamless integration with existing educational platforms and systems, enhancing data consistency and operational efficiency.

Technical Architecture:

Web Frontend: Developed using HTML, CSS, and JavaScript, ensuring a responsive and intuitive user interface.

Backend: Powered by PHP with the Laravel framework, adopting an MVC architecture for efficient data processing and application logic.

Database: Utilizes a MySQL database for robust and reliable data storage and management.

Mobile App: Built with Flutter, providing a consistent and seamless user experience across Android and iOS devices.

Development Approach: The project was executed using an Agile methodology, allowing for iterative development, continuous feedback, and adaptive planning.