**Database Development and Class Registration**

Jonathan Schaub

The University of Arizona Global Campus

CST 499 Capstone for Computer Software Technology

Dr. Charmelia Butler

October 23, 2023

**Figure 1**

*Reworked landing page*

A screenshot of a computer

Description automatically generated

**Figure 2**

*Course Enrollment Catalogue, course\_enroll.php*

A screenshot of a computer

Description automatically generated

**Figure 3**

*List of enrolled courses for student roger@123.com, my\_courses.php*

A screenshot of a computer

Description automatically generated

**Figure 4**

*PHPMyAdmin view of tblcourses*

A screenshot of a computer

Description automatically generated

**Figure 5**

*PHPMyAdmin view of tblenrollment*

A screenshot of a computer

Description automatically generated

**Figure 6**

*course\_enroll.php*

<?php

require('db.php');

session\_start();

$courses = mysqli\_query($con, "SELECT \* FROM `tblcourses`");

if(isset($\_POST['course\_id'])) {

$course\_id = $\_POST['course\_id'];

$email = $\_SESSION['email'];

$checkEnrollment = mysqli\_query($con, "SELECT \* FROM `tblenrollment` WHERE email='$email' AND course\_id='$course\_id'");

if(mysqli\_num\_rows($checkEnrollment) == 0) {

mysqli\_query($con, "INSERT INTO `tblenrollment` (email, course\_id) VALUES ('$email', '$course\_id')");

}

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title> Course Enrollment </title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>

<script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>

</head>

<body>

<?php include 'master.php'; ?>

<div class="container">

<h2>Enroll in a Course</h2>

<ul>

<?php while($course = mysqli\_fetch\_assoc($courses)): ?>

<li>

<?php echo $course['course\_name']; ?> -

<?php echo $course['course\_code']; ?>

<form method="post" action="course\_enroll.php">

<input type="hidden" name="course\_id" value="

<?php echo $course['course\_id']; ?>">

<input type="submit" value="Enroll">

</form>

</li>

<?php endwhile; ?>

</ul>

</div>

<?php include 'footer.php'; ?>

</body>

</html>

**Figure 7**

*my\_courses.php*

<?php

require('db.php');

session\_start();

$email = $\_SESSION['email'];

if(isset($\_POST['unenroll\_id'])) {

$course\_id = $\_POST['unenroll\_id'];

mysqli\_query($con, "DELETE FROM `tblenrollment` WHERE email='$email' AND course\_id='$course\_id'");

}

$enrollments = mysqli\_query($con, "SELECT \* FROM `tblenrollment` JOIN `tblcourses` ON tblenrollment.course\_id = tblcourses.course\_id WHERE email='$email'");

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title> Enrolled Courses </title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>

<script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>

</head>

<body>

<?php include 'master.php'; ?>

<div class="container">

<h2>My Courses</h2>

<ul>

<?php while($course = mysqli\_fetch\_assoc($enrollments)): ?>

<li>

<?php echo $course['course\_name']; ?> -

<?php echo $course['course\_code']; ?>

<form method="post" action="my\_courses.php">

<input type="hidden" name="unenroll\_id" value="

<?php echo $course['course\_id']; ?>">

<input type="submit" value="Unenroll">

</form>

</li>

<?php endwhile; ?>

</ul>

</div>

<?php include 'footer.php'; ?>

</body>

</html>

**Introduction**

Adding a course registration and management functionality to an existing student registration and login portal adds a few layers of complexity, from creating relationships between the users and courses to querying the database to return an array of data. This paper will examine the use of PHP, HTML, and SQL to create the course\_enroll.php and my\_courses.php webpages with the functionality of students being able to enroll and unenroll in courses.

First, let’s examine the construction of course\_enrollment.php:

<?php while($course = mysqli\_fetch\_assoc($courses)): ?>

<li>

<?php echo $course['course\_name']; ?> -

<?php echo $course['course\_code']; ?>

<form method="post" action="course\_enroll.php">

<input type="hidden" name="course\_id" value="

<?php echo $course['course\_id']; ?>">

<input type="submit" value="Enroll">

</form>

</li>

<?php endwhile; ?>

A new table tblcourses that contains the couse data with attributes of course\_name and course\_code. In the line of code $courses = mysqli\_query($con, "SELECT \* FROM `tblcourses`"); , the $courses variable holds the result set from querying the database to SELECT all courses FROM the tblcourses table. This returns all courses in the catalogue. Now that we have a list of all the courses, we can use a while loop <?php while($course = mysqli\_fetch\_assoc($courses)): ?> to iterate through each record in the $courses result set (Agarwal, 2023). The mysqli\_fetch\_assoc($courses) function fetches one row from the $courses set as an associative array, assigning it to the $course variable (Ray, 2023). This then echoes, or prints, the course\_name and course\_code in a form with an Enroll button before looping back to iterate the next record, or course\_name and course\_code, in the set (Agarwal, 2023).

Let’s examine the form closer:

<form method="post" action="course\_enroll.php">

<input type="hidden" name="course\_id" value="

<?php echo $course['course\_id']; ?>">

<input type="submit" value="Enroll">

The form uses the POST method to send data when submitted. ***Figure 4***

*PHPMyAdmin view of tblcourses* shows a list of the 25 courses in tblcourses with course\_id being the primary key. Although the course\_id field is hidden, it is still captured in order to be passed along when querying the database to enroll in a course:

if(isset($\_POST['course\_id'])) {

$course\_id = $\_POST['course\_id'];

$email = $\_SESSION['email'];

$checkEnrollment = mysqli\_query($con, "SELECT \* FROM `tblenrollment` WHERE email='$email' AND course\_id='$course\_id'");

if(mysqli\_num\_rows($checkEnrollment) == 0) {

mysqli\_query($con, "INSERT INTO `tblenrollment` (email, course\_id) VALUES ('$email', '$course\_id')");

}

}

When the user clicks Enroll in the form for a course, the code checks whether a value for course\_id has been submit using POST. If so, the $course\_id and $email, as dictated by the e-mail of the user logged in from the session token, are captured and used to check if the student is already enrolled in the course. If the check returns a value of 0 (zero), a query is performed to INSERT INTO the new tblenrollment and add a row with the attribute values of the student’s email and course\_id that the user clicked Enroll for. This code provides the student enrollment catelog mechanism where a full list of available courses is displayed, the user is able to click Enroll for a course, a check is made to confirm the user is not already enrolled in the course and, if they are not already enrolled, the enrollment data table is updated to reflect they are now enrolled in the course.

Next, we’ll examine the construction of my\_courses.php, **Figure 7**

*my\_courses.php* shows an SQL JOIN query fetches all of the courses the user is enrolled in by joining the tblenrollment and tblcourses tables based on their course\_id, then stores the result set in the $enrollments variable (Koidan, 2021):

$enrollments = mysqli\_query($con, "SELECT \* FROM `tblenrollment` JOIN `tblcourses` ON tblenrollment.course\_id = tblcourses.course\_id WHERE email='$email'");

Next, another while loop is used to iterate through each record of this time the $enrollments result set. The mysqli\_fetch\_assoc($courses) function fetches one row from the $enrollments set as an associative array, assigning it to the $course variable (Ray, 2023). This process is almost identical to the course\_enroll.php source code in that it uses the same form while iterating through each row of the array to display each course the student is enrolled in, and a submit button to Unenroll. When a user clicks the Unenroll button, and after checking if a POST has been initiated, the code instantly jumps to querying the database to DELETE the course\_id for the email of the user session FROM the tblenrollment table. Unlike class\_enroll.php which checks if the user is already enrolled before querying the database to INSERT INTO and add the enrollment, it is not necessary in the my\_courses.php unenroll process because the check if the user is currently enrolled was already performed in the $enrollments query in order to be displayed in the student’s currently registered courses, therefore checking again would be redundant. In **Figure 5** *PHPMyAdmin view of tblenrollment* we can see the user (email) primary key enrollment\_id in the rows with values 28 and 29 show user (email) roger@123.com has the course\_id attribute value of 22, then in **Figure 4** *PHPMyAdmin view of tblcourses* we can see the primary key course\_id values 22 corresponds to the course\_code and course\_name of GEN103 and Information Literacy. This is how both tables can be joined so when the user roger@123.com navigates to the my\_courses.php page, the GEN103 Information Literacy course is displayed with the option to Unenroll.

**Summary of Experience**

My experience implementing these two new files and the enroll/unenroll functionality was simple because I did not have to modify any of the other pages, with the exception of the master page to add the “My Courses” and “Enroll in Courses” hyperlinks in the navigation bar. I also had to create tblenrollment and tblcourses tables in the MySQL database. From there, it was a simple matter of creating the course\_enroll.php and my\_courses.php files inside the htdocs folder of XAMPP. One minor adjustment was also changing the link from my prior login profile page that displayed “Login Successful” to the new my\_courses.php page. I felt this would create a better user experience rather than bringing the user to a landing page that simply acknowledged their login was successful, and instead brought them to their current course enrollment list.

**References**

Agarwal, Y. (2023). PHP while loop - Scaler topics. *Scaler Topics*. https://www.scaler.com/topics/php-tutorial/while-loop-in-php/

Koidan, K. (2021). *7 SQL JOIN Examples With Detailed Explanations*. LearnSQL. https://learnsql.com/blog/sql-join-examples-with-explanations/

Ray, S. (2023). PHP mysqli\_fetch\_assoc() Function - Scaler Topics. *Scaler Topics*. https://www.scaler.com/topics/mysqli-fetch-assoc/