

Database Development and Class Registration

Anders Berg

University of Arizona Global (Formally Ashford University)

CST 499–Capstone for Computer Software Technology

Amjad Alkilani

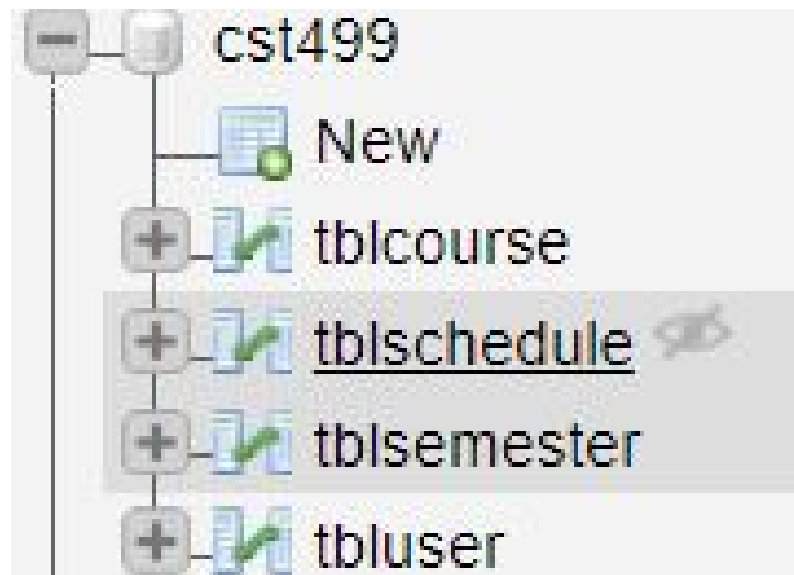
11/11/2021

Database Development and Class Registration

Database Development

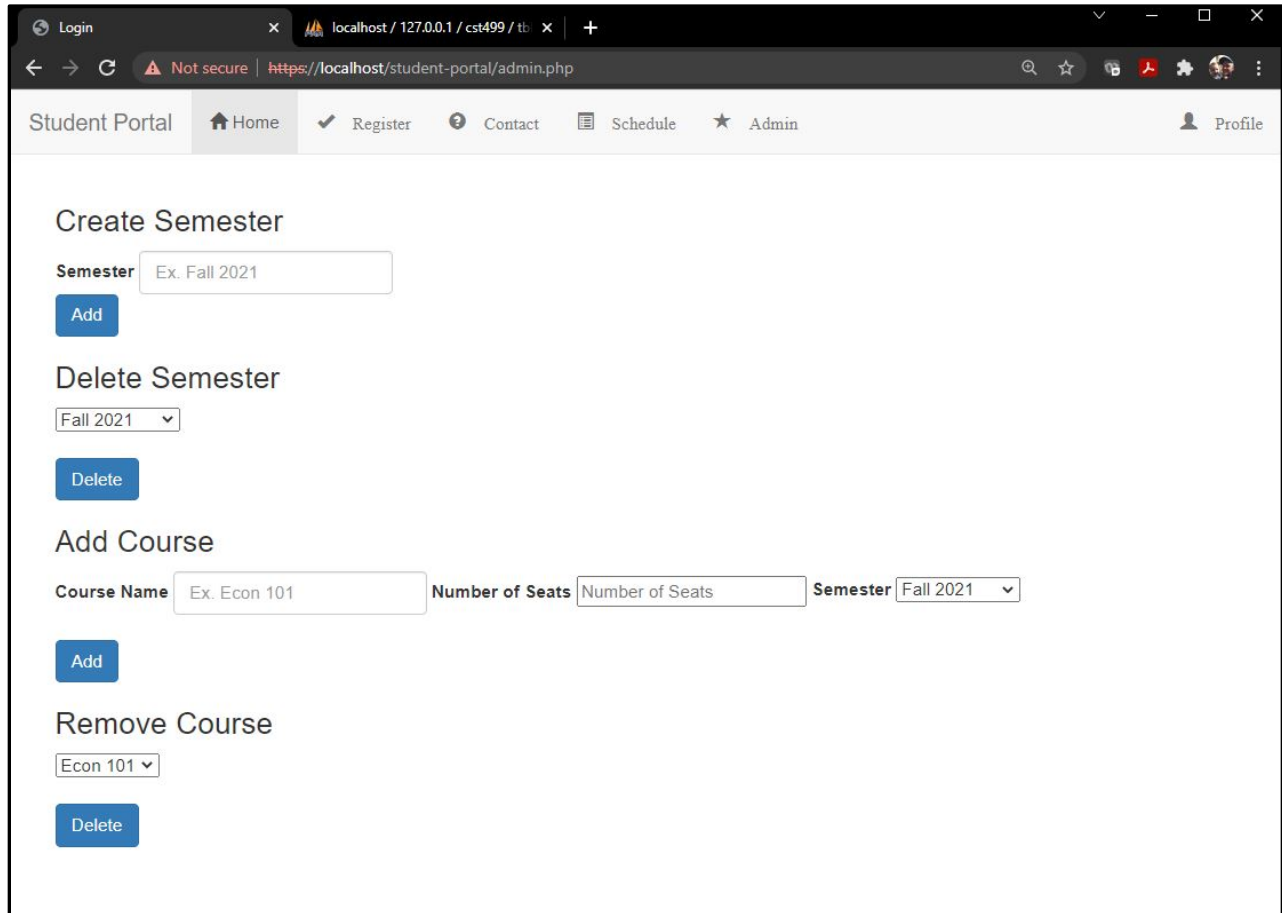
Relational Databases are efficient for storing a collection of data or related entities. The SQL engine of a database is mostly hidden from the user, but with SQL queries, users can join and create data that is needed. One of the biggest advantages of a relational database, is that it can be normalized to reduce the complexities of the data from the user. Users can access tables independently or use SQL queries to join tables together. An example would be the student portal registration system, that uses HTML, bootstrap to present the values from the MySQL database. Using PHP code, the values are returned, altered, and removed. (Coronel, Morris, 2019)

The database for this course registration website is built with four tables: tbluser, tblcourse, tblschedule and tblsemester. Respectively they keep track of user, course, schedule and semester data.



Registration Pages

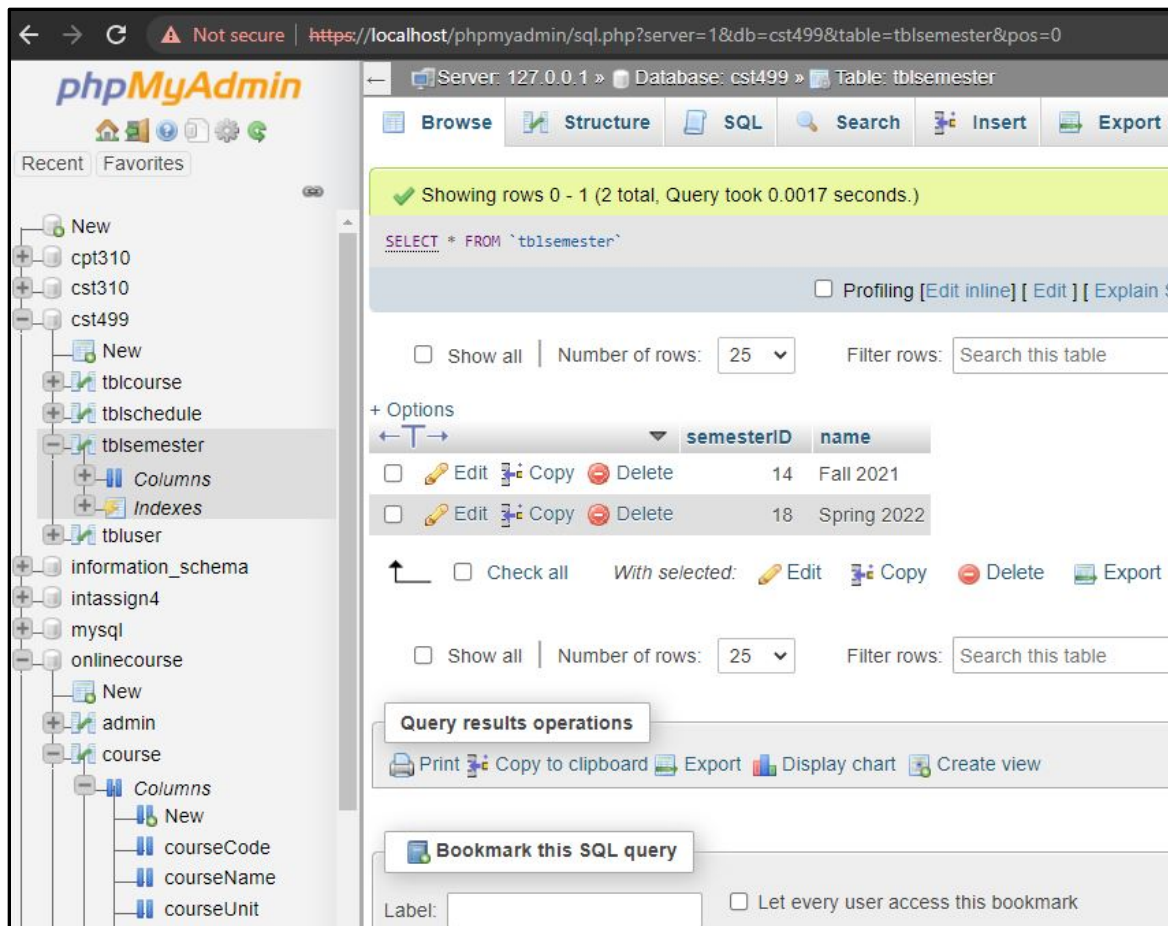
Below is the layout of the admin page, which is a form that collects that allows administrators the ability to create semesters, delete semesters, add courses, and remove courses. This layout has been built using HTML, bootstrap and PHP.



The screenshot shows a web browser window with the URL `https://localhost/student-portal/admin.php`. The page has a navigation bar with links: Student Portal, Home, Register, Contact, Schedule, Admin, and Profile. The main content area contains four sections:

- Create Semester**: A text input field with the placeholder "Ex. Fall 2021" and a blue "Add" button.
- Delete Semester**: A dropdown menu showing "Fall 2021" and a blue "Delete" button.
- Add Course**: Three input fields: "Course Name" (placeholder "Ex. Econ 101"), "Number of Seats" (placeholder "Number of Seats"), and "Semester" (dropdown showing "Fall 2021"). A blue "Add" button is below them.
- Remove Course**: A dropdown menu showing "Econ 101" and a blue "Delete" button.

When the administrator uses this form, data is sent and saved in a mySQL database. Using a session variable, this admin page is only available to administrator when they login. Regular users only have access to the home, register, contact, schedule, and profile tabs.



In a custom class called “connectDB”, `mysqli_connect` is used to create a connection to the database. This class also has a function called `executeQuery` that is used for inserting data into the database. `executeQuery` uses a `connect_error` function to validate the connection to the database.

```

<?php

class connectDB {

    function executeSelectQuery($con,$sql){

        $result = mysqli_query($con, $sql);
        echo "<br>";
        echo "<table border='1'>";
        while ($row = mysqli_fetch_assoc($result)) {
            echo "<tr>";
            foreach ($row as $field => $value) {
                echo "<td>" . $value . "</td>";
            }
            echo "</tr>";
        }
        echo "</table>";
    }

    function executeQuery($con,$sql){

        if ($con->connect_error) {
            die("Connection failed: " . $con->connect_error);
        }

        if ($con->query($sql) === TRUE) {
            echo "New record created successfully";
        } else {
            echo "Error: " . $sql . "<br>" . $con->error;
        }

        $con->close();
    }

}

```

addcourse.php, removecourse.php, addsemester.php and removessemester all use this class with a require statement to execute queries to add and remove information from the database. Below is the addcourse.php that is called as an action from an HTML button to insert course's information into the database.

```

1 <?php
2 require 'connectDB.php';
3
4 $connect1 = new connectDB();
5
6 require 'connectToDB.php';
7
8
9 // Taking all 5 values from the form data(input)
10 $name = $_POST['course'];
11 $noofseats = $_POST['numseats'];
12 $semester = $_POST['semester'];
13
14 $sql = "INSERT INTO tblcourse (name, semesterID, noofSeats)
15 VALUES ('$name','$semester','$noofseats')";
16
17 $connect1->executeQuery($con,$sql);
18
19 header("Location: admin.php?course=added");
20 ?>

```

Below is the code that is used in the form that the administrator interacts with to add a course to the database catalog.

```

<div class="container">
  <h3>Add Course</h3>
  <form class="form-inline" action="addcourse.php" method="post">

    <div class="form-group">
      <label for="course">Course Name</label>
      <input type="text" name="course" class="form-control" id="course" placeholder="Ex. Econ 101">
    </div>

    <div class="form-group">
      <label for="numseats">Number of Seats</label>
      <input type="text" name="numseats" class="numseats" id="numseats" placeholder="Number of Seats">
    </div>

    <div class="form-group">
      <label for="numseats">Semester</label>
      <select name="semester">
        <?php while($row1 = mysqli_fetch_array($semesterResult2)); ?>
        <option value="<?php echo $row1[0];?>"><?php echo $row1[1];?></option>
        <?php endwhile;?>
      </select>
    </div>
    <br><br>
    <button type="submit" class="btn btn-primary">Add</button>

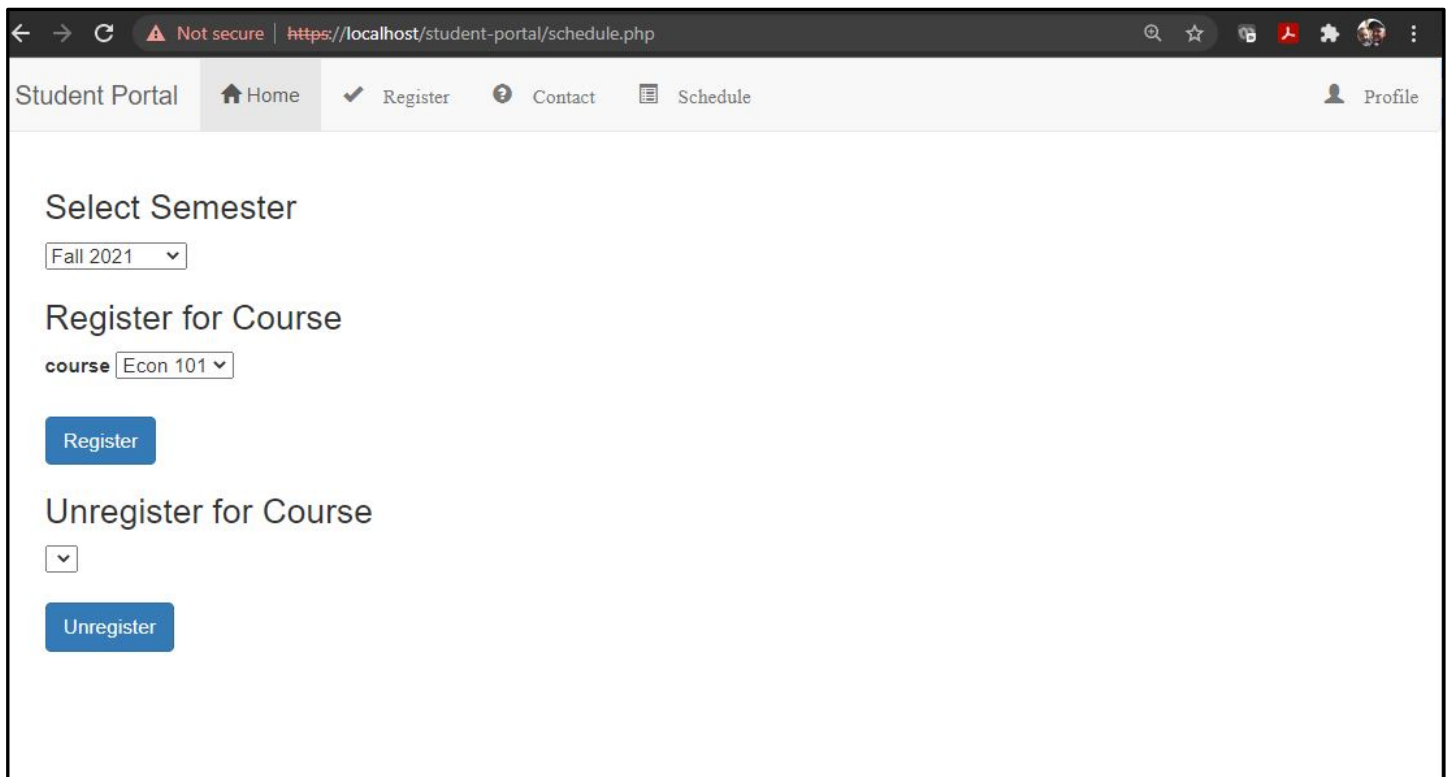
  </form>

</div>

```

Schedule Page

Below is the HTML, bootstrap layout of the schedule registration page. Users use this page to sign up for class based on semester and data is added and removed from the database using PHP. Users select the semester and course that they would like to register for. Once users are register for a course, they can unregister for a course. Also as seen in this image, regular users do not have access to the admin tab.



The screenshot shows a web browser window with the URL `https://localhost/student-portal/schedule.php`. The page has a navigation bar with links: Home, Register, Contact, and Schedule. The main content area is titled "Student Portal" and contains three sections: "Select Semester" with a dropdown menu showing "Fall 2021", "Register for Course" with a "course" dropdown menu showing "Econ 101" and a "Register" button, and "Unregister for Course" with a dropdown menu and an "Unregister" button.

The logic behind this code is similar to that which was used for the admin tab. In the following image php code fetches course and semester values from the database that users can register for. Using the same class that was used for the admin page for connecting to the database, values are added and removed from the database using SQL queries through PHP.

```

<div class="container">
  <h3>Select Semester</h3>
  <form class="form-inline" action="" method="post">

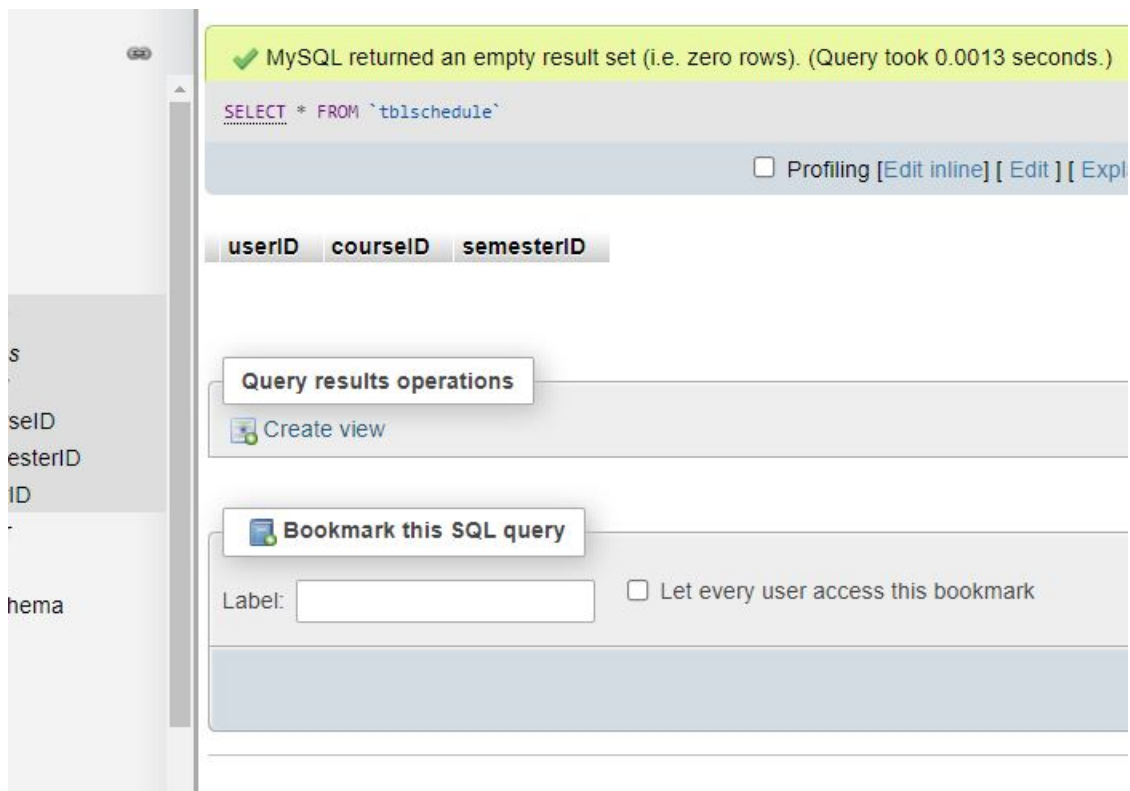
    <div class="form-group">
      <select name="activesemester">
        <?php while($row1 = mysqli_fetch_array($semesterResult)); ?>
        <option value="<?php echo $row1[0];?>"><?php echo $row1[1];?></option>
        <?php
          // $id = $_POST[$row1[0]];

        endwhile;?>
      </select>
    </div>
  </form>

</div>
<div class="container">

```

The following screen shots illustrate the process of registering for course with this application.



Login

localhost / 127.0.0.1 / cst499 / th

Not secure | https://localhost/student-portal/schedule.php?register=success

Student Portal Home Register Contact Schedule Profile

Select Semester

User ID:

Select Semester

Fall 2021

Register for Course

course Econ 101

Register

Unregister for Course

Unregister

Login

localhost / 127.0.0.1 / cst499 / th

Not secure | https://localhost/student-portal/schedule.php?register=success

Student Portal Home Register Contact Schedule Profile

Select Semester

User ID:

Select Semester

Fall 2021

Register for Course

course Econ 101

Register

Unregister for Course

Unregister

The screenshot shows a database management interface. On the left is a tree view of the database structure. The 'tblschedule' table is selected, and its columns are listed: 'New', 'courseID', 'semesterID', and 'userID'. The main panel on the right displays a query result for the 'tblschedule' table. The query is 'SELECT * FROM `tblschedule`'. The result shows 2 rows. The first row has values 2, 1, and 14 for the columns 'userID', 'courseID', and 'semesterID' respectively. The second row has values 2, 5, and 14. The interface also includes a status bar at the top indicating 'Showing rows 0 - 1 (2 total, Query took 0.0018 seconds.)' and a 'Query results operations' button at the bottom.

Showing rows 0 - 1 (2 total, Query took 0.0018 seconds.)

`SELECT * FROM `tblschedule``

☐ Profiling [\[Edit inline\]](#) [\[Edit\]](#) [\[Explain SQL\]](#) [\[Create PHP code\]](#) [\[Refresh\]](#)

☐ Show all | Number of rows: 25 Filter rows:

+ Options

userID	courseID	semesterID
2	1	14
2	5	14

☐ Show all | Number of rows: 25 Filter rows:

Query results operations

In summary, the process of implementing the design of the web application was strait forward. It took me the most time to figure out the syntax in php to alter the database. There are still a few features that I would try like to add, like limiting the number of users that can register for a course and waitlisting them.

References

Coronel, C., & Morris, S. (2019). Database systems: Design, implementation, and management (13th ed.). Retrieved from <https://www.vitalsource.com>

Sommerville, I. (2018). *Software Engineering*. Hallbergmoos/Germany: Pearson.