Login, and Enrollment Pages Development

Anders Berg

University of Arizona Global (Formally Ashford University)

CST 499–Capstone for Computer Software Technology

Amjad Alkilani

11/11/2021

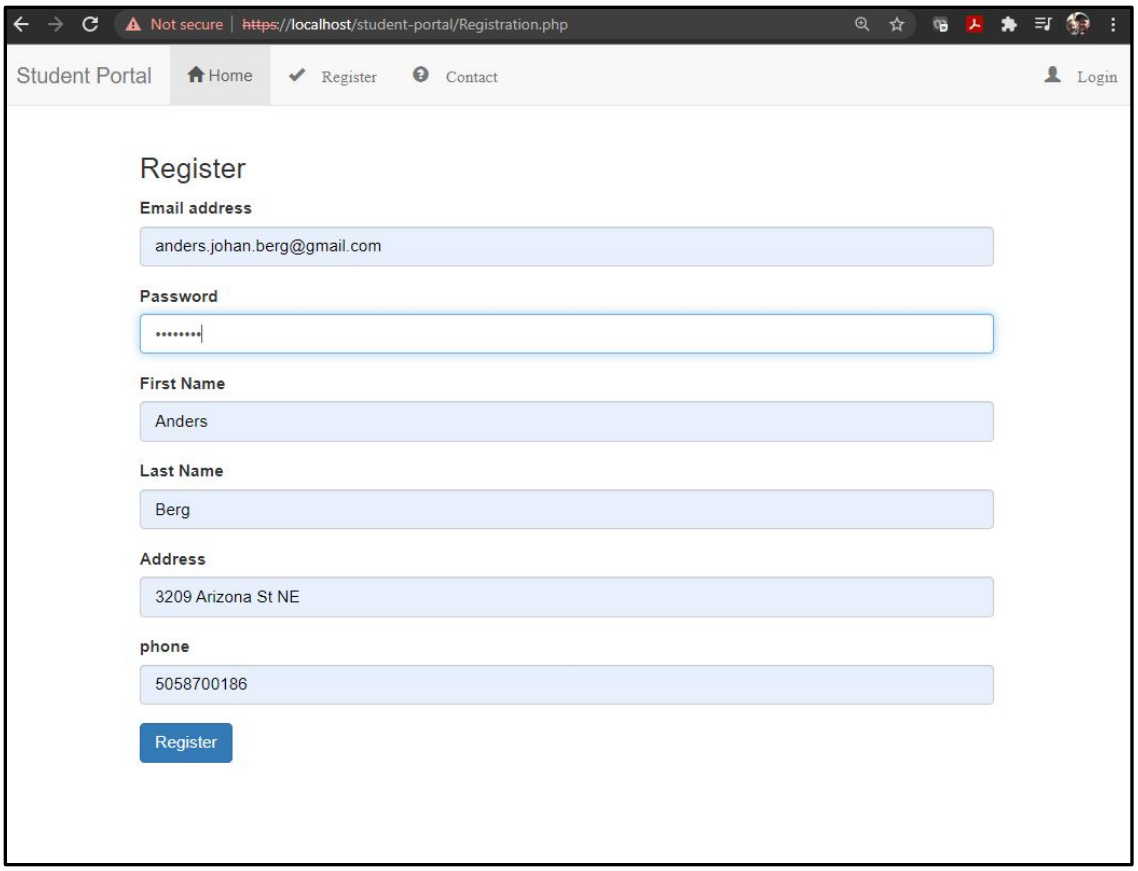
Login, and Enrollment Pages Development

How to run a PHP file in XAMPP

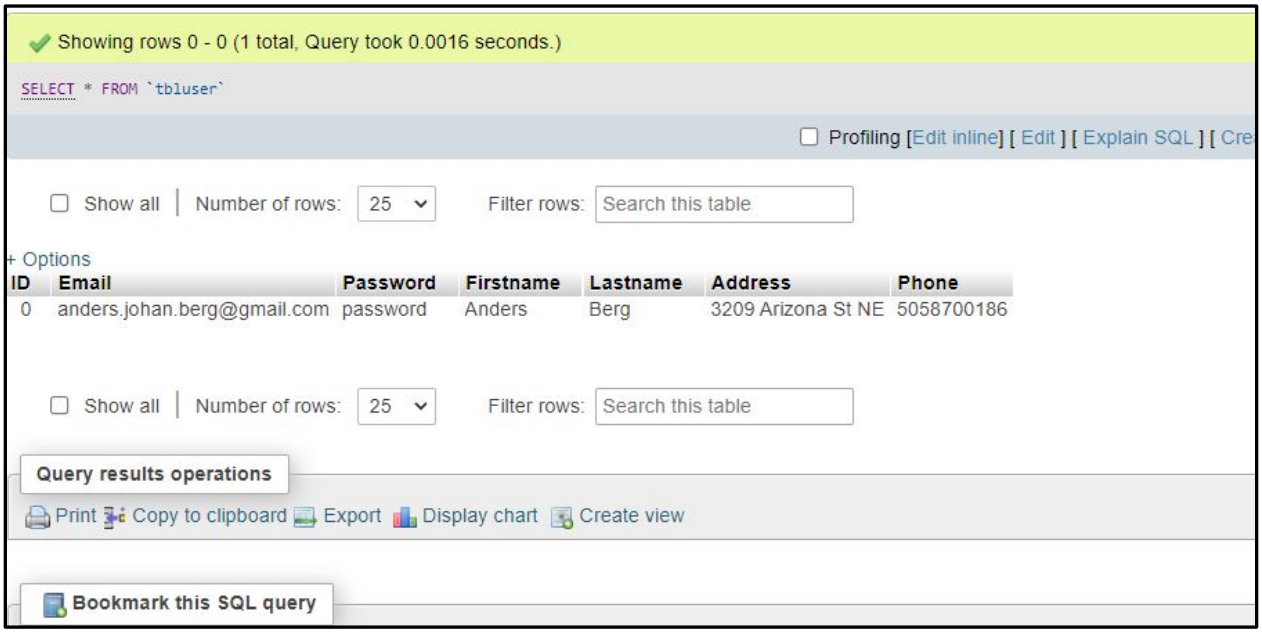
XAMMP is a popular development environment that allows users to a local apache distribution that contains MariaDB, PHP and Perl. XAMMP is a great environment for developers to be able to replicate a server on their local machine and test code. To do so, user must first install XAMPP, which is free and open source web server stack solution. Once installed, a localized server environment can be accessed by starting the XAMPP control panel and starting the Apache module. To run PHP code on XAMPP, all files must be saved to the "xampp\htdocs" folder on the machine. Once they have been saved, the can be run by going to https://localhost/wk1/index.php, or whatever the file name is called. When Navigating to PHP files on the server, the automatically open a display their content. As opposed to HTML files, that appear in a directory that then must be clicked on to view their content. (VMWare, 2021)

Registration Pages

Below is the layout of the registration page, which is a form that collects the email address, password, first name, last name, address and phone number of the user.



When the user registers with this form the data is sent and saved in a mySQL database



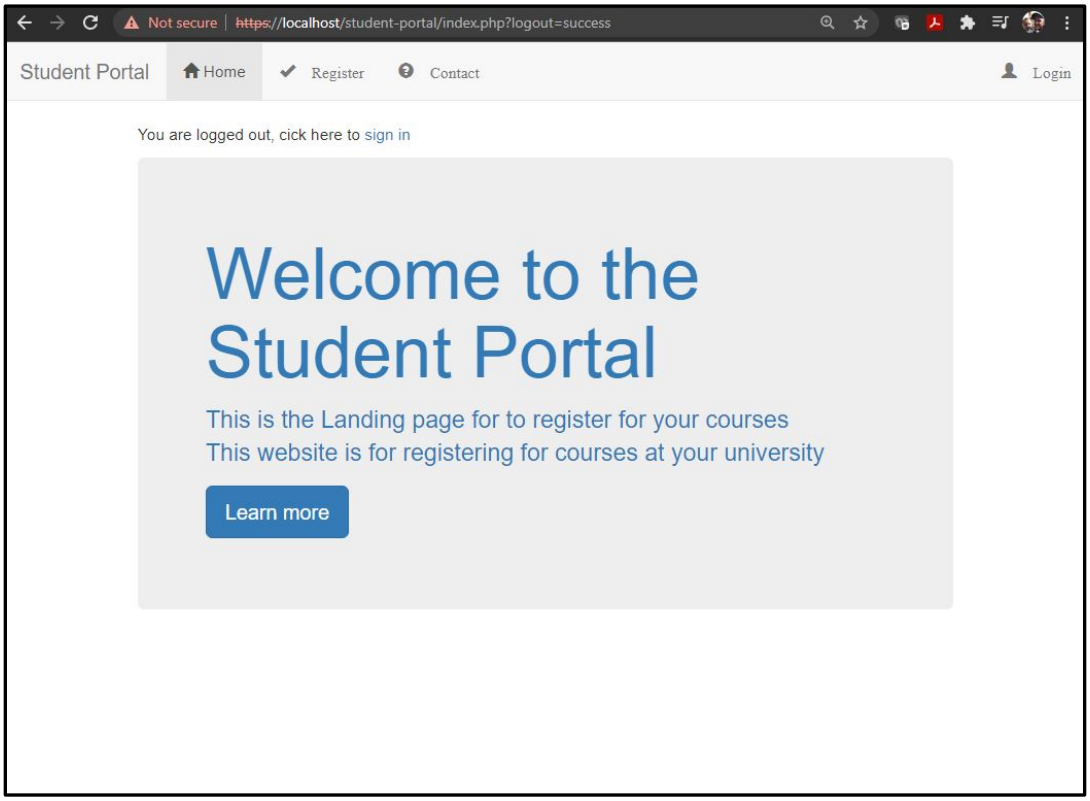
In a custom class called “connectDB”, mysqli_connect is used to create a connection to the database. This class also has a function called executeQuery that is used for inserting data into the database. executeQuery uses a connect_error function to validate the connection to the database.

```php
<?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();


  }
}
```

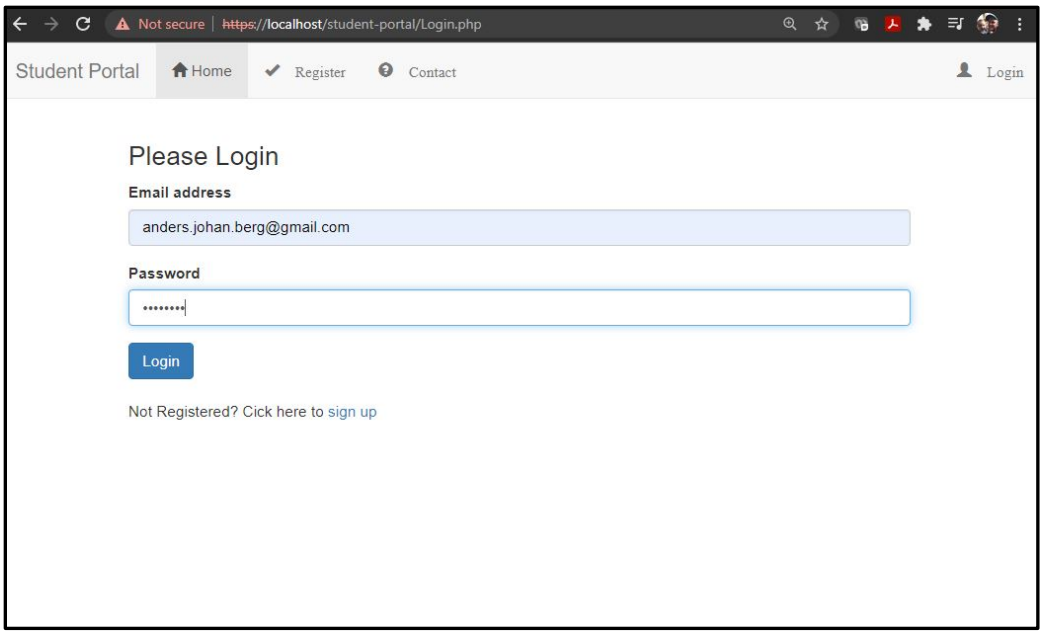
Register.php uses this class with a require statement to collect attributes and post them to the database. These attributes are formatted in a SQL query and inserted into the database using the execute query from the “connectDB” class.

Login Pages

Below are sample screen shots of my website using HTML, bootstrap and PHP. From the main page, users have the option to click the login link the upper left-hand corner or status link at the top of the page that is generated by the session variable. There is a variable that is based on login status that tracks if the user is logged into the website or not.



After the user click one of these two links, the page is redirected to the login page. The login page requests an email and password for the user to login. There is button for the user to submit a login request. There is also a link that will redirect the user if they need to sign up.



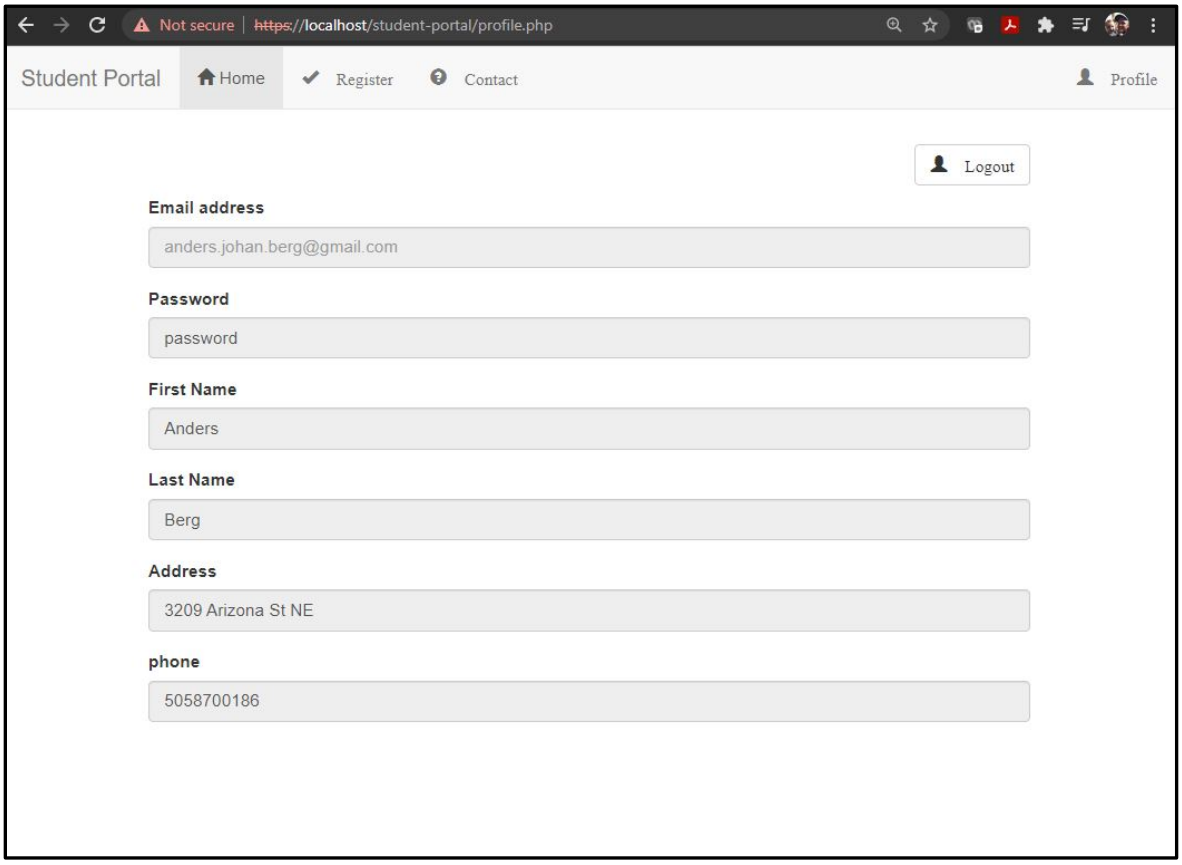
The code behind this login page is a simple HTML form with that is stylized with bootstrap. The button on this page calls out to the access.php file validate the email and password provided by the user. The form on the sign in page calls access.php as the action of the form and using the post method, and request information from the database with the click of the submit button.

C: > xampp > htdocs > employee-portal > access.php >

```php
<?php

if(isset($_POST['login-submit'])){

    require 'connectToDB.php';

    $userid = $_POST['userid'];
    $password = $_POST['pwd'];

    if (empty($userid) || empty($password)) {
        header("Location: index.php?error=emptyinput");
        exit();
    }
    else {
        $sql = "SELECT id, email, password FROM tbluser WHERE email=?";
        $stmt = mysqli_stmt_init($con);
        if (!mysqli_stmt_prepare($stmt,$sql)){
           header("Location: index.php?error=sqlerror");
           exit();
        }
        else{

           mysqli_stmt_bind_param($stmt,"s",$userid);
           mysqli_stmt_execute($stmt);
           $results = mysqli_stmt_get_result($stmt);

           if ($row = mysqli_fetch_assoc($results)){

                if(strcmp($password,$row['password'])!==0){
                    header("Location: index.php?error=wrongpassword");
                    exit();
                }
                else{
                    session_start();
                    $_SESSION['id']=$row['id'];

                    header("Location: index.php?login=success");
                    exit();
                }
           }
           else{
            header("Location: index.php?error=nouser");
            exit();
           }
        }
}
}
else{
    header("Location: index.php?error=failed");
    exit();
}
?>
```

Profile Page

The access.php is designed to perform a query on the database based on the information provided by user. Using the email that the user provided, a query is generated to select the id, email and password records from the database. The password given is then is compared against the value that is recorded in the database. This logic is nested with a few error statements to check if there is an empty input from the user, a sql error, the wrong password or a general error.



Once the user is logged in, they can click on the profile button to access the profile page. This page presents to the user the details of their account that is stored withing the MySQL database. Each profil can view email, password, first name, last name, address and phone number. On this page, the data is presented the user in a form, like the registration page, except at this time

they cannot change the values. This page was designed in a form, that easily allows for the implementation of an update button in the future. On the profile page, there is a logout button that terminates the user session and redirects the user to the home page.

The code behind the profile page, is a php file that selects data from the database and populates it into an HTML form. The HTML form is stylized with bootstrap. The profile.php page performs a query on the database with the session id variable that is created when the user logs in. Using the user's unique id, a query is generated to select all the attributes for that users from the database. After these records have been fetched, they are echoed with in each form box in the HMTL code. The HMTL code uses a "readonly" tag, so that for now, the user can only look at this information and not edit it.

```
<body>
<?php
$currentUser = $_SESSION['id'];
$sqlUser = "SELECT * FROM `tbluser` WHERE `id`=$currentUser";
$results = mysqli_query ($con,$sqlUser);

if ($results){
    if(mysqli_num_rows($results)>0){
        while( $row = mysqli_fetch_array($results)){
            //print_r ($row['firstName']);
            ?>

            <div class="container">
            <form action ="logout.php" method="post">
            <button style="float: right;" type="submit" name="logout-submit" class="btn btn-default navbar-btn"><span class= "glyphicon glyphicon-user"></span> Logout</but
            </form>
            </div>
            <div class="container">
            <div class="form-group">
              <label for="exampleInputEmail1">Email address</label>
              <input type="email" name="email" class="form-control" id="exampleInputEmail1" placeholder = "<?php echo $row['email'];?>" readonly>
            </div>
            <div class="form-group">
              <label for="exampleInputPassword1">Password</label>
              <input type = "text" name="pw" class="form-control" id="exampleInputPassword1" value = "<?php echo $row['password'];?>" readonly>
            </div>
            <div class="form-group">
              <label for="firstname">First Name</label>
              <input type="firstname" name="first_name"class="form-control" id="firstname" value = "<?php echo $row['firstName'];?>" readonly>
            </div>
            <div class="form-group">
              <label for="lastname">Last Name</label>
              <input type="lastname" name="last_name" class="form-control" id="lastname" value = "<?php echo $row['lastName'];?>" readonly>
            </div>
            <div class="form-group">
              <label for="address">Address</label>
              <input type="address"  name="address" class="form-control" id="address" value = "<?php echo $row['address'];?>" readonly>
            </div>
            <div class="form-group">
              <label for="phone">phone</label>
              <input type="phone" name="phone" class="form-control" id="phone" value = "<?php echo $row['phone'];?>" readonly>
            </div>
            <div class="form-group">
              <label for="salary">salary</label>
              <input type="salary" name="salary" class="form-control" id="salary" value = "<?php echo $row['salary'];?>" readonly>
            <div class="form-group">
```

References

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

VMWare. (2021). *Apache Friends*. Apache Friends RSS. Retrieved November 1, 2021, from https://www.apachefriends.org/index.html.