**Landing, Login, and Enrollment Pages Development**

Tony Carolla

The University of Arizona Global Campus

CST 499 Capstone for Computer Software Technology

Amr Elchouemi

January 23, 2023

**Landing, Login, and Enrollment Pages Development**

I completed the first release of the MetricuL8 product, including the landing, login, and new student registration pages. I brought several technologies together to deliver the required functionality to a web browser. Deploying modern browser-based solutions requires the use of several computer languages, the configuration of several server types, and the understanding required to coordinate the flow of data needed by the application.

**PHP and XAMPP**

The application requires functionality on both the client and the server, so I wrote the core functionality of the application using the PHP language, and it runs on a web server with a PHP interpreter. I used a PC server and installed the XAMPP package (downloaded from [www.apachefriends.org](http://www.apachefriends.org)), which includes the Apache web server and a PHP interpreter (Kumari & Nandal, 2017). I installed it on the C drive of the server with the default web directory in Apache set to \xampp\htdocs. I created a folder named 499 in the \xampp\htdocs folder, and I placed all PHP files in the \xampp\htdocs\499 folder. When a user visits the application, their browser requests the application folder. The Apache web server responds by looking through the application folder. It first looks for a file named “index.php,” which is the application landing page. The Apache configuration file (httpd.conf) contains the files that the web server looks for when a client requests a folder name and does not specify a file. When the Apache web server finds the PHP file, it hands the file to the PHP interpreter. The PHP interpreter executes the PHP file and returns the result to the Apache web server, which returns the result to the client (Tsui et al., 2018).

**Application Pages**

Please find examples of the Landing, Login, and Registration pages below in figures 1, 2, and 3, respectively.

**Figure 1**

*Landing Page*

Graphical user interface, website

Description automatically generated

**Figure 2**

*Login Page*

Graphical user interface, website

Description automatically generated

**Figure 3**

*Registration Page*

Graphical user interface, website

Description automatically generated

**Database Layout**

The application uses the MySQL database management system (DBMS). I created the script below to generate the database layout in figure 4.

**Figure 4**

*Database Layout and Script*

Graphical user interface, text

Description automatically generated

**Custom Connection Class**

The MetricuL8 application uses a combination of languages to deliver functionality. For example, the web pages are written in PHP and use Structured Query Language (SQL) to access the database. Both the login and registration pages use the database in this first release of MetricuL8, and there are plans for other pages that will use the database. Rather than repeat the core database code on every page, I created a custom connection class with three database functions in the MetricuL8Store PHP file in figure 5.

**Figure 5**

*Custom Connection Class MetricuL8Store.php*

<?php

class MetricuL8Store {

function executeSelectQuery($sql, $con="mysql:host=localhost; dbname=metricul8; user=EmeraldV; password=goraider$") {

try {

$pdo = new PDO($con);

$pdo->setAttribute(PDO::ATTR\_EMULATE\_PREPARES, false);

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

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

$resultSet = array();

while($row = $result->fetch()) {

$resultSet[] = $row;

}

$pdo = null;

return $resultSet;

}

catch (PDOException $e) {

die( $e->getMessage() );

}

}

function executeQuery($sql, $con="mysql:host=localhost;dbname=metricul8;user=EmeraldV;password=goraider$") {

try {

$pdo = new PDO($con);

$pdo->setAttribute(PDO::ATTR\_EMULATE\_PREPARES, false);

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

$count = $pdo->exec($sql);

return $count;

}

catch (PDOException $e) {

die( $e->getMessage() );

}

}

function checkCred($usernameTry, $passKeyTry) {

try {

$con='mysql:host=localhost;dbname=metricul8;user=EmeraldV;password=goraider$';

$pdo = new PDO($con);

$pdo->setAttribute(PDO::ATTR\_EMULATE\_PREPARES, false);

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

$stmt = $pdo->prepare('SELECT COUNT(\*) AS stdcount FROM student WHERE email = :usernametry AND password= :passkeytry');

$stmt->execute([ 'usernametry' => $usernameTry, 'passkeytry' => $passKeyTry ]);

foreach($stmt as $row) {

return ($row['stdcount'] == 1);

}

}

catch (PDOException $e) {

die( $e->getMessage() );

}

}

}

?>

The first function executeSelectQuery allows the execution of an SQL query. It accepts two parameters: one for the SQL statement and one for the connection string. It uses a default connection string that points to the MetricuL8 database. It creates a connection using the connection string and a PHP Data Objects (PDO) instance. A PDO allows the PHP interpreter to send commands to a database server and receive the results back (Arnautović & Bundalo, 2013). It is defined with a connection string that indicates the address of the database server, the database name, and the credentials. Once it is created, the function uses the PDO to execute the query from the SQL statement parameter. To get the list of results, the function creates an array *$resultSet*, iterates over the results from the PDO execution, and stores each row in the *$resultSet* array. It closes the connection by disposing of the variable for the PDO and returns the result. If an error occurs during the execution, the error is returned, and the process is stopped.

The executeQuery function allows the execution of an SQL query that returns the result rather than a list of data. It is the same as the executeSelectQuery except that it uses the exec PDO function instead of the query function. Exec does not return data rows, it only returns the number of rows affected.

The checkCred function accepts a username and password. It connects to the MetricuL8 database and runs a query to see if the user and password matches one in the database. It uses the COUNT() function of the database so that a zero is returned if the username and password do not match, and a 1 if a row is found that does match. Either way, a value is returned, and there is never a NULL returned.

**Registration Page Functionality**

The Registration page in figure 6 below allows students to register by adding their information to the MetricuL8 database.

**Figure 6**

*Registration Page*

<?php

error\_reporting(E\_ALL ^ E\_NOTICE);

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title> Registration Page </title>

<meta charset="utf8" />

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

<link rel="stylesheet" href="css/normalize.css">

<link rel="stylesheet" href="css/main.css">

<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 class="bgBody">

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

<?php

if ($\_SERVER["REQUEST\_METHOD"]=="POST") {

$anyErr = false;

$errorsExist = "";

$emailError = "";

$passKeyError = "";

$lastNameError = "";

$firstNameError = "";

$phoneError = "";

$dobError = "";

if (empty($\_POST["emailAddr"])) {

$anyErr = true;

$emailError = "Email required";

} else {

$emailAddr = $\_POST["emailAddr"];

}

if (empty($\_POST["passKey"])) {

$anyErr = true;

$passKeyError = "Password required";

} else {

$passKey = $\_POST["passKey"];

}

if (empty($\_POST["firstName"])) {

$anyErr = true;

$firstNameError = "First name required";

} else {

$firstName = $\_POST["firstName"];

}

if (empty($\_POST["lastName"])) {

$anyErr = true;

$lastNameError = "Last name required";

} else {

$lastName = $\_POST["lastName"];

}

if (empty($\_POST["birthDate"])) {

$anyErr = true;

$dobError = "Birth Date required";

} else {

$birthDate = $\_POST["birthDate"];

}

$phoneArea = $\_POST["phoneArea"];

$phoneNum = $\_POST["phoneNum"];

if ( empty($\_POST["phoneArea"]) || empty($\_POST["phoneNum"]) ) {

$anyErr = true;

$phoneError = "Phone# required";

}

if ($anyErr)

{

$errorsExist = "Fix the errors below";

} else {

// save to the database

$dbStore = new MetricuL8Store();

// get the max ID from the database

$sql = 'select max(id)+1 AS maxid from metricul8.student';

if($qResult = $dbStore->executeSelectQuery($sql)){

foreach ($qResult as $row){

$newId = $row['maxid'];

}

} else {

$newId = 1;

}

// build the insert statement

$sql = "insert into metricul8.student(id, email, password, firstName, "

. "lastName, phoneArea, phoneNumber, birthDate) values("

. strval($newId)

. ",'" . $\_POST["emailAddr"] . "'"

. ",'" . $\_POST["passKey"] . "'"

. ",'" . $\_POST["firstName"] . "'"

. ",'" . $\_POST["lastName"] . "'"

. "," . strval($\_POST["phoneArea"])

. "," . strval($\_POST["phoneNum"])

. ",'" . strval($\_POST["birthDate"]) . "'"

. ")";

if ($dbStore->executeQuery($sql)) {

$errorsExist = "Student Added!";

$emailAddr = "";

$passKey = "";

$lastName = "";

$firstName = "";

$phoneArea = 0;

$phoneNum = 0;

$birthDate = "";

$emailError = "";

$passKeyError = "";

$lastNameError = "";

$firstNameError = "";

$phoneError = "";

$dobError = "";

} else{

$errorsExist = "Can't save to the database. Contact your system administrator.";

}

}

}

?>

<div class="container">

<h1>New Student Registration</h1>

<p class="error"><?php echo $errorsExist;?></p>

<form class="form-horizontal" role="form" method="post" action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]);?>">

<div class="form-group">

<label class="control-label col-sm-2" for="f-email">Email Address</label>

<div class="col-sm-10">

<input type="email" name="emailAddr" id="f-email" placeholder="email@provider.com" value="<?php echo $emailAddr;?>" />

<span class="error"><?php echo $emailError; ?></span>

</div>

</div>

<div class="form-group">

<label class="control-label col-sm-2" for="f-password">Password</label>

<div class="col-sm-10">

<input type="password" name="passKey" id="f-password" placeholder="8 or more characters" value="<?php echo $passKey;?>" />

<span class="error"><?php echo $passKeyError; ?></span>

</div>

</div>

<div class="form-group">

<label class="control-label col-sm-2" for="f-firstName">First Name</label>

<div class="col-sm-10">

<input type="text" name="firstName" id="f-firstName" value="<?php echo $firstName;?>" />

<span class="error"><?php echo $firstNameError; ?></span>

</div>

</div>

<div class="form-group">

<label class="control-label col-sm-2" for="f-lastName">Last Name</label>

<div class="col-sm-10">

<input type="text" name="lastName" id="f-lastName" value="<?php echo $lastName;?>" />

<span class="error"><?php echo $lastNameError; ?></span>

</div>

</div>

<div class="form-group">

<label class="control-label col-sm-2" for="f-birthDate">Date of Birth</label>

<div class="col-sm-10">

<input type="date" name="birthDate" id="f-birthDate" value="<?php echo $birthDate;?>" />

<span class="error"><?php echo $dobError; ?></span>

</div>

</div>

<div class="row show-grid FloatInput">

<div class="form=group">

<label class="control-label col-sm-2" for="f-phoneArea">Area Code/Phone#</label>

<div class="col-sm-1">

<input type="number" name="phoneArea" class="PhoneMedium" id="f-phoneArea" placeholder="###" value="<?php echo $phoneArea;?>" />

</div>

<div class="col-sm-2">

<input type="number" name="phoneNum" class="PhoneLarge" placeholder="#######" value="<?php echo $phoneNum;?>" />

</div>

<div class="col-sm-3"><span class="error"><?php echo $phoneError; ?></span></div>

</div>

</div>

<div class="form=group">

<div class="col-sm-5 col-sm-offset-1">

<button type="submit" title="Save Registration" class="ButtonFont">

<span class="glyphicon glyphicon-cloud-upload"></span>

</button>

</div>

<div class="col-sm-5">

<button type="reset" title="Reset Form Fields" class="ButtonFont">

<span class="glyphicon glyphicon-refresh"></span>

</button>

</div>

</div>

<div class="col-sm-5 col-sm-offset-1">

<p class="ButtonText">Save</p>

</div>

<div class="col-sm-5">

<p class="ButtonText">Reset</p>

</div>

</form>

</div>

<?php require\_once 'footer.php';?>

</body>

</html>

The client browser issues a GET to the web server when the registration page is requested. The PHP interpreter looks at the request, and since it is a GET instead of a POST, it skips most of the PHP code due to the “if” statement and paints a blank registration form. Once the user fills the form out and presses the “save” button, their browser issues a POST. The PHP interpreter sees this and executes the PHP code due to the “if” statement. The PHP code first resets all error messages in case this is a subsequent submission of the form. Next, it checks each field to ensure that they are all filled out. If any are not, it populates the corresponding error messages. If there are no errors, it saves the new registration to the database. It uses the custom connection classes to do this. First, it runs a query to get the next available student ID from the database using the executeSelectQuery() function. It uses the new ID along with the form data to save the registration using the executeQuery() function. It then clears the data from the screen and redisplays it to the user with the message that a student was added.

**Conclusion**

Deploying modern browser-based solutions requires the use of several computer languages, the configuration of several server types, and the understanding required to coordinate the flow of data needed by the application. All source code is below, including the Cascading Style Sheet (CSS) in figure 7, the master PHP page in figure 8, the landing page in figure 9, the footer page in figure 10, and the login page in figure 11.

**Figure 7**

*Main.CSS*

#jt {

font-family: "Franklin Gothic Medium", serif;

background-image: url("../img/OceanGrid.png");

background-repeat: repeat;

color: #0C2E7E;

}

#epNavBar {

font-family: "Arial Narrow", sans-serif;

}

#homeIcon {

font-size: 64px;

}

.AddressLong {

width: 40%;

}

.bgBody {

background-color: #e1fafd;

}

.ButtonFont {

font-size: 32px;

}

.ButtonText {

font-weight: bold;

}

.error {

color: red;

}

.FloatInput {

margin-bottom: 15px;

}

.ItemName {

font-weight: bold;

}

.ListLabel {

font-style: italic;

color: #0C2E7E;

}

.StateShort {

width: 60%;

}

.ZipMedium {

width: 60%;

}

.PhoneLarge {

width: 80%;

}

.PhoneMedium {

width: 90%;

}

**Figure 8**

*Master.PHP*

<?php

error\_reporting(E\_ALL ^ E\_NOTICE);

if (session\_id() == "") {

ini\_set('session.use\_only\_cookies','1');

session\_start();

}

if(isset($\_SESSION['username']))

echo "Welcome: " . $\_SESSION['username'];

require\_once 'MetricuL8Store.php';

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

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

<div class="jumbotron" id="jt">

<div class="containter text-center">

<h1>MetricuL8</h1>

<h2>Student Enrollment System</h2>

</div>

</div>

<nav class="navbar navbar-inverse">

<div class="containter-fluid">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#epNavBar">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

</div>

<div class="collapse navbar-collapse" id="epNavBar">

<ul class="nav navbar-nav">

<li class="active"><a href="index.php"><span class="glyphicon glyphicon-home"></span> Home</a></li>

</ul>

<ul class="nav navbar-nav navbar-right">

<?php

if (session\_id() == "")

{

ini\_set('session.use\_only\_cookies','1');

session\_start();

}

if(isset($\_SESSION['username']))

{

echo "<li><a href=\"index.php?Logout=1\"><span class=\"glyphicon glyphicon-off\"></span> Logout</a></li>";

}

else

{

echo "<li><a href=\"login.php\"><span class=\"glyphicon glyphicon-user\"></span> Login</a></li>";

echo "<li><a href=\"registration.php\"><span class=\"glyphicon glyphicon-pencil\"></span> Registration</a></li>";

}

?>

</ul>

</div>

</div>

</nav>

</body>

</html>

**Figure 9**

*Index.PHP*

<?php

error\_reporting(E\_ALL ^ E\_NOTICE);

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title> Home Page </title>

<meta charset="utf8" />

<meta name="description" content="" />

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

<link rel="stylesheet" href="css/main.css">

<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 class="bgBody">

<?php

if (session\_id() == "") {

ini\_set('session.use\_only\_cookies','1');

session\_start();

}

if(isset($\_GET['Logout']) && $\_GET['Logout']=='1') {

unset($\_SESSION['username']);

}

?>

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

<div class="container">

<div class="row">

<div class="col-md-3" id="homeIcon">

<span class="glyphicon glyphicon-user"></span>

</div>

<div class="col-md-9">

<h2>Welcome to the MetricuL8 Student Enrollment System!</h2>

<p>Use the links above to access your enrollment records. </p>

</div>

</div>

</div>

<?php require\_once 'footer.php';?>

</body>

</html>

**Figure 10**

*Footer.PHP*

<?php

error\_reporting(E\_ALL ^ E\_NOTICE)

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf8" />

<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>

<div class="navbar-fixed-bottom row-fluid">

<div class="navbar-inner">

<div class="container text-center">

Copyright @ 2023

</div>

</div>

</div>

</body>

</html>

**Figure 11**

*Login.PHP*

<?php

error\_reporting(E\_ALL ^ E\_NOTICE);

require\_once 'MetricuL8Store.php';

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title> Login Page </title>

<meta charset="utf8" />

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

<link rel="stylesheet" href="css/main.css">

<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 class="bgBody">

<?php

if (session\_id() == "") {

ini\_set('session.use\_only\_cookies','1');

session\_start();

}

if( !isset($\_SESSION['username']) && $\_SERVER['REQUEST\_METHOD']=='POST' ) {

$dbStore = new MetricuL8Store();

if( $dbStore->checkCred($\_POST['usernameTry'], $\_POST['passKey']) ) {

$\_SESSION['username'] = $\_POST['usernameTry'];

if(isset($\_SESSION['upError'])){

unset($\_SESSION['upError']);

}

} else {

$\_SESSION['upError'] = true;

}

}

?>

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

<?php

if (session\_id() == "") {

ini\_set('session.use\_only\_cookies','1');

session\_start();

}

if (isset($\_SESSION['username'])) {

echo makeWelcomeForm($\_SESSION['username']);

} else {

echo makeLoginForm();

}

function makeWelcomeForm($uName){

$welcome = '<div class="container"><p>Welcome '. $uName .'</p></div>';

return $welcome;

}

function makeLoginForm() {

if(isset($\_POST['usernameTry'])) {

$usernameTry = $\_POST['usernameTry'];

$passKey = $\_POST['passKey'];

} else{

$usernameTry='';

$passKey='';

}

$html = '<div class="container">';

$html .= '<h1>Log in to MetricuL8</h1>';

if (isset($\_SESSION["upError"])) {

$html .= '<span class="error">Error unable to log in.</span>';

unset($\_SESSION['upError']);

}

$html .='<form class="form-horizontal" role="form" method="post" action="'

. htmlspecialchars($\_SERVER["PHP\_SELF"]) .'">';

$html .= '<div class="form-group">'

.'<label class="control-label col-sm-2" for="f-usernameTry">User Email</label>'

.'<div class="col-sm-10">'

.'<input type="email" name="usernameTry" id="f-usernameTry" placeholder="email@provider.com" value="'

.$usernameTry

.'" /></div></div>';

$html .= '<div class="form-group">'

.'<label class="control-label col-sm-2" for="f-password">Password</label>'

.'<div class="col-sm-10">'

.'<input type="password" name="passKey" id="f-password" value="'.$passKey.'" />'

.'</div></div>';

$html .= '<div class="form=group">'

.'<div class="col-sm-5 col-sm-offset-4">'

.'<button type="submit" title="Login" class="ButtonFont">'

.'<span class="glyphicon glyphicon-ok"></span>'

.'</button></div></div>'

.'</form></div>';

return $html;

}

?>

<?php require\_once 'footer.php';?>

</body>

</html>

**References**

Arnautović, M., & Bundalo, Z. (2013). Comparison of Programming Languages ASP (Commercial) and PHP (Open Source) while Designing Small/Typical Websites. TEM Journal, 2(3), 261–264.

Kumari, P., & Nandal, R. (2017). A Research Paper OnWebsite Development Optimization Using Xampp/PHP. International Journal of Advanced Research in Computer Science, 8(5), 1231–1235.

Tsui, F., Karam, O., & Bernal, B. (2018). Essentials of software engineering (4th ed.). Jones & Bartlett Learning.