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

Austin Blair

The University of Arizona Global Campus

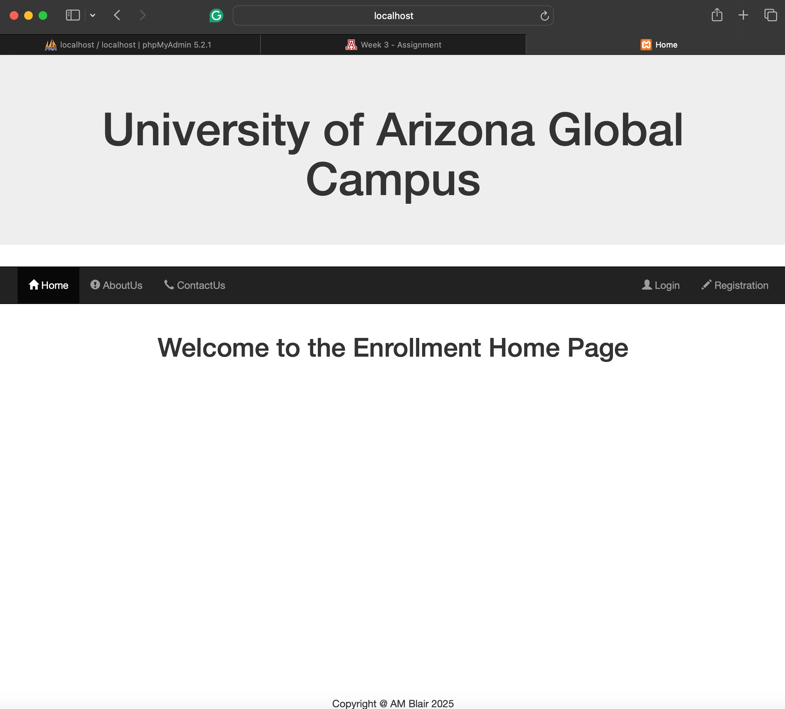
CST499: Capstone for Computer Software Technology

Joseph Rangitsch

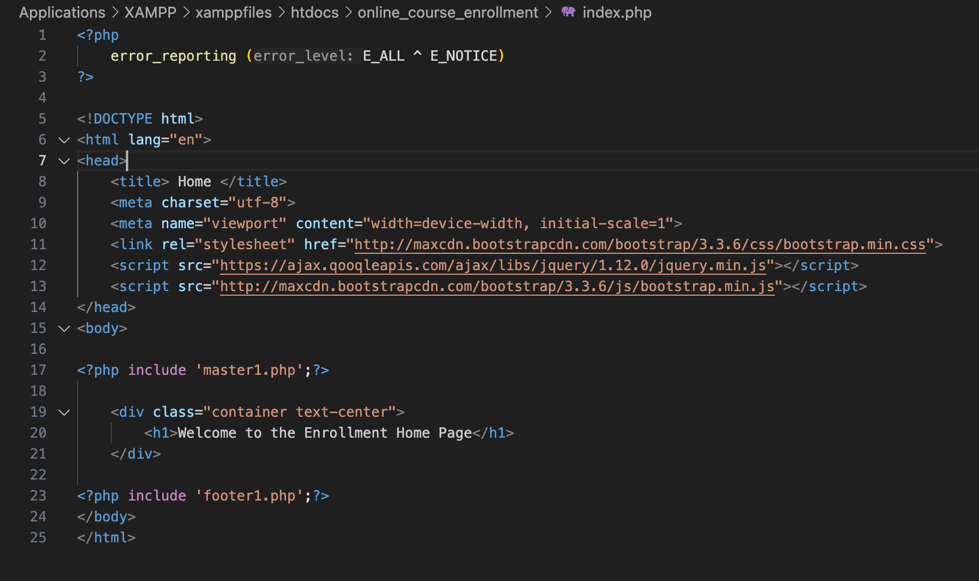
2-17-2025

**How to run a PHP file in XAMPP**

1. Download XAMPP through the web browser & install.
2. Open whichever Independent Development Environment (IDE) that will used for writing the php code or download an IDE if necessary. I use Visual Studio Code.
3. Navigate to the applications folder on the PC and find the XAMPP folder. Select the XAMPP folder and then proceed to opening the htdocs folder. Once inside this folder, a new folder can be created for saving whichever code that will be utilized for the website. I named mine, online\_course\_enrollment.
4. Select open new file within VS Code and begin writing the code for your website. Once completed, save that file into the online\_course\_enrollment folder within the XAMPP application folder.
5. For my website, the file index.php will take you to the home page.
6. To access this, open the web browser and type localhost/online\_course\_enrollment/index.php
7. If all code is correctly entered, your newly designed home page will be displayed. (See below images for the online enrollment website home page & index.php code)



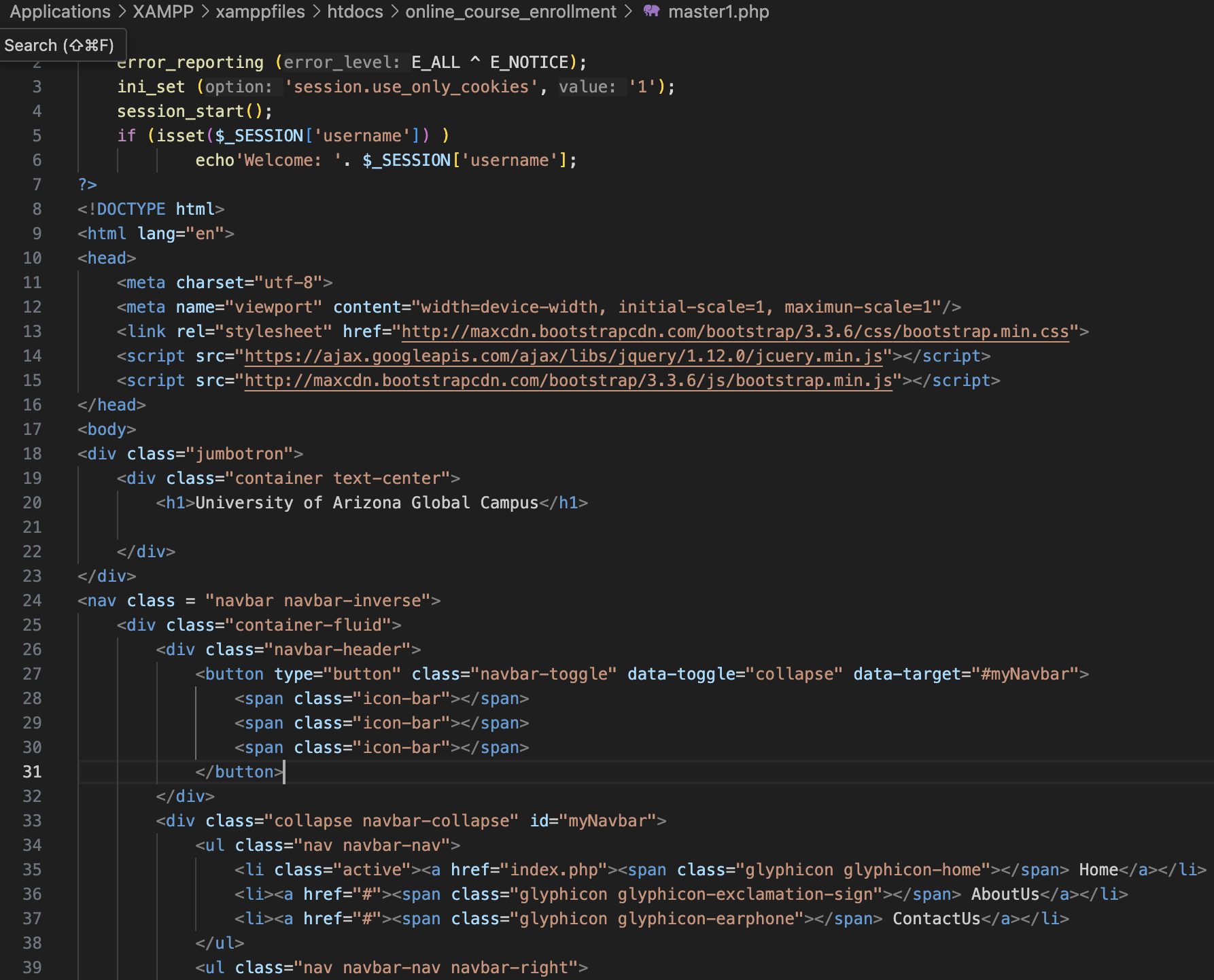
Home Page



index.php code

**Home, Login, and Registration Page**

The base design of the online enrollment website was encapsulated in the master1.php code. This file included the html code that would provide the display for every page within the website. This display would be consistent across the home, login, and registration pages. The master1.php code will be referenced in any subsequent created pages to ensure the website styles are consistent. See below of the source code for the master1.php file.

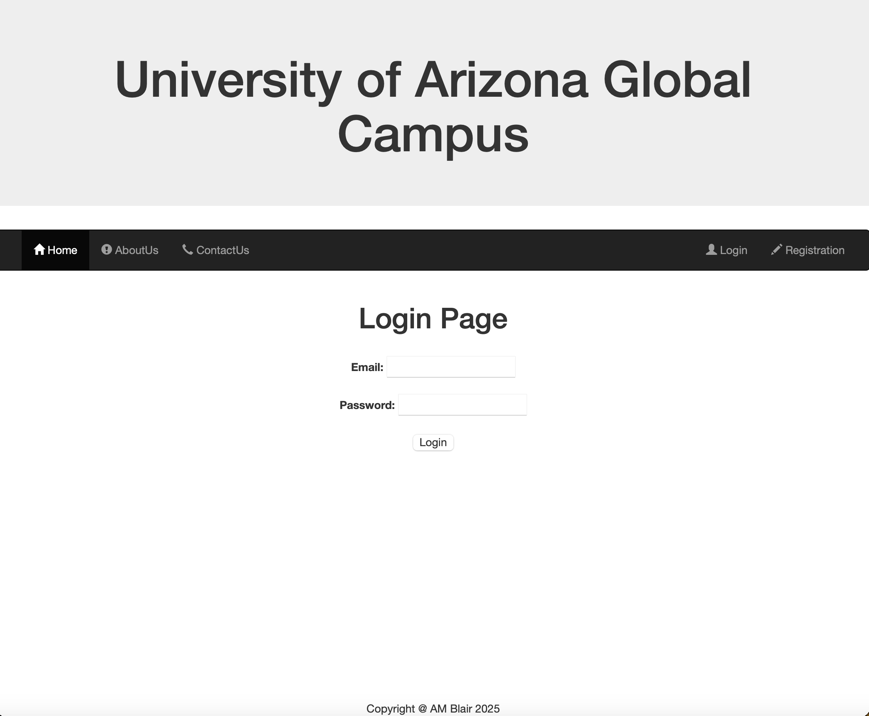


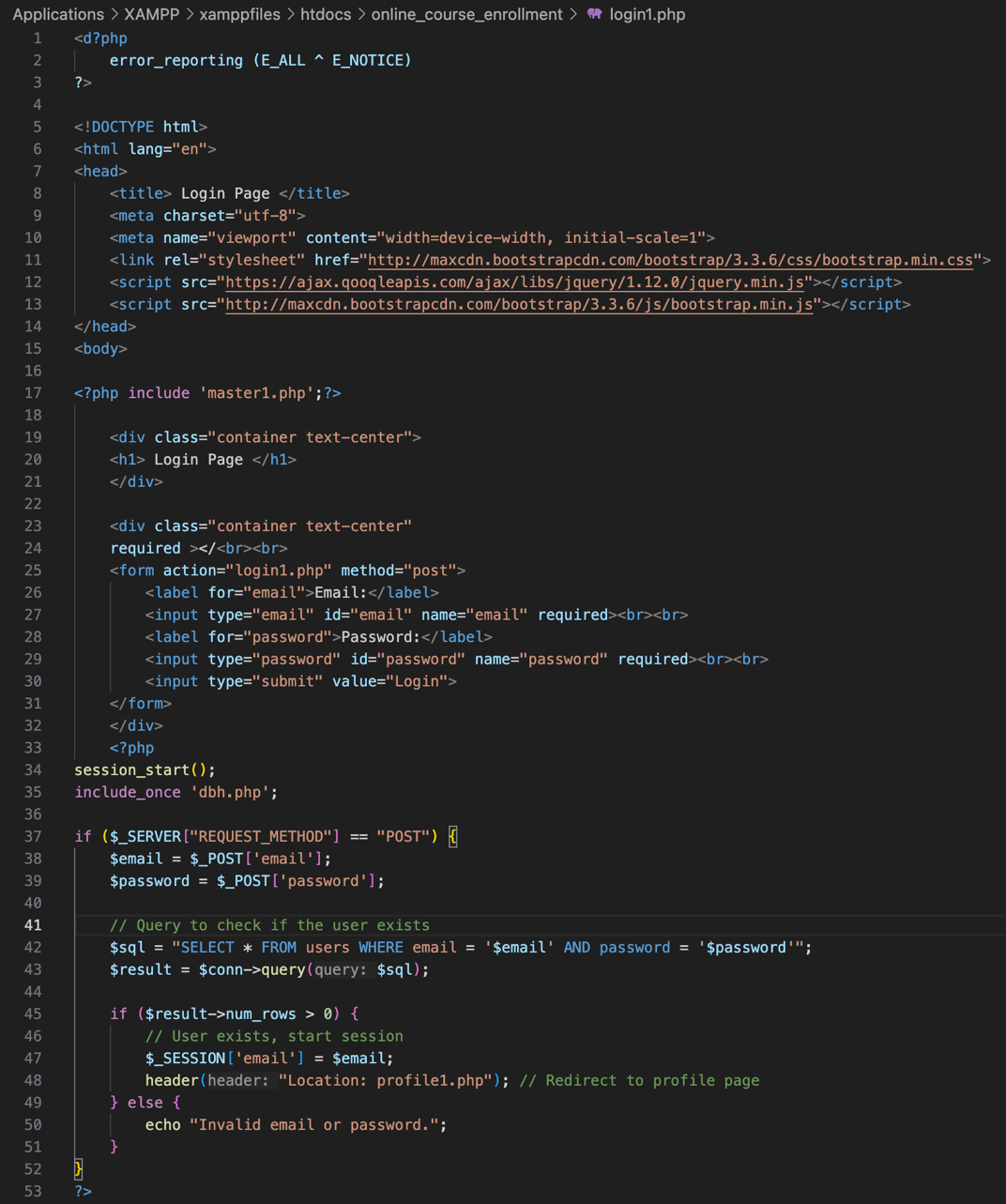


master1.php

The first objective for completing the login page was to create an HTML form that was to capture the user’s email and password. In the body of the html form, the action to connect the form to the subsequent php code (login1.php) was established, along with the post method. Within this form, the display was defined which would be used to capture entered information from users and allowing submission once the email and password fields were completed. This action was achieved by using the label and input functions.

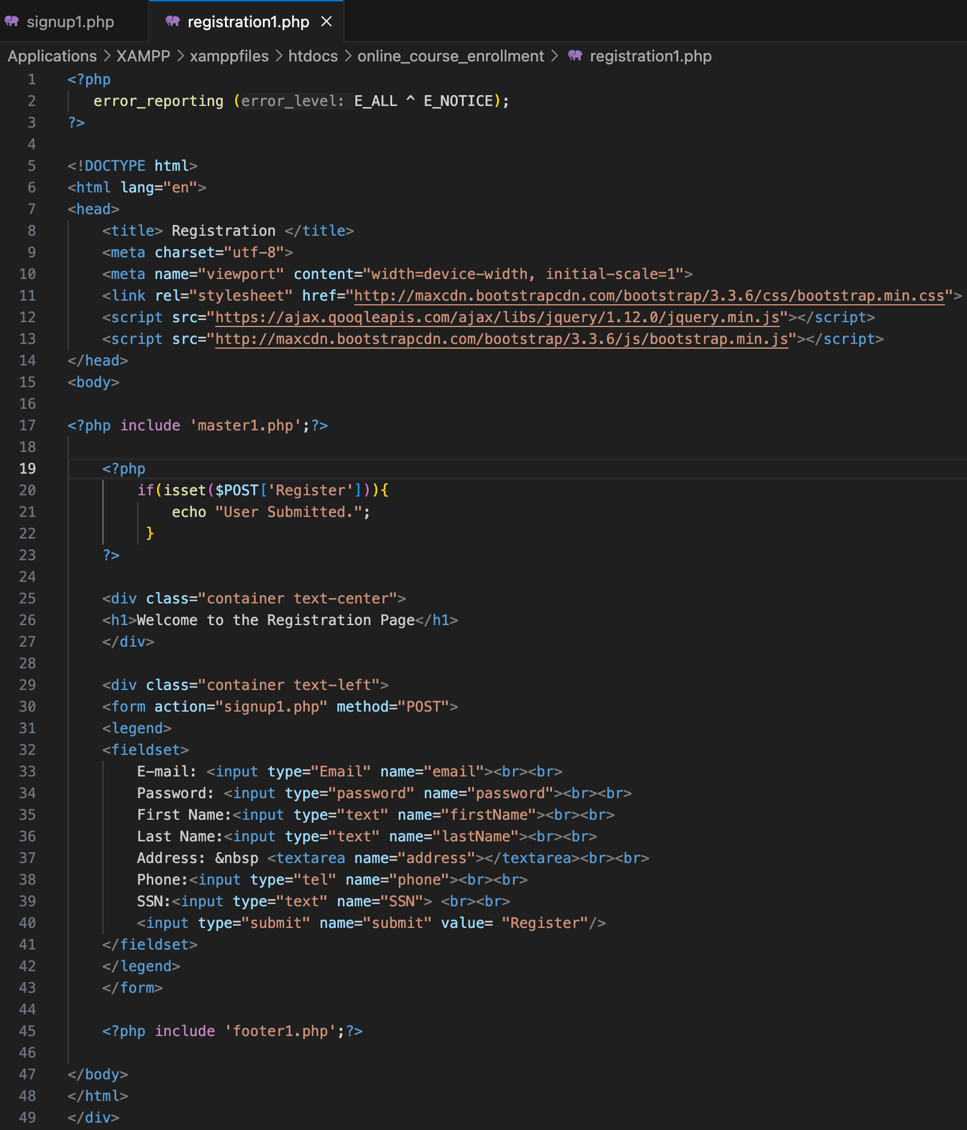
The second portion of the source code defined the php code that communicates to the database and verifies that entered data entries are valid. The first step was connecting to the database by using the session\_start() and including that it should connect to ‘dbh.php’, which is the database connection php file. Once connection is established, the inputs will trigger the system to check if the entered credentials are present in the database or not. If they are, the user will be directed to the profile page (profile1.php). If the credentials are not in the database, then an error message will indicate so, and display the message “Invalid email or password”. This will then clear the form and allow the user to re-enter credentials to try to re-login. See below snippets of the user login page and the associated php code file.



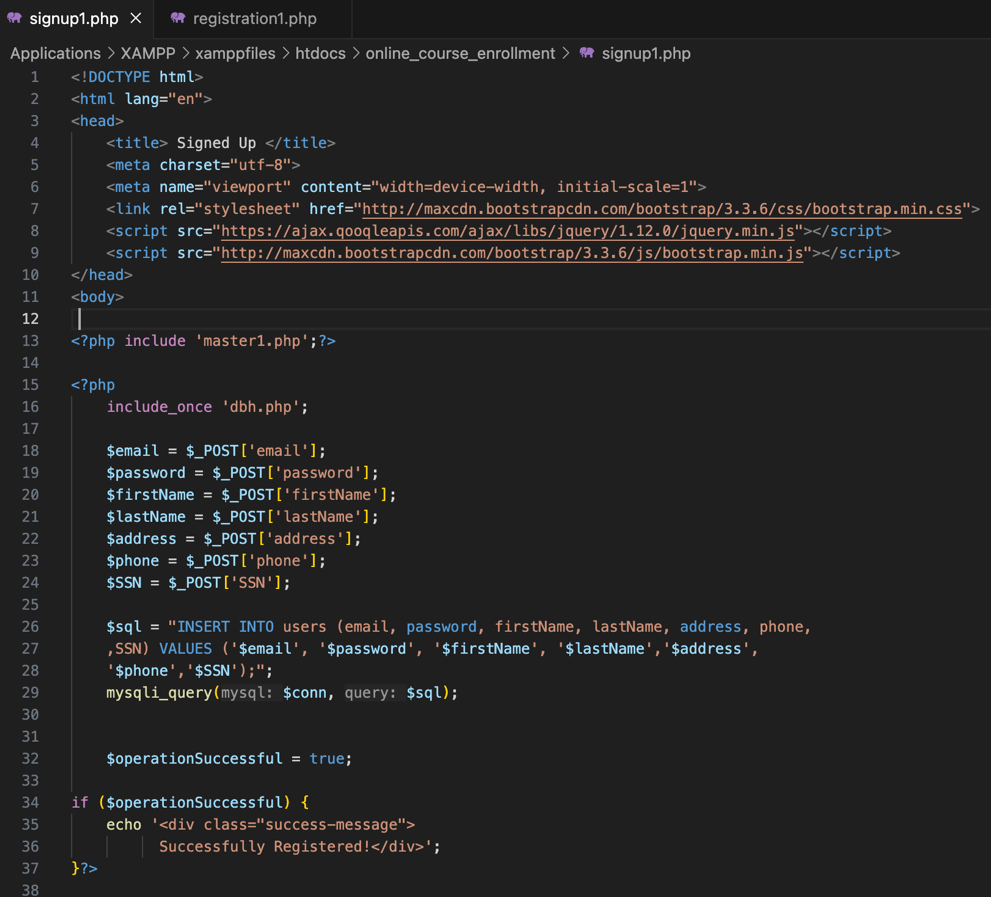


login1.php

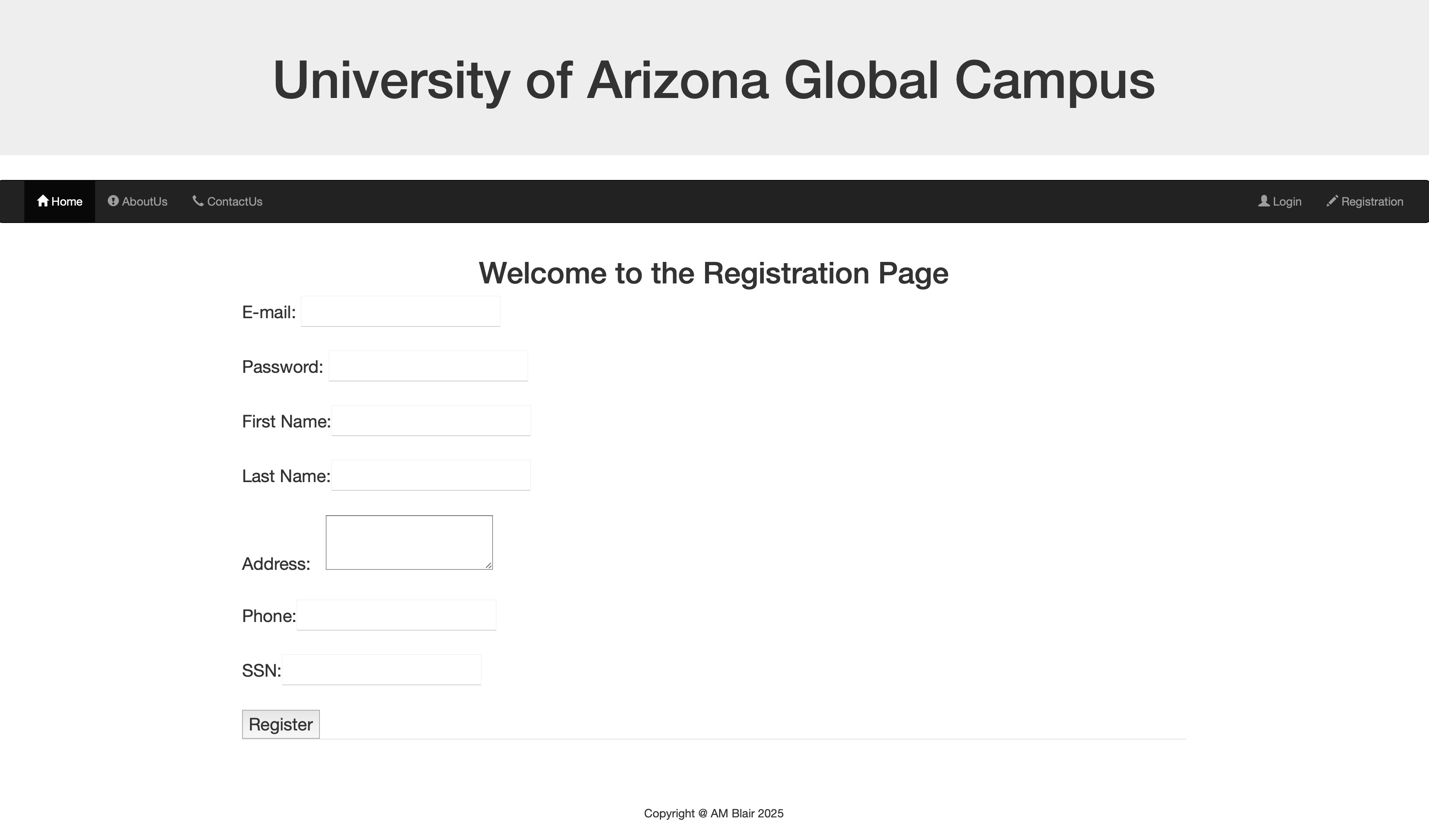
To create the Registration page, I had to first begin by using html to create the outline of the website. I titled the page to be called ‘Register’ because it would be used for the registration process within the website. Next, I specified what I wanted the style and format of the page to be, such as the font characteristics and the webpage background. From there the headings were defined in accordance with the design style I wanted to use, I decided to keep it simple. To design the format of the registration form, I used the ‘legend’ and ‘fieldset’ commands to list out the format for users to type input into the form. This was related to the signup1.php file through the ‘action’ command and ‘POST’ method. The signup1.php source code included variables that correlated to SQL commands which would accept data and communicate with the database. This was made possible by connecting the signup1.php source code to the dbh.php source code using the include\_once command. Once each of these separate segments of source code were created and linked, the registration page was therefore able receive user input. Then once the register button is pressed, the data will be transmitted and saved into the database. See below images of the php code for the registration and sign-up pages, and the registration page display.



registration1.php

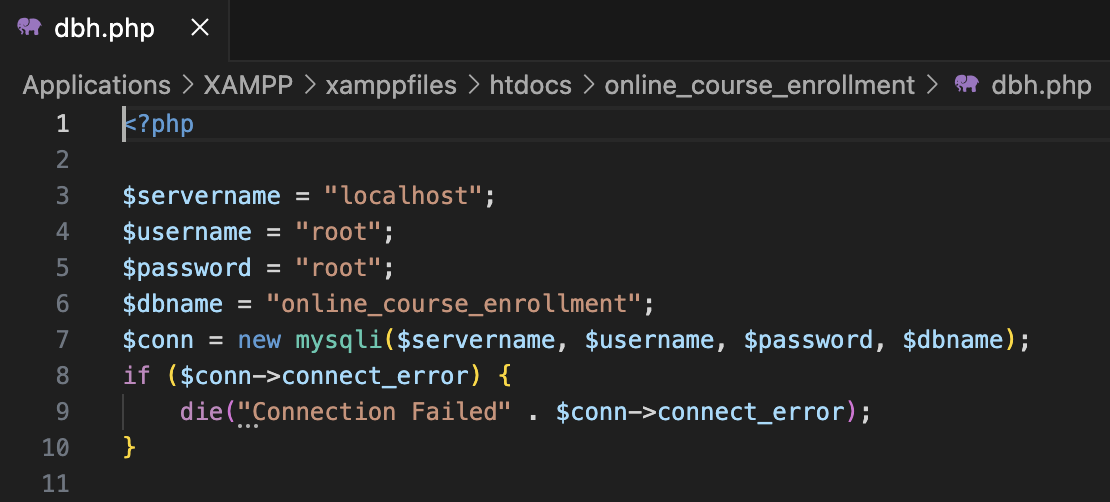


signup1.php



**Connecting to the Database**

To ensure that all of the functionalities remain connected, I first created the connection to the database through file, dbh.php. The dbh.php source code established the connection to the database through php code using the $conn = mysqli() connection command. Variables were set for each of the key elements of the database connection, which includes the server name, username, password, and lastly the database name. The mysqli command connects to the established database created in phpMyAdmin. See below for the dbh.php source code.



dbh.php

**MySQL Database Creation & Tables**

To create a new database with phpMyAdmin is simple. The first step is to select ‘New’ on the left navigation bar where all other databases are found. Once on the create database page, you simply enter the name of the database that you would like to create, and then select the create submission button. Once the database is created, you will be able to create tables within the database. When new table is selected, the user is prompted to name the table and then assign labels to the different columns that will incorporate data in the table. As you can see in the screenshots below, my columns were named: id, email, password, firstname, lastname, address, phone, and SSN. Within the table structure, I defined that the id column would auto-increment and that this value would be utilized as the primary key of the table. To ensure that data can be retrieved by means other than using the id, I indexed the email as the foreign or secondary key. The email is therefore used when the user signs into their enrollment account on the website, when accompanied by the corresponding password for the account. Figure 1 below shows the creation of the table named ‘users’ in the ‘online\_course\_enrollment’ database. Figure 2 below shows a submitted user into the database, with the auto-incremented id of 1. Lastly, Figure 3 below shows the structure of the table elements. This includes the name, type, collation, attributes, etc. of each column. In the table structure there is a gold key symbol beside ‘id’, this is indicating the ‘id’ as the primary key of the table. Subsequently, the grey key symbol beside ‘email’ is indicating the indexed foreign or secondary key.

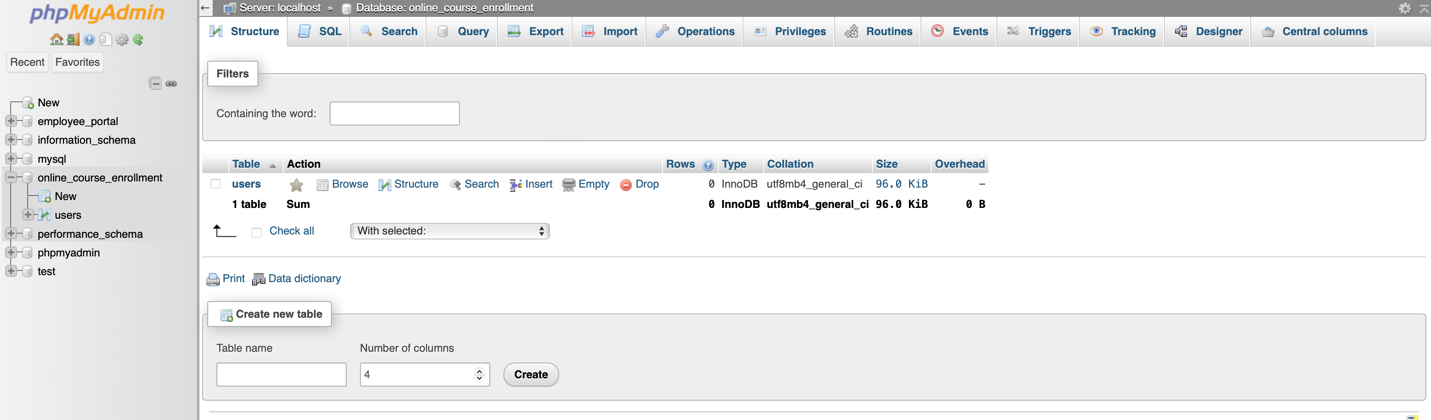


Figure 1

A screenshot of a computer

Description automatically generated

Figure 2

A screenshot of a computer

Description automatically generated

Figure 3