**Database Development and Class Registration**

Nicholi D’Amato

The University of Arizona Global Campus

CST 499: Capstone for Computer Software Technology

Instructor: Joseph Rangitsch

April 24, 2024

Adding more functionality the website required learning a lot more about how functions worked so that I could create object oriented code to make reusing segments later, such as being able to register for a class. By using the workflows and requirements as a guide, I was able to break some of the work down and understand the different tasks required and make sure those tasks were accounted for with methods. We can see the work needed below.

## Create your tables within the MySQL database related to the rest of your design.

All required tables were created at the start:

* Classes – a list of available classes with relevant information
* Students – a list of all students with login information
* Registrations – Association tables for associating students with a class
* Waitlist – Association tables for associating students with a waitlist for a class

## Create the different pages related to the rest of the requirements and the design per your work in Week 1 and Week 2.

The only page that needed to be created after registration and login was the Class portal page. This page allows registering for any listed class. If there are not enough available spots the student is added to the waitlist. If there are the student is added to the class and the student count is updated.

When a student withdraws from a class, they are removed from the class and the student count is updated. If there are any students on the waitlist for that class, they are added to the class and the waitlist is updated.

When a student withdraws from a waitlist, they are removed from the list and all students higher on the waitlist have their position updated.

Additionally, the user can filter the list of available classes by semester: Spring or Summer.

## Generate screenshots of the database and the tables you created.

A screenshot of a computer

Description automatically generated

Figure 2: Students Table

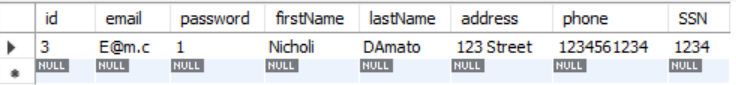


Figure 3: Students Post Registration



Figure 4: Classes Table

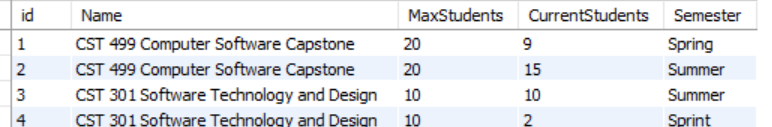


Figure 5: Classes Populated



Figure 6: Registrations Table

A close up of a computer screen

Description automatically generated

Figure 7: Post Class Registration



Figure 8: Waitlist Table

A screenshot of a computer

Description automatically generated

Figure 9: Post Add to Waitlist

## Generate screenshots of the different pages that you created.

A screenshot of a computer

Description automatically generated

Figure 1: Class Registration Page

## Summarize your experience going through the implementation phase.

Overall, the experience of creating the implementation phase was very rewarding. I started out with simple queries to register for a class with the idea that each workflow would be one set of logic and queries. However, once I began creating the workflows for withdrawing from classes I felt that a more Object Oriented approach would be better, so I broke everything up into a series of methods. Some of the methods are only used once, such as “CheckWaitlistPosition” while others are used a lot such as “RegisterForAClass”. I felt that it would be worthwhile to create each step as a method incase, I needed to come back to it for something unaccounted for in a workflow.

I also had some issues with populating the list of classes based on availability. In the end I decided to find all classes that the student is not in, and not on the waitlist for by joining those queries and looking for only rows with null.

## Generate screenshots of the PHP code to implement your logic.

A screen shot of a computer program

Description automatically generated

Figure 10: First half of Class Registration Logic

A screenshot of a computer program

Description automatically generated

Figure 11: Second half of Class Registration Logic

A screen shot of a computer program

Description automatically generatedFigure 12: First half of class portal page

A screenshot of a computer program

Description automatically generated

Figure 13: Second half of class portal page

## Provide screenshots of all developed pages, database, tables, layout, and source code. A screenshot of a computer program Description automatically generatedA screen shot of a computer program Description automatically generated

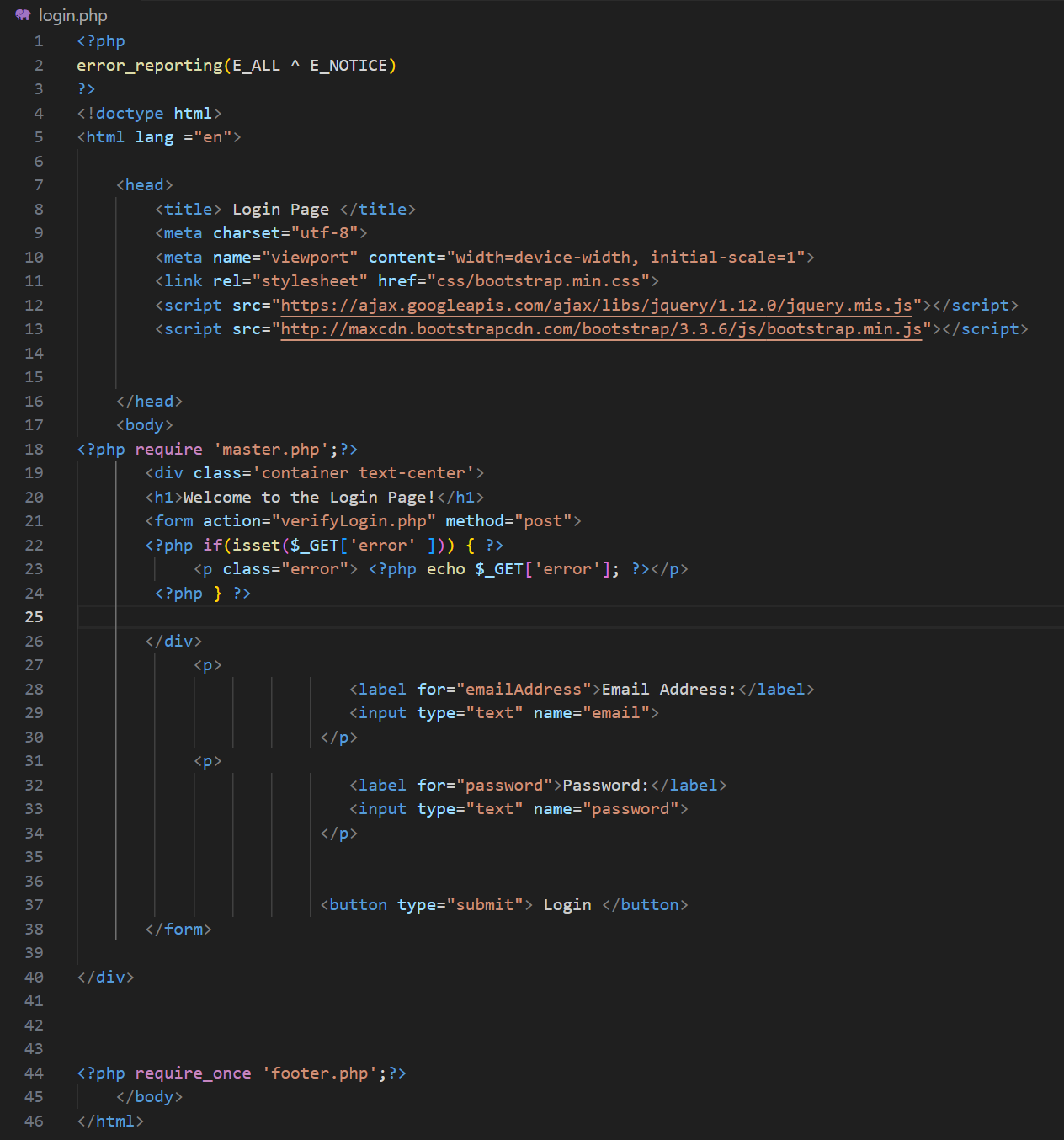
A computer screen shot of a program

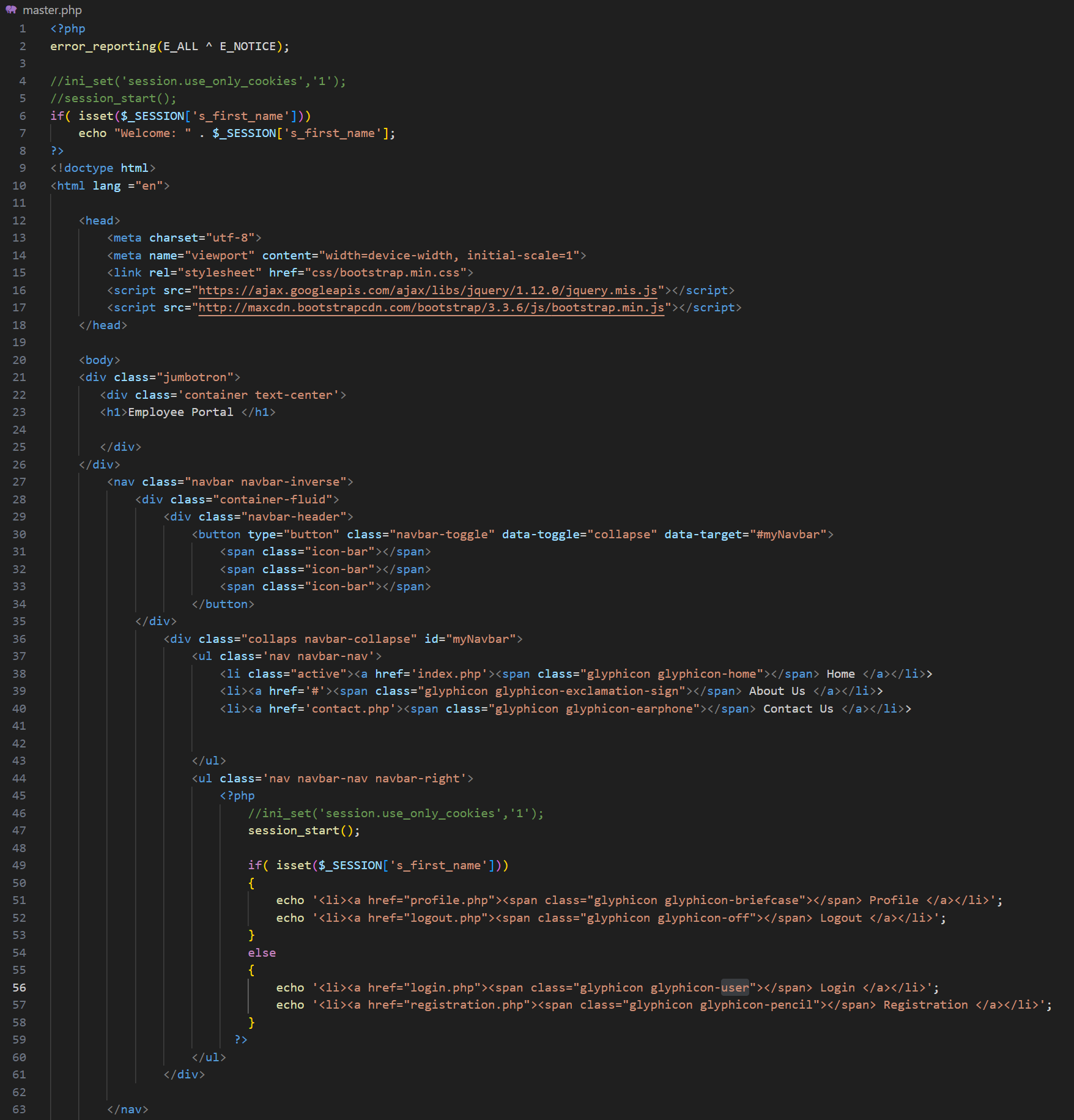
Description automatically generated A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated





With all the work required to get these pages working in the way that I wanted them to I feel like I learned a lot. I specifically took the time and effort to create what would be useful code, or hopefully good code in a real-world environment to help with my learning. Taking this extra effort made the work more rewarding, and I look forward to being able to use this knowledge for future, real life work.

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

Nicholi D’Amato

The University of Arizona Global Campus

CST 499: Capstone for Computer Software Technology

Instructor: Joseph Rangitsch

April 24, 2024

Creating a website that allows for interactions with a database, such as a student portal page that students can use to register for classes has a fair number of steps. We also need to run a database for data to be entered, creating webpages with code such as PHP to interact with that database. And ensuring that everything works correctly with a minimum number of bugs and good workflows. We will showcase some of those steps below, as well as what you might expect the code itself to look like.

## Explain how to run a PHP file in XAMPP.

Running a PHP file in XAMPP can be simple if your environment is setup correctly, if not, there may be some troubleshooting!

1. First, download and install XAMPP from [here.](https://sourceforge.net/projects/xampp/)
   1. Default settings are fine.
2. Once installed, launch the XAMPP Control Panel **as an administrator.**
3. If you have previously installed Apache or MySQL, or if you have anything running on the default ports, you can change the ports these services run on in the Configuration -> Service and Port Settings.
   1. If you change these ports, you must then go to the “Config” button on the line for that service, select the hddpd.conf and change any instances of the original port, to the new one. For example, changing Apache to run from port 80 to 85
4. Now, place your PHP file in the directory you installed XAMPP in, preferably in a new folder for that website.
5. Navigate to the port in your browser, such as http://localhost:85/CST%20310%20Site/index.php

## Create the landing page, login page, and registration page for new users.

A black and white text

Description automatically generated

Figure 1: Index

A screenshot of a login page

Description automatically generated

Figure 2: Login

A close-up of a login

Description automatically generated

Figure 3: Registration

## Create the MySQL database and tables.

A screenshot of a computer

Description automatically generated

## Discuss the MySQL database functions that you used and the steps you took to create the database connection custom class.

To create the database with the users table:

* I created the database cst310website
* I used the UI to add a users table
* I added all the fields that were required, and designated the id and email fields as primary keys, with the other fields not-null to ensure they are filled in on insert.

For my custom connection class I have:

## Develop the registration page layout.

A screenshot of a computer

Description automatically generated

## Develop the registration page PHP source code.



## Develop the table that saves the user information in the database.

A screenshot of a computer

Description automatically generated

Figure 4: Table After Insert

## Explain the steps taken to create the registration page and save the user information in the database.

The first step was to create a form on the registration page that will pass information into the insert code running on the insert.php page. The form has a label and an input for each of the fields that we are gathering, as well as a submit button.

On hitting the submit button, we post the inputs to the insert.php page. The insert then connects to the database, gathers the values we are going to insert into variables, then inserts those variables into the database. Template used can be found on the GeeksForGeeks *How to Insert Form Data into Database using PHP ? page.*

## Provide screenshots of all developed pages, database, tables, layout, and source code.

A screenshot of a computer

Description automatically generated

Figure 5: Students Table



Figure 6: Classes Table

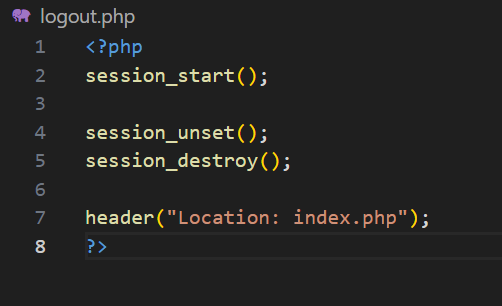


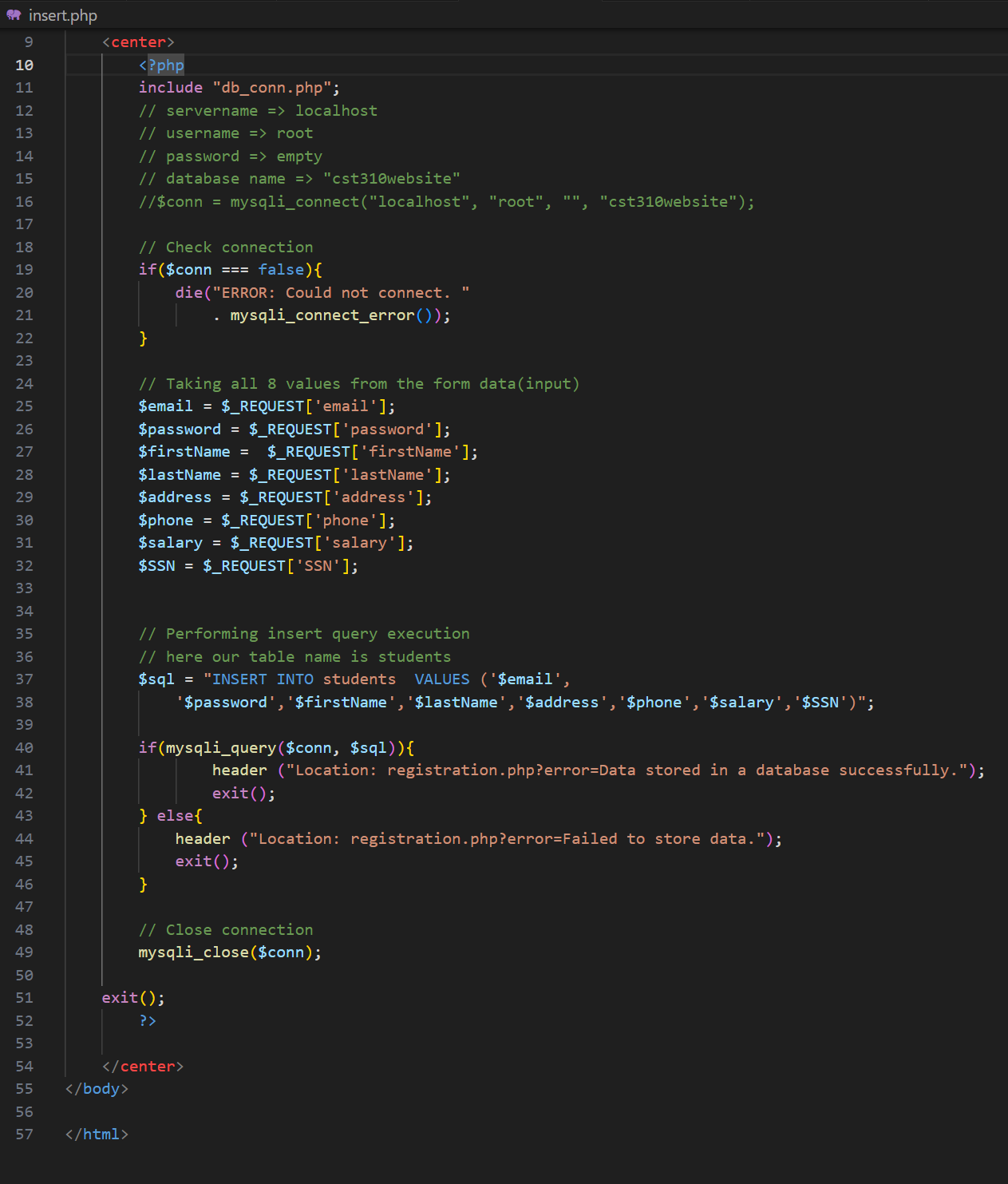
Figure 7: Registrations Table

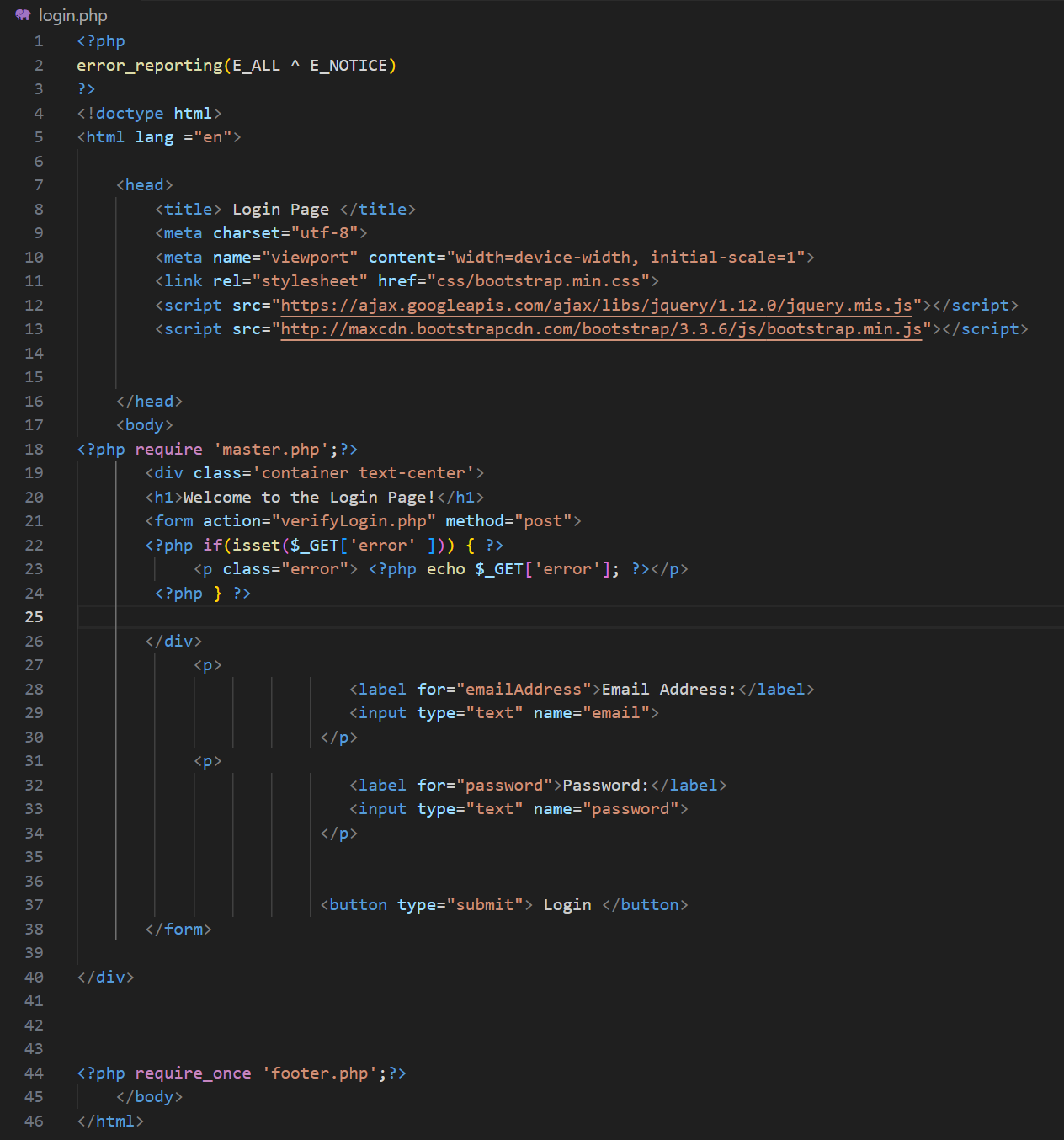


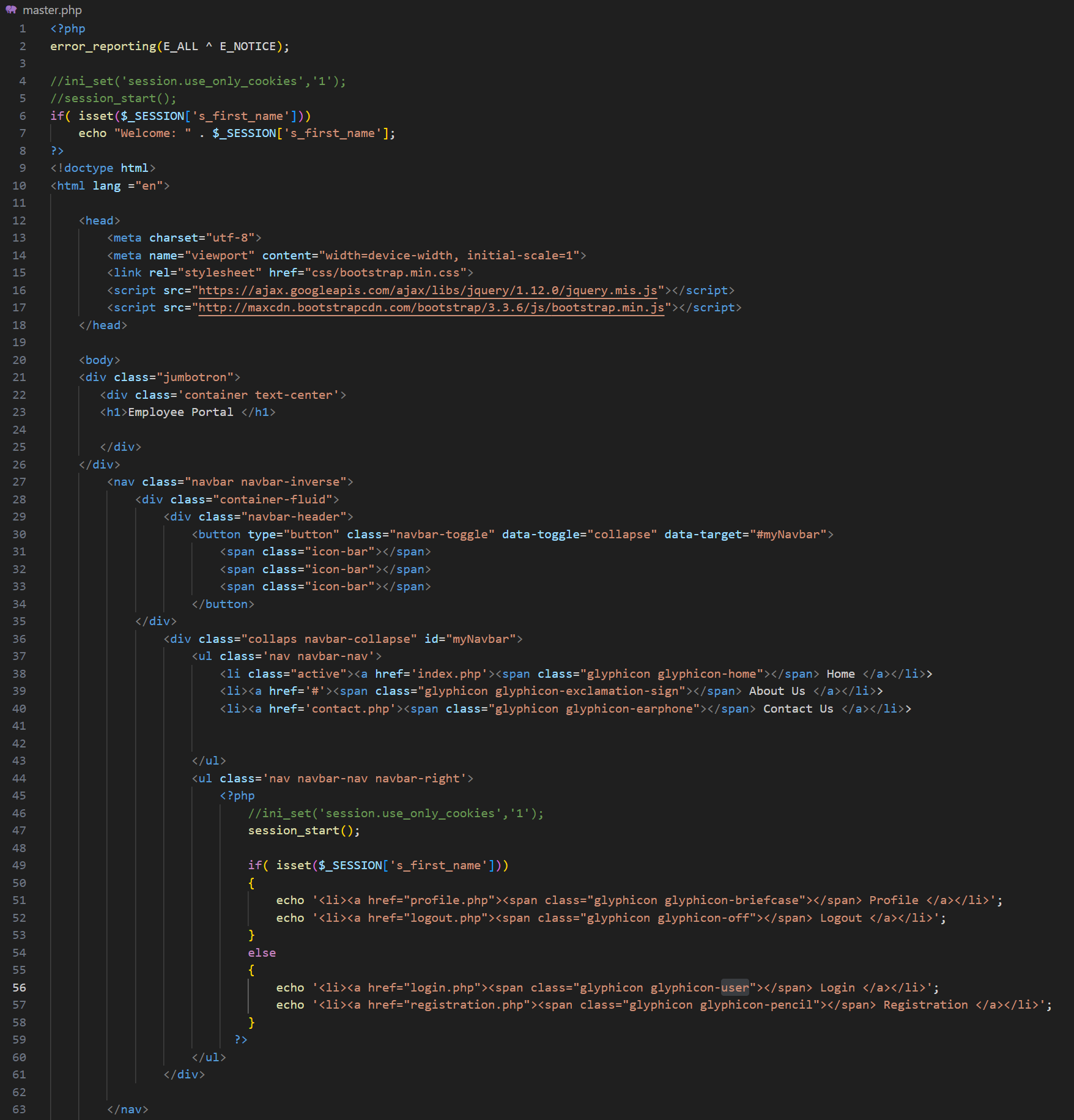
Figure 8: Waitlist Table







All the steps above are just the start of the work we are doing on this project and cover only the bare bones. We still need to create the classes, the class schedule, the pages to view those classes, and the methods and logic to register and withdraw from classes. Working through these issues will be a fun challenge and be quite educational!

References:

*How to Insert Form Data into Database using PHP ?* (n,d,). GeeksForGeeks. Retrieved December 14, 2023 from [https://www.geeksforgeeks.org/how-to-insert-form-data-into- database-using-php/](https://www.geeksforgeeks.org/how-to-insert-form-data-into-%09database-using-php/)