**Course Registration System:**

**Website Layout (Home/Login/Registration) & Database Development/Class Registration**

Angelica Topete

University of Arizona Global Campus

CST 499: Computer Software Technology Capstone

Joseph Rangitsch

July 17, 2023

**Database Development & Class Registration**.

This week's focus was the continuation of the course registration systems development phase, which was initiated in Week 3. This week's primary objective was to complete the remaining functionalities of the course registration system. This activity involves creating additional tables in the database, establishing seamless connections between pages and the backend, and implementing essential features like class registration, viewing enrolled courses, adding new courses to the semester, and managing class deletions from users' schedules. This paper aims to provide an overview of this week's development process by summarizing key aspects of the implementation phase. Screenshots of the created databases/tables, the different pages designed, and the logic applied to achieve functionality showcase the progress in building an efficient and user-friendly course registration system.

**1. New Database & Tables (created this week)**

*Figure 1.1 'project' database (created week 1, updated tables week 4).*

A screenshot of a computer

Description automatically generated

Note. The image displays the 'project' database along with its corresponding tables, showcasing the foundational structure of the course registration system.

*Figure 1.2 Screenshot of Structure of 'courses' Table* A screenshot of a computer

Description automatically generated

Note. The image provides an insight into the 'courses' table, a crucial component of the course registration system. This table houses comprehensive university course information, including its 'id' (serving as the primary key), 'name', 'semester', 'max\_enrollment', 'enrollment', and 'waiting\_list' data, enabling efficient management of course registrations and waiting lists.

*Figure 1.3 Screenshot for 'courses' Table (Including data)*

A screenshot of a computer

Description automatically generated

Note. The image showcases the 'courses' table containing 22 distinct courses essential for the course registration system. These courses are displayed on the 'current registration' page (Figure 4.1). The 'id' serves as the primary key, allowing proper classification of student course enrollments in the enrollment table.

*Figure 1.4 Structure of 'enrollment' Table* A screenshot of a computer

Description automatically generated

Note. The image illustrates the 'enrollment' table, containing essential data fields such as 'course\_id', 'student\_id', 'name', and 'email'. This data plays a crucial role in the process of enrolling a student into a specific course, facilitating smooth course registration within the system.

*Figure 1.5 Structure of 'waiting\_list' Table*

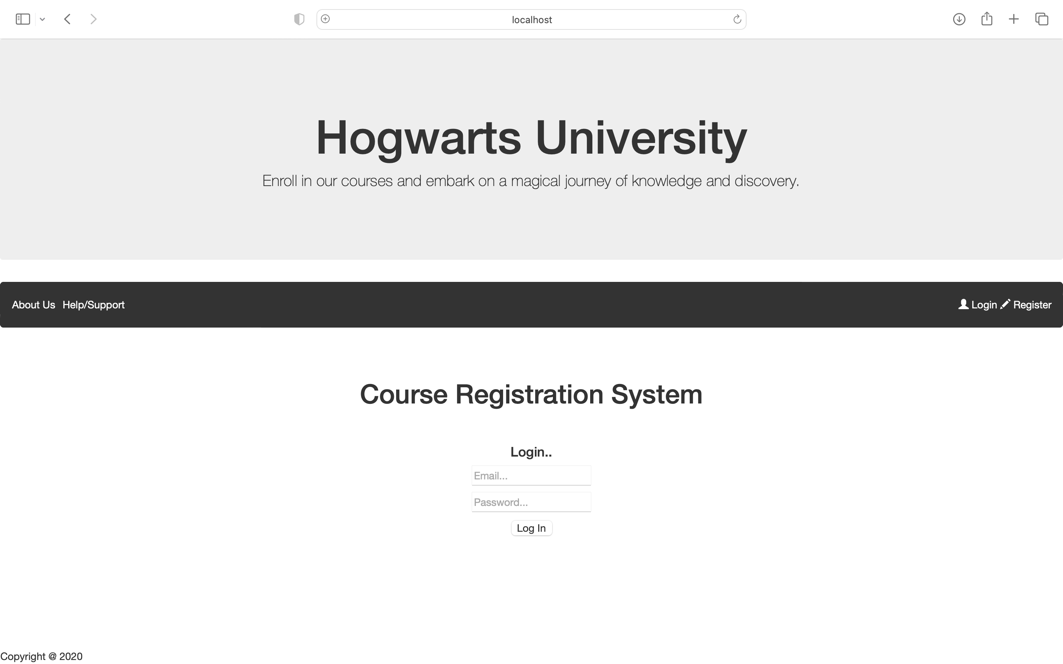
A screenshot of a computer

Description automatically generated

Note. The image displays the 'waiting\_list' table, featuring important data fields such as the primary key 'id', 'course\_id', 'student\_id', 'name', 'email', and 'timestamp'. This data serves a crucial role in enrolling a student into the waitlist for a specific course, ensuring effective management of waitlist registrations within the system.

***2. Landing Page, Registration, & Log in***

*Figure 2.1 Fictional University Course Enrollment Home Page/Login*



Note. The image depicts the home page of the fictional Hogwarts University course enrollment system. Upon accessing the site, users are greeted with the website's name. The navigational bar features intuitive icons that guide users to their desired pages, ensuring seamless navigation and a user-friendly experience throughout the site.

*Figure 2.2 Layout of the Registration Page for User Profile Creation*

**A screen shot of a computer

Description automatically generated**

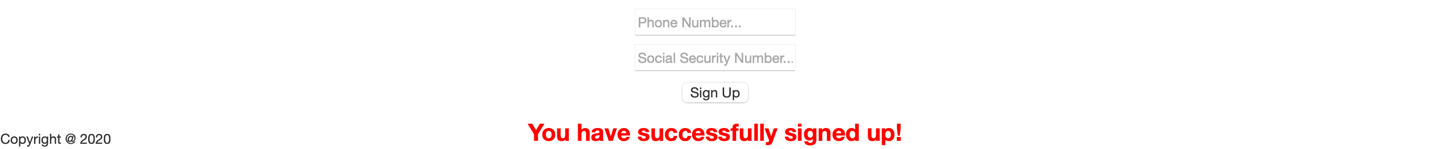
Note. The image displays the registration page layout, where users can create their profiles by providing essential information such as email, password, name, address, phone number, and social security number. The data entered on this page is seamlessly linked with the 'tblUser' table, ensuring secure and accurate storage of user profile details within the system.

*Figure 2.3 Input Data on the Registration Page & Database Update*A screen shot of a computer

Description automatically generated

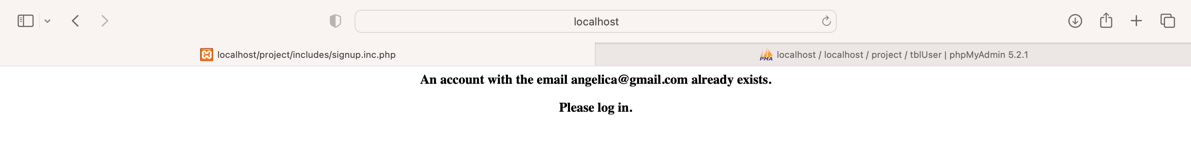
Note. The image depicts the input data fields on the registration page, where users enter their information, including email, password, name, address, phone number, and social security number. The system processes the data upon clicking the signup button, and the user is redirected to a successful registration page. Simultaneously, the corresponding tables in the database are updated with the newly registered user's information, enabling seamless login functionality with the provided credentials.

*Figure 2.4 Successful User Registration*



Note. The image displays the message users receive after successfully registering to the system. This confirmation message acknowledges the successful completion of the registration process, assuring users that their registration was completed successfully.

*Figure 2.5 Error message for Existing Account Creation*



Note. The image illustrates the error message displayed by the system when a user attempts to register for the application with an existing account. This informative message alerts users that the provided account details exist within the system, prompting them to log in with their existing credentials.

*Figure 2.6 Updated 'tblUser' after Successful Account Registration*

*A screenshot of a computer

Description automatically generated*

Note. The image showcases the 'tblUser' table, reflecting the latest update following the successful registration of a user account. The data captured during registration, including login credentials and personal details, is securely stored within this table. Subsequently, this information is the foundation for various system functionalities, such as user login, course registration, and class deletion.

***3 User Login***

*Figure 3.1 User Logging into the System*

*A screenshot of a login page

Description automatically generated*

Note: The image captures the user using their credentials to log into the course registration system. This secure login process enables authorized users to access their accounts and use the system's functionalities, such as enrolling in courses, managing their schedules, and accessing personalized information.

*Figure 3.2 Successful User Login & System Access*

A screenshot of a computer

Description automatically generated

Note: The image captures the user successfully logging into the system using their credentials. Upon successful login, the system redirects users to their the home page, where they can view, access and manage their desired courses. The navigational icons are updated to offer a broader range of options exclusive to registered users. This includes access to course registration, the user's profile, and their schedule. Additionally, the user can log out of their profile once they have completed their desired tasks, ensuring a secure and user-friendly experience within the course registration system.

**4 Course Enrollment & Waitlist Enrollment (Avalible at login)**

Figure 4.1 Course Enrollment Page -Available Course & Registration Options

A screenshot of a computer

Description automatically generated

Note. The image showcases the course enrollment page, accessible by clicking on the 'Course Registration' icon in the navigational bar. The page displays a comprehensive list of available courses, providing essential information such as their unique 'id', semester details, and the current number of enrollments. Each course listing also features a button allowing users to enroll in the course directly or join the waiting list, depending on availability.

*Figure 4.2 Course Enrollment Verification – Name & Email Confirmation*

A screenshot of a computer

Description automatically generated

Note. The image represents the process after a user chooses a course and clicks 'Enroll'. To verify their enrollment, the user is prompted to provide their name and email. This crucial step ensures accurate and secure registration, confirming the user's enrollment in the selected course. By validating their identity through name and email, the system guarantees that the enrollment process is smooth and error-free, providing the user with a seamless course registration experience.

*Figure 4.3 PHP Source Code for Registration Process*

*A computer screen shot of a program code

Description automatically generated*

The provided PHP code handles the enrollment process in a course registration system. It establishes a connection to the database using the 'dbh.inc.php' file. When a POST request is received, the code verifies if the required data (course\_id, name, and email) is set. It then retrieves the logged-in user's student ID from the session or another appropriate method. Next, the code updates the 'enrollment' table by inserting the enrollment data for the selected course. It prepares and executes a SQL statement to insert the enrollment details into the 'enrollment' table and updates the 'enrollment' count in the 'courses' table for the specific course.

In case of successful enrollment, the code redirects the user to an 'enrollment\_success.php' page, indicating successful registration. If the form submission is invalid or data is missing, the user is redirected back to the 'courseregistration.php' page, or an error message is displayed. Overall, the code efficiently handles the course enrollment process, ensuring that user data is properly stored and updated in the database while providing feedback to users about the enrollment status.

*Figure 4.4 Successful Course Enrollment Page*

A screenshot of a computer

Description automatically generated

Note. The image displays the page that appears after a successful course enrollment. This confirmation page assures users that their course enrollment was completed successfully.

*Figure 4.6 Updated' enrollment' Table After Course Registration*

A screenshot of a computer

Description automatically generated

Note. The image displays the 'enrollment' table, showcasing the recent update following user registration. The table now includes the user's name, email, and the chosen course id, reflecting the successful enrollment of the user into specific courses. This data will play a pivotal role in generating the schedule page, where users can view their current registrations and have the option to withdraw from courses as needed. The updated 'enrollment' table ensures seamless management of user course enrollments, providing a personalized and convenient experience within the course registration system.

*Figure 4.3 PHP Source Code for Wait List Process*

*A computer screen shot of a program

Description automatically generated*

The provided PHP code handles adding students to a waiting list in a course registration system. When a POST request is received, the code checks if the required data (course\_id, student\_id, name, and email) is set. The code proceeds with the waiting list registration process if the data is present. The code establishes a connection to the database using the 'dbh.inc.php' file and prepares an SQL statement to insert the waiting list data into the 'waiting\_list' table. It uses a prepared statement to prevent SQL injection and binds the values to the statement. Upon execution of the prepared statement, the student's details are inserted into the 'waiting\_list' table. If the operation is successful, the code redirects the user to a 'waiting\_list\_success.php' page, indicating a successful addition to the waiting list. In case of any errors during execution, appropriate error messages are displayed. After registration, the code closes the database connection, ensuring proper resource management. If the form submission is invalid or data is missing, the code redirects the user back to the 'courseregistration.php' page or displays an error message. Overall, the code efficiently handles the process of adding students to the waiting list, ensuring that the database is updated with the relevant details and providing feedback to users about the success of their waiting list registration.

*Figure 4.3 Waitlist Enrollment Verification – Name & Email Confirmation*

*A screenshot of a computer

Description automatically generated*

Note. The image illustrates the process after a user chooses a course and clicks 'Join Waitlist'. To verify their enrollment to the waitlist, the user is required to provide their name and email. This essential step ensures accurate and secure waitlist registration, confirming the user's position in the selected course's waitlist. By validating their identity through name and email, the system ensures a seamless and efficient waitlist enrollment process, allowing users to join courses that are currently at capacity.

*Figure 4.5 Successful Waitlist Enrollment Page*

A screenshot of a computer

Description automatically generated

Note. The image displays the page that appears after a successful waitlist enrollment. This confirmation page assures users that their waitlist enrollment was completed successfully.

*Figure 4.7 Updated 'waiting\_list' Table After Waitlist Registration*

*A screenshot of a computer

Description automatically generated*

Note. The image displays the 'waiting\_list' table, reflecting the recent update following the user's enrollment into a course. The table includes essential data fields such as 'id', 'course\_id', 'student\_id', 'name', 'email', and 'timestamp'. This data will be instrumental in generating the schedule page, where users can view their current enrollments, including waitlisted courses, and have the flexibility to withdraw enrollment as needed. The updated 'waiting\_list' table ensures efficient management of user course enrollments, providing a comprehensive and user-friendly experience within the course registration system.

*Figure 4.7 Schedule Page – Registered Classes & Enrolled Waitlists*

*A screenshot of a computer

Description automatically generated*

Note. The image showcases the schedule page, accessible via the navigational bar. This page provides users a comprehensive view of their registered classes and enrolled waiting lists. Users can conveniently track their course enrollments and waitlisted courses within the system. Additionally, the page allows users to delete their registrations as needed, empowering them to manage their schedules effectively.

Figure 4.8 PHP Code for Course Schedule Page

A screenshot of a computer

Description automatically generated

The provided PHP code is responsible for displaying the registered classes and the courses in the waiting list for a specific student in a course registration system. It starts by establishing a database connection and retrieving the student's unique identifier ('username') from the session. The code then executes SQL queries to select the registered classes and waiting list courses for the student from the corresponding database tables. If registered classes or courses are on the waiting list, the code displays them in separate tables, showing the course name and semester and offering an option to delete or remove the course from the waiting list. The 'delete' option uses a form with hidden input fields to pass the relevant course and student identifiers to the 'delete\_course.php' script for processing. The code displays the appropriate messages if no registered classes or courses are found on the waiting list. Finally, the code closes the database connection and completes its execution. Overall, this implementation provides an interactive interface for students to effectively view and manage their course enrollments and waiting list entries.

**5. Course and Waitlist Deletion**

5*.1 Users Delete Enrollment to Courses (x's 3)*

*A screenshot of a computer

Description automatically generated*

Note. The image captures the process after a user clicks the 'delete' button for a course on their schedule page. This results in a successful enrollment deletion message, confirming that the selected course has been removed from the user's schedule. The system provides this confirmation message to reassure users that the enrollment deletion was completed successfully, ensuring an organized and tailored course registration experience within the system.

*5.2 Table After User Deleted All Courses ('enrollment')*

A screenshot of a computer

Description automatically generated

Note. The image depicts the table after the user successfully deletes all their courses. This process results in an updated table that no longer contains any records of the user's enrolled courses.

*5.3 Schedule Page After Course Deletion*

A screenshot of a computer

Description automatically generated

Note. The image displays the schedule page after the user deletes their enrolled courses. As a result, the schedule page no longer shows any records of the previously registered courses.

*5.4 Successful Waitlist Deletion Confirmation*

A screenshot of a computer

Description automatically generated

Note. The image captures the process after users click the 'delete' button for a waitlist registration on their schedule page. This results in a successful waitlist deletion message, confirming that the selected waitlist registration has been removed from the user's schedule. The system provides this confirmation message to assure users that the waitlist deletion was completed successfully.

*A screenshot of a computer

Description automatically generated5.5 'waiting\_list' Table After Deletion of all Waitlist Registrations*

Note. The image illustrates the 'waiting\_list' table after users successfully deletes all their registrations. As a result, the table no longer contains records of the user's previously waitlisted courses.

5.6 PHP Code for Course & Waitlist Deletion

A screenshot of a computer

Description automatically generated

This PHP code handles the deletion of course enrollments and waiting list entries for a specific student in a course registration system. Upon receiving a POST request, the code checks if the required parameters' course\_id' and 'student\_id' are set. It then performs SQL queries to check if the student is either enrolled in the course or on the waiting list for the given course. If the student is registered in the course, the code deletes the enrollment record from the 'enrollment' table and accordingly updates the 'enrollment' count in the 'courses table. It then redirects the user to the 'enrollment\_deleted.php' page to display a success message.

If the student is on the waiting list for the course, the code deletes the waiting list entry from the 'waiting\_list' table and redirects the user to the 'waiting\_list\_deleted.php' page to display a success message. If the student is neither enrolled in the course nor on the waiting list, the code redirects the user to the 'courseregistration.php' page or displays an appropriate error message. Overall, this implementation provides a robust mechanism for students to remove themselves from course enrollments or waiting lists in the course registration system.

*5.6 User's Schedule Page after Withdrawal of All Courses & Waitlist Registration*

A screenshot of a computer

Description automatically generated

Note. The image displays the user's schedule page after successfully withdrawing their course enrollments and waitlist registrations. As a result, the schedule page no longer shows any records of the previously enrolled courses or waitlisted courses.

**Experience During Implementation**

During the implementation phase, I had a very rewarding experience that taught me valuable lessons through trial and error. It was a challenging process, but it ultimately made me a better coder and provided practical skills to benefit my future in software technology. The project was enlightening, allowing me to apply the theoretical knowledge I had acquired through my educational experience and see it come to life in a practical setting. Working on this project gave me a deeper understanding of database management, PHP programming, and web development concepts. The hands-on experience and problem-solving in this phase helped me build confidence as a developer and student. Overall, the implementation phase was a significant learning opportunity that I am grateful for, and it has motivated me to continue exploring and growing in the field of software development.