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

Joshua Zehms

The University of Arizona Global Campus

CST 499: Capstone for Computer Software Technology

Professor Charmelia Butler

26 June 2023

To run a PHP file in XAMPP, follow these steps:

1. Start the XAMPP control panel: Open XAMPP and start the Apache server by clicking the "Start" button next to Apache.
2. Locate the "htdocs" folder: By default, XAMPP creates a folder called "htdocs" where you can place your PHP files. The location of this folder depends on the installation directory of XAMPP. For example, if you installed XAMPP on Windows, the default location might be "C:\xampp\htdocs".
3. Place your PHP file in the "htdocs" folder: Copy or move your PHP file to the "htdocs" folder. Make sure the file has a ".php" extension (e.g., "example.php").
4. Start the browser: Open your web browser (e.g., Chrome, Firefox).
5. Access the PHP file: In the browser's address bar, enter the URL "localhost/yourfilename.php". Replace "yourfilename.php" with the actual name of your PHP file.
6. Run the PHP file: Press Enter to load the PHP file. XAMPP's Apache server will process the PHP code and display the output in the browser.

Note: Make sure that XAMPP's Apache server is running before accessing the PHP file in the browser. Additionally, ensure that the PHP file's syntax and code are correct to avoid any errors during execution.

By following these steps, you can successfully run a PHP file in XAMPP and see the output in your web browser.

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

1. CREATE DATABASE: The CREATE DATABASE function is used to create a new database. It takes the name of the database as a parameter and creates an empty database with that name. This function allows you to define the structure and organization of your data.
2. USE: The USE statement is used to select a specific database to work with. Once you create a database, you need to select it using the USE statement before executing any queries within that database. This statement allows you to set the default database for subsequent operations.
3. CREATE TABLE: The CREATE TABLE function is used to create a new table within a database. It specifies the table name and defines the columns, their data types, and any constraints. In the provided example, the CREATE TABLE statement is used to create a table named "users" with columns like "id," "name," "email," and "age." The specified data types for these columns are INT, VARCHAR, and AUTO\_INCREMENT is used for the "id" column to generate a unique value for each new row. Additionally, the PRIMARY KEY constraint is applied to the "id" column to uniquely identify each record in the table.

Here are the steps taken to create the registration page and save user information in the database:

1. Set up the HTML form: Create an HTML form that collects user information such as name, email, password, address, phone, salary, and SSN. Each input field should have a corresponding label and appropriate attributes (e.g., name, id, type). Add a submit button to the form.
2. Style the registration page: Apply CSS styles to the form and its elements to improve the visual presentation. This step involves setting font styles, spacing, and other visual properties using CSS rules.
3. Configure form submission: Specify the action attribute of the <form> tag to point to the PHP script that will handle form submission and save user information in the database. Use the method="POST" attribute to send the form data via POST method, which is more secure for sensitive information like passwords.
4. Create the PHP script: Develop a PHP script that receives the form data, validates it, and saves it in the database. This script should establish a connection to the MySQL database, retrieve the form data using the $\_POST superglobal array, sanitize and validate the input, and insert the data into the appropriate database table.
5. Establish a database connection: Within the PHP script, create a connection to the MySQL database using the appropriate MySQL extension or library (e.g., mysqli, PDO). Provide the necessary credentials such as the host, username, password, and database name to establish the connection.
6. Retrieve form data: Access the form data submitted via the POST method using the $\_POST superglobal array in the PHP script. Retrieve the values entered by the user for each input field using their respective names ($\_POST['name'], $\_POST['email'], etc.).
7. Sanitize and validate input: Before saving the user information in the database, it is important to sanitize and validate the input to prevent potential security vulnerabilities like SQL injection. Use appropriate functions or techniques (e.g., prepared statements, parameter binding) to sanitize and validate the form data.
8. Insert data into the database: Construct an SQL query to insert the sanitized form data into the appropriate table in the MySQL database. Execute the query using the database connection established earlier.
9. Handle success or failure: Check if the database insertion was successful or if any errors occurred during the process. Display appropriate messages or redirect the user to a success page or an error page accordingly.
10. Close the database connection: After executing the query and performing necessary operations, close the database connection to free up resources.
11. Test the registration page: Run the registration page in a web browser and fill out the form with test data. Verify that the form submission works as expected and the user information is successfully saved in the database.

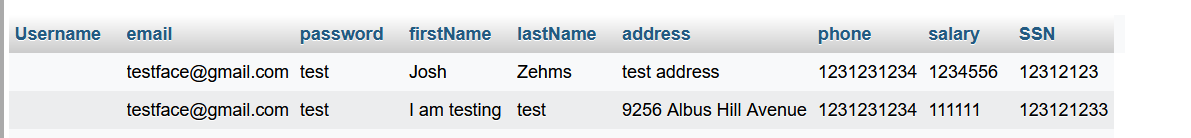
**Screenshots:**

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with low confidenceA screenshot of a computer

Description automatically generated with medium confidenceA picture containing text, screenshot, diagram, font

Description automatically generated

**Source code:**



**References:**

GeeksforGeeks. (2021). PHP | MySQL Database Introduction. https://www.geeksforgeeks.org/php-mysql-database-introduction/#

Nagaraj, P., Muneeswaran, V., Pavan Naidu, A. V. S. R., Shanmukh, N., Kumar, P. V., & Satyanarayana, G. S. (2023). Automated E-Commerce Price Comparison Website using PHP, XAMPP, MongoDB, Django, and Web Scrapping. *2023 International Conference on Computer Communication and Informatics (ICCCI), Computer Communication and Informatics (ICCCI), 2023 International Conference On*, 1–6. https://doi.org/10.1109/ICCCI56745.2023.10128573

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

Top of Form

Bottom of Form