**Experiment 9: Connect a PHP application to a MySQL DB**

**Aim:** Connect a PHP application to a MySQL database and perform CRUD operations

**Learning Objective:**

To establish a secure connection between a PHP application and a MySQL database, and perform Create, Read, Update, and Delete (CRUD) operations efficiently.

**Tools Used:**

* XAMPP
* PHP
* MySQL
* VS Code

**Theory:**

Connecting a PHP application to a MySQL database is a fundamental process in web development, allowing for the dynamic management of data. This theory outlines the key concepts involved:

#### 1. ****Database Connection****

Establishing a connection between PHP and MySQL is crucial for data interaction. PHP uses extensions such as **MySQLi** (MySQL Improved) and **PDO** (PHP Data Objects) to facilitate this connection. The connection requires specific parameters, including the database server's hostname, username, password, and database name.

#### 2. ****CRUD Operations****

CRUD refers to the four basic operations performed on data:

* **Create**: This operation involves inserting new records into a database table. It allows the application to add data, such as user information or product details.
* **Read**: The read operation retrieves existing data from the database. It is essential for displaying information, such as user profiles or product listings, within the application.
* **Update**: Updating existing records modifies data already stored in the database. This operation is necessary for editing user details or adjusting inventory levels.
* **Delete**: The delete operation removes records from the database. It is important for managing data lifecycle, such as deleting user accounts or removing discontinued products.

#### 3. ****Prepared Statements****

Prepared statements are a security feature that separates SQL logic from user input. By using prepared statements, applications can prevent SQL injection attacks, which occur when malicious users attempt to manipulate queries by injecting harmful data.

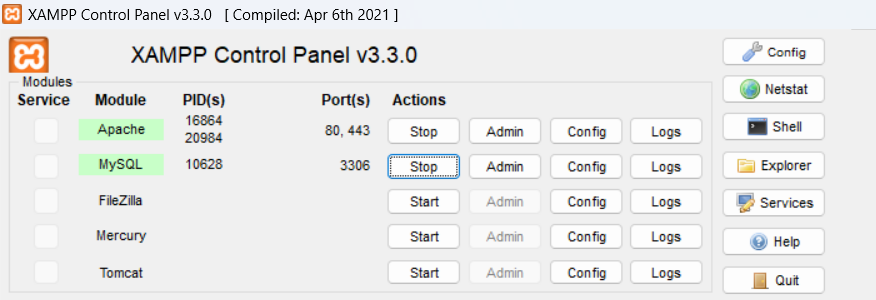
#### 4. ****Error Handling****

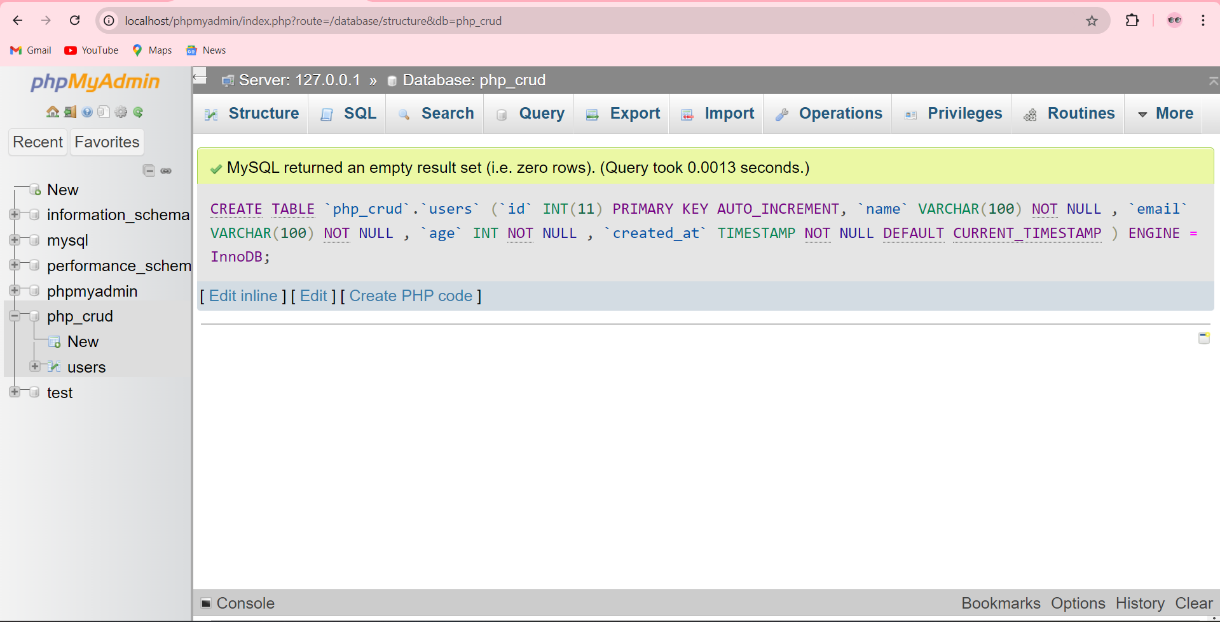
Proper error handling is vital for maintaining the reliability of the application. This includes managing connection failures, query errors, and exceptions. Effective error handling ensures that issues are logged and can be addressed without exposing sensitive information to users.

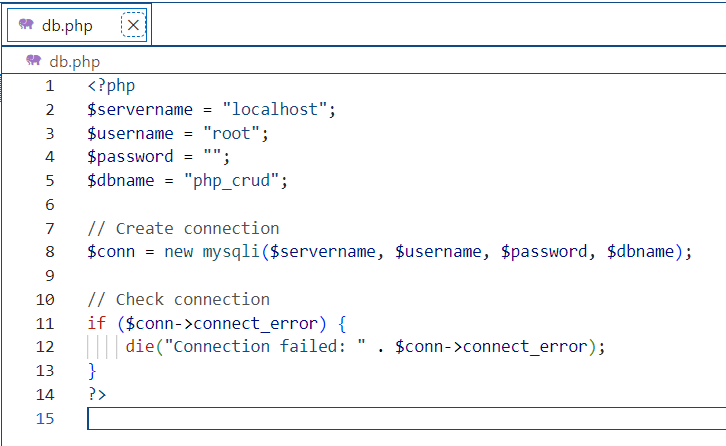
#### 5. ****Closing the Connection****

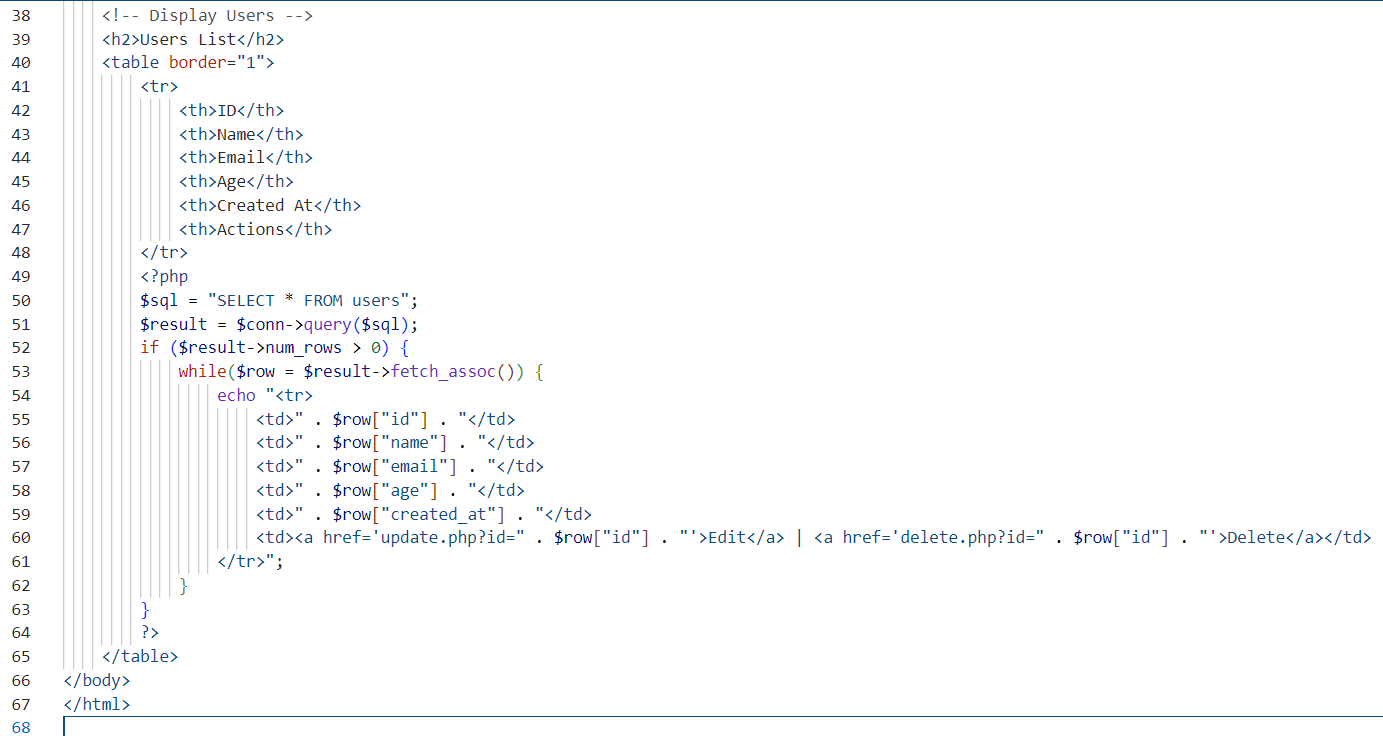
Once the necessary database operations are completed, it is important to close the connection. This action releases resources and prevents potential memory leaks, ensuring optimal performance of the application.

**Implementation/Code:**

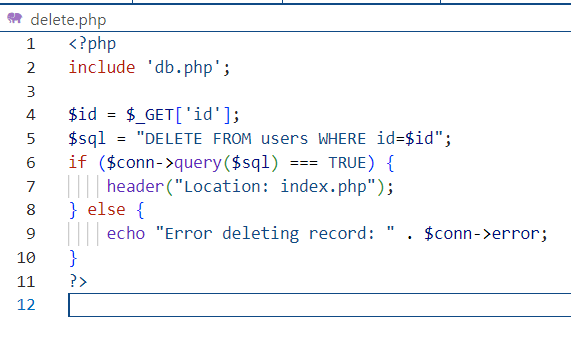


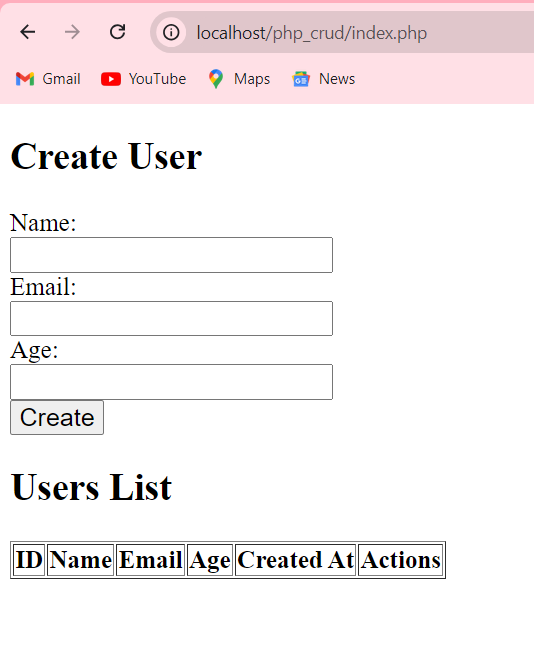






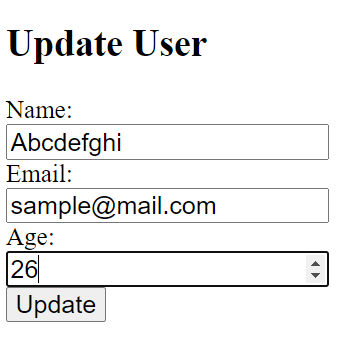
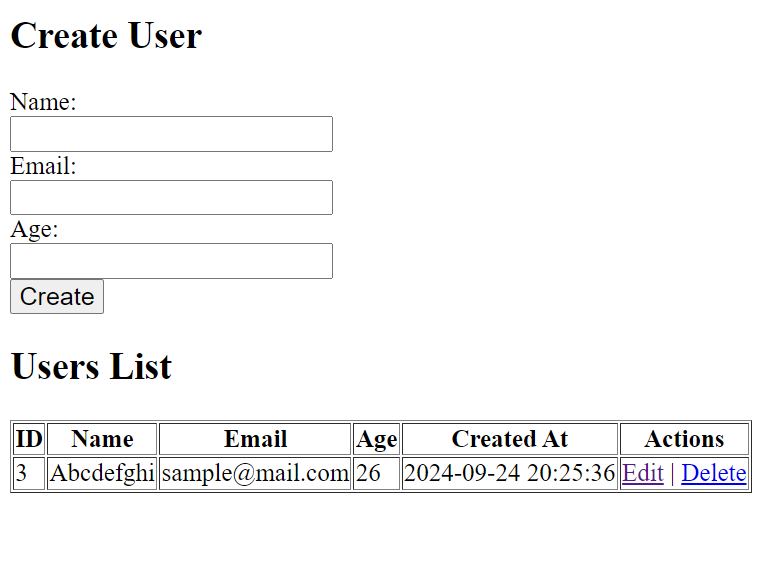




Create:

Read:



Update

Delete: