**CRUD Operations Using PHP and MySQL**

**Introduction**

CRUD stands for Create, Read, Update, and Delete. These are the fundamental operations for managing data in a database. PHP, a popular server-side scripting language, is often used in combination with MySQL, a relational database management system, to build dynamic web applications.

**Prerequisites**

1. Basic knowledge of PHP.
2. Understanding of HTML and CSS.
3. A working installation of PHP, MySQL, and a local server (e.g., XAMPP, WAMP, or MAMP).
4. A code editor (e.g., VS Code, Notepad++).

**Step 1: Setting Up the Database**

1. Open your MySQL database tool (e.g., phpMyAdmin).
2. Create a database named crud\_app.
3. Create a table named users with the following structure:

CREATE TABLE users (

-- Creates a new table named 'users'.

id INT AUTO\_INCREMENT PRIMARY KEY,

-- Defines an 'id' column as an integer, auto-incrementing, and the primary key for the table.

name VARCHAR(100) NOT NULL,

-- Defines a 'name' column as a variable-length string with a maximum of 100 characters, and it cannot be null.

email VARCHAR(100) NOT NULL,

-- Defines an 'email' column as a variable-length string with a maximum of 100 characters, and it cannot be null.

phone VARCHAR(15),

-- Defines a 'phone' column as a variable-length string with a maximum of 15 characters; it can be null.

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

-- Defines a 'created\_at' column to store a timestamp, with a default value of the current date and time.

);

**Step 2: Project Structure**

Organize your project as follows:

crud\_app/

|-- index.php // Displays records (Read)

|-- create.php // Adds new records (Create)

|-- update.php // Edits existing records (Update)

|-- delete.php // Deletes records (Delete)

|-- db.php // Database connection file

|-- styles.css // Optional: Styling

**Step 3: Database Connection**

Create a file named db.php for the database connection:

<?php

// Starts the PHP script.

$servername = "localhost";

// Stores the server name (usually 'localhost' for local development) in a variable.

$username = "root";

// Stores the database username (default is 'root' for many local servers) in a variable.

$password = "";

// Stores the database password (empty for local servers by default) in a variable.

$database = "crud\_app";

// Stores the name of the database to connect to in a variable.

// Create connection

$conn = new mysqli($servername, $username, $password, $database);

// Initializes a new MySQLi connection using the server name, username, password, and database name.

// Check connection

if ($conn->connect\_error) {

// Checks if there was an error in establishing the connection.

die("Connection failed: " . $conn->connect\_error);

// Terminates the script and displays an error message if the connection fails.

}

?>

// Ends the PHP script.

**Step 4: Create (Insert Data)**

Create a form in create.php:

<!DOCTYPE html>

<!-- Specifies the document type and version as HTML5. -->

<html>

<!-- Opens the HTML document. -->

<head>

<title>Create User</title>

<!-- Sets the title of the web page to "Create User". -->

</head>

<body>

<!-- Opens the body of the HTML document. -->

<h2>Create User</h2>

<!-- Displays a heading with the text "Create User". -->

<form action="" method="POST">

<!-- Creates a form with the POST method to send data to the server. -->

<label>Name:</label><br>

<!-- Displays a label for the name input field and breaks the line. -->

<input type="text" name="name" required><br>

<!-- Creates a text input for the name, marked as required, and breaks the line. -->

<label>Email:</label><br>

<!-- Displays a label for the email input field and breaks the line. -->

<input type="email" name="email" required><br>

<!-- Creates an email input for the email, marked as required, and breaks the line. -->

<label>Phone:</label><br>

<!-- Displays a label for the phone input field and breaks the line. -->

<input type="text" name="phone"><br>

<!-- Creates a text input for the phone number and breaks the line. -->

<button type="submit" name="submit">Create</button>

<!-- Adds a submit button with the label "Create". -->

</form>

<?php

include 'db.php';

// Includes the database connection file 'db.php'.

if (isset($\_POST['submit'])) {

// Checks if the form's submit button has been clicked.

$name = $\_POST['name'];

// Retrieves the 'name' value from the form input.

$email = $\_POST['email'];

// Retrieves the 'email' value from the form input.

$phone = $\_POST['phone'];

// Retrieves the 'phone' value from the form input.

$sql = "INSERT INTO users (name, email, phone) VALUES ('$name', '$email', '$phone')";

// Prepares an SQL query to insert the name, email, and phone into the 'users' table.

if ($conn->query($sql) === TRUE) {

// Executes the query and checks if it was successful.

echo "New record created successfully!";

// Displays a success message if the record is inserted successfully.

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

// Displays an error message if the query fails, showing the query and error details.

}

}

?>

</body>

<!-- Closes the body of the HTML document. -->

</html>

<!-- Closes the HTML document. -->

**Step 5: Read (Retrieve Data)**

Create index.php to display all records:

<!DOCTYPE html>

<!-- Specifies the document type and version as HTML5. -->

<html>

<!-- Opens the HTML document. -->

<head>

<title>Users</title>

<!-- Sets the title of the web page to "Users". -->

</head>

<body>

<!-- Opens the body of the HTML document. -->

<h2>Users List</h2>

<!-- Displays a heading with the text "Users List". -->

<table border="1">

<!-- Creates a table with a border width of 1. -->

<tr>

<!-- Creates a table row for the column headers. -->

<th>ID</th>

<!-- Defines the header for the "ID" column. -->

<th>Name</th>

<!-- Defines the header for the "Name" column. -->

<th>Email</th>

<!-- Defines the header for the "Email" column. -->

<th>Phone</th>

<!-- Defines the header for the "Phone" column. -->

<th>Actions</th>

<!-- Defines the header for the "Actions" column. -->

</tr>

<?php

include 'db.php';

// Includes the database connection file 'db.php'.

$sql = "SELECT \* FROM users";

// Prepares an SQL query to retrieve all records from the 'users' table.

$result = $conn->query($sql);

// Executes the query and stores the result.

if ($result->num\_rows > 0) {

// Checks if there are any records in the result.

while ($row = $result->fetch\_assoc()) {

// Loops through each record and fetches it as an associative array.

echo "<tr>

<td>{$row['id']}</td>

<!-- Displays the 'id' value of the current record in a table cell. -->

<td>{$row['name']}</td>

<!-- Displays the 'name' value of the current record in a table cell. -->

<td>{$row['email']}</td>

<!-- Displays the 'email' value of the current record in a table cell. -->

<td>{$row['phone']}</td>

<!-- Displays the 'phone' value of the current record in a table cell. -->

<td>

<a href='update.php?id={$row['id']}'>Edit</a> |

<!-- Adds an "Edit" link to update the current record, passing its 'id' as a query parameter. -->

<a href='delete.php?id={$row['id']}'>Delete</a>

<!-- Adds a "Delete" link to delete the current record, passing its 'id' as a query parameter. -->

</td>

</tr>";

}

} else {

echo "<tr><td colspan='5'>No records found</td></tr>";

// If no records are found, displays a single row with a message spanning all columns.

}

?>

</table>

<!-- Closes the table. -->

</body>

<!-- Closes the body of the HTML document. -->

</html>

<!-- Closes the HTML document. -->

**Step 6: Update (Edit Data)**

Create update.php to edit a record:

<!DOCTYPE html>

<!-- Specifies the document type and version as HTML5. -->

<html>

<!-- Opens the HTML document. -->

<head>

<title>Update User</title>

<!-- Sets the title of the web page to "Update User". -->

</head>

<body>

<!-- Opens the body of the HTML document. -->

<h2>Update User</h2>

<!-- Displays a heading with the text "Update User". -->

<?php

include 'db.php';

// Includes the database connection file 'db.php'.

if (isset($\_GET['id'])) {

// Checks if an 'id' is passed as a query parameter in the URL.

$id = $\_GET['id'];

// Retrieves the 'id' value from the query parameter.

$sql = "SELECT \* FROM users WHERE id = $id";

// Prepares an SQL query to fetch the record with the specified 'id'.

$result = $conn->query($sql);

// Executes the query and stores the result.

$row = $result->fetch\_assoc();

// Fetches the record as an associative array.

}

if (isset($\_POST['update'])) {

// Checks if the form is submitted by checking the 'update' button.

$name = $\_POST['name'];

// Retrieves the 'name' value from the form.

$email = $\_POST['email'];

// Retrieves the 'email' value from the form.

$phone = $\_POST['phone'];

// Retrieves the 'phone' value from the form.

$sql = "UPDATE users SET name = '$name', email = '$email', phone = '$phone' WHERE id = $id";

// Prepares an SQL query to update the record with the new values.

if ($conn->query($sql) === TRUE) {

// Executes the query and checks if it was successful.

echo "Record updated successfully!";

// Displays a success message if the query executed successfully.

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

// Displays an error message if the query failed.

}

}

?>

<form method="POST">

<!-- Opens a form with the POST method to send data to the server. -->

<label>Name:</label><br>

<!-- Displays a label for the "Name" field. -->

<input type="text" name="name" value="<?php echo $row['name']; ?>" required><br>

<!-- Creates a text input field pre-filled with the current "name" value. -->

<label>Email:</label><br>

<!-- Displays a label for the "Email" field. -->

<input type="email" name="email" value="<?php echo $row['email']; ?>" required><br>

<!-- Creates an email input field pre-filled with the current "email" value. -->

<label>Phone:</label><br>

<!-- Displays a label for the "Phone" field. -->

<input type="text" name="phone" value="<?php echo $row['phone']; ?>"><br>

<!-- Creates a text input field pre-filled with the current "phone" value. -->

<button type="submit" name="update">Update</button>

<!-- Creates a submit button labeled "Update" to submit the form. -->

</form>

</body>

<!-- Closes the body of the HTML document. -->

</html>

<!-- Closes the HTML document. -->

**Step 7: Delete (Remove Data)**

Create delete.php to delete a record:

<?php

include 'db.php';

// Includes the database connection file 'db.php' to establish a connection with the database.

if (isset($\_GET['id'])) {

// Checks if an 'id' is passed as a query parameter in the URL.

$id = $\_GET['id'];

// Retrieves the 'id' value from the query parameter to identify which record to delete.

$sql = "DELETE FROM users WHERE id = $id";

// Prepares an SQL query to delete the record with the specified 'id' from the 'users' table.

if ($conn->query($sql) === TRUE) {

// Executes the query and checks if it was successful.

echo "Record deleted successfully!";

// Displays a success message if the query executed successfully.

} else {

echo "Error deleting record: " . $conn->error;

// Displays an error message if the query failed.

}

}

?>

**Next Steps**

1. Add validation to forms to improve security.
2. Use prepared statements to prevent SQL injection.
3. Style the application using CSS.
4. Expand functionality, e.g., pagination, search, and filtering.

**Conclusion**

CRUD operations are essential for database-driven web applications. By mastering these operations with PHP and MySQL, you can build robust and dynamic applications efficiently.