

Laboratory 05

Registration Module

This module will walk you through the process of building a simple user registration system. You will learn how to handle form data, securely store it in a database, and then retrieve and display it. This is a fundamental exercise for anyone learning full-stack web development.

Objectives

By the end of this lab, you should be able to:

- **Create** a MySQL database and a users table.
 - **Design** a user registration form using HTML.
 - **Process** form submissions using PHP.
 - **Insert** user data into a database.
 - **Validate** and **sanitize** user input.
 - **Display** all registered users in an HTML table.
-

Laboratory Instruction: Prerequisites and File Structure

Before you begin, ensure you have **XAMPP** installed and running. Both the **Apache** and **MySQL** modules must be active.

Step 1: Database and Table Creation

1. Open your web browser and navigate to <http://localhost/phpmyadmin/>.
2. Click the **New** button on the left sidebar to create a new database. Name it `user_registration_db`.
3. Inside the `user_registration_db` database, create a new table. Name the table `users` and set the number of columns to 5.
4. Configure the columns as follows:
 - `id`: INT, PRIMARY KEY, A_I (Auto Increment)
 - `name`: VARCHAR, Length: 100

- password: VARCHAR, Length: 255 (We use a larger length for the hashed password)
- city_address: VARCHAR, Length: 255
- contact_number: VARCHAR, Length: 20

Step 2: File and Folder Setup

1. Navigate to your XAMPP installation directory (e.g., C:\xampp).
2. Open the htdocs folder.
3. Create a new folder inside htdocs and name it registration_lab.
4. Inside registration_lab, create the following three files:
 - register.php: This file will contain the HTML form and the PHP logic to handle the registration.
 - display_users.php: This file will fetch and display all registered users in a table.
 - style.css: This file will be used to style your HTML pages.

Your final folder structure should look like this:

C:\xampp\htdocs\

└─ registration_lab\

│ └─ register.php

│ └─ display_users.php

└─ style.css

Task 1: Creating the Registration Form and PHP Logic

This is the core of the module, where you'll create the form and the PHP script that processes it.

File: register.php

Open register.php in your code editor and add the following code:

PHP

```
<?php

// Check if the form was submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {

    // Database credentials

    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "user_registration_db";

    // Create connection

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

    // Check connection
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    // Sanitize and validate input
    $name = $conn->real_escape_string($_POST['name']);
    $password = $conn->real_escape_string($_POST['password']);
    $city_address = $conn->real_escape_string($_POST['city_address']);
    $contact_number = $conn->real_escape_string($_POST['contact_number']);

    // Hash the password for security
    $hashed_password = password_hash($password, PASSWORD_DEFAULT);
```

```

// SQL query to insert data into the 'users' table

$sql = "INSERT INTO users (name, password, city_address, contact_number)
VALUES (?, ?, ?, ?)";

// Prepare the statement to prevent SQL injection

$stmt = $conn->prepare($sql);

$stmt->bind_param("ssss", $name, $hashed_password, $city_address,
$contact_number);

// Execute the statement
if ($stmt->execute()) {
    $message = "Registration successful!";
} else {
    $message = "Error: " . $stmt->error;
}

$stmt->close();
$conn->close();
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User Registration</title>
    <link rel="stylesheet" href="style.css">
</head>

```

```
<body>
```

```
<div class="container">
```

```
<h1>User Registration</h1>
```

```
<?php if (isset($message)): ?>
```

```
<p class="message"><?= $message ?></p>
```

```
<?php endif; ?>
```

```
<form action="register.php" method="post">
```

```
<div class="form-group">
```

```
<label for="name">Name:</label>
```

```
<input type="text" id="name" name="name" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="password">Password:</label>
```

```
<input type="password" id="password" name="password" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="city_address">City Address:</label>
```

```
<input type="text" id="city_address" name="city_address" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="contact_number">Contact Number:</label>
```

```
<input type="text" id="contact_number" name="contact_number" required>
```

```
</div>
```

```
<button type="submit">Register</button>

</form>

<p class="view-link"><a href="display_users.php">View Registered Users</a></p>

</div>

</body>

</html>
```

- **Explanation:**

- `$_SERVER["REQUEST_METHOD"] == "POST"`: This checks if the page was accessed via a form submission.
- **Password Hashing**: We use `password_hash()` to securely hash the user's password before storing it. **Never store plain text passwords in a database.**
- **Prepared Statements**: The code uses **prepared statements** (`$stmt = $conn->prepare($sql);`). This is a critical security measure to prevent **SQL injection attacks**. The `?` placeholders are replaced with the actual data later.

Task 2: Displaying Registered Users

This module will fetch all users from the database and display them in a table.

File: `display_users.php`

Open `display_users.php` and add the following code:

PHP

```
<?php
```

```
// Database credentials
```

```
$servername = "localhost";
```

```
$username = "root";
```

```
$password = "";
$dbname = "user_registration_db";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// SQL query to fetch all users
$sql = "SELECT id, name, city_address, contact_number FROM users";
$result = $conn->query($sql);

?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Registered Users</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>

    <div class="container">
```

```
<h1>Registered Users</h1>
```

```
<?php
```

```
if ($result->num_rows > 0) {
```

```
    echo "<table>";
```

```
    echo "<thead><tr><th>ID</th><th>Name</th><th>City  
Address</th><th>Contact Number</th></tr></thead>";
```

```
    echo "<tbody>";
```

```
    while($row = $result->fetch_assoc()) {
```

```
        echo "<tr>";
```

```
        echo "<td>" . $row["id"] . "</td>";
```

```
        echo "<td>" . $row["name"] . "</td>";
```

```
        echo "<td>" . $row["city_address"] . "</td>";
```

```
        echo "<td>" . $row["contact_number"] . "</td>";
```

```
        echo "</tr>";
```

```
    }
```

```
    echo "</tbody>";
```

```
    echo "</table>";
```

```
} else {
```

```
    echo "<p>No users registered yet.</p>";
```

```
}
```

```
$conn->close();
```

```
?>
```


</div>

</body>

</html>

- **Explanation:**

- This script is similar to the data fetching module from the previous response.
- SELECT id, name, city_address, contact_number FROM users: This SQL query retrieves all users from the database. **Note that we do not select the password column** for security reasons.

Task 3: Styling the Pages with CSS

This file provides basic styling to make the registration form and the users table readable.

File: style.css

Open style.css and add the following code:

CSS

```
body {  
    font-family: Arial, sans-serif;  
    background-color: #f0f2f5;  
    color: #333;  
    display: flex;  
    justify-content: center;  
    align-items: center;  
    min-height: 100vh;  
}
```

```
.container {  
  width: 90%;  
  max-width: 700px;  
  margin: 20px auto;  
  background: #fff;  
  padding: 30px;  
  border-radius: 10px;  
  box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1);  
}
```

```
h1 {  
  text-align: center;  
  color: #007bff;  
  margin-bottom: 25px;  
}
```

```
.form-group {  
  margin-bottom: 15px;  
}
```

```
label {  
  display: block;  
  margin-bottom: 5px;  
  font-weight: bold;  
}
```

```
input[type="text"], input[type="password"] {  
    width: 100%;  
    padding: 10px;  
    border: 1px solid #ccc;  
    border-radius: 5px;  
}
```

```
button {  
    width: 100%;  
    padding: 12px;  
    background-color: #007bff;  
    color: white;  
    border: none;  
    border-radius: 5px;  
    font-size: 16px;  
    cursor: pointer;  
    transition: background-color 0.3s ease;  
}
```

```
button:hover {  
    background-color: #0056b3;  
}
```

```
.message {  
    text-align: center;  
    padding: 10px;
```

```
border-radius: 5px;
background-color: #d4edda;
color: #155724;
border: 1px solid #c3e6cb;
margin-bottom: 20px;
}
```

```
.view-link {
    text-align: center;
    margin-top: 20px;
}
```

```
.view-link a {
    color: #007bff;
    text-decoration: none;
    font-weight: bold;
}
```

```
/* Table styling */
```

```
table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 20px;
}
```

```
th, td {
```

```
padding: 12px 15px;

text-align: left;

border-bottom: 1px solid #ddd;

}
```

```
thead th {

    background-color: #007bff;

    color: white;

    font-weight: bold;

}
```

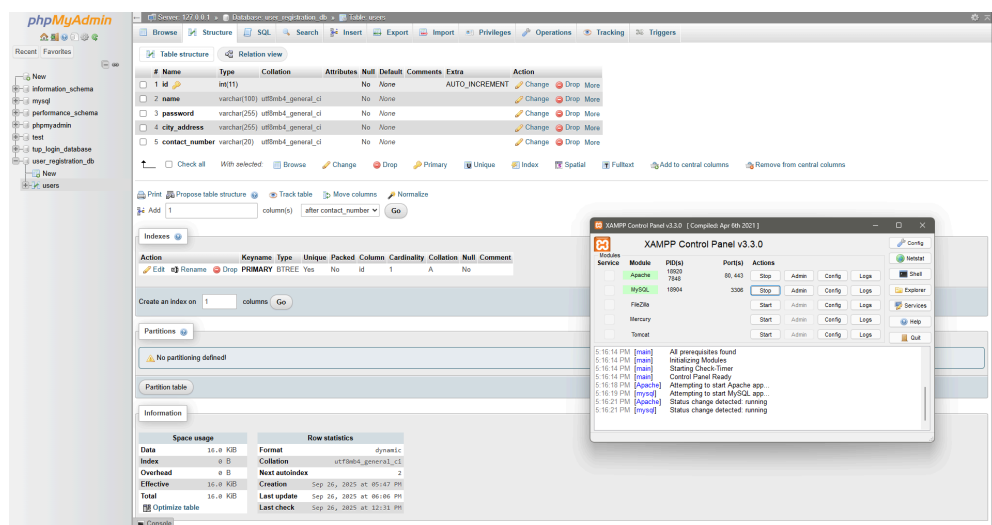
```
tbody tr:nth-child(even) {

    background-color: #f8f9fa;

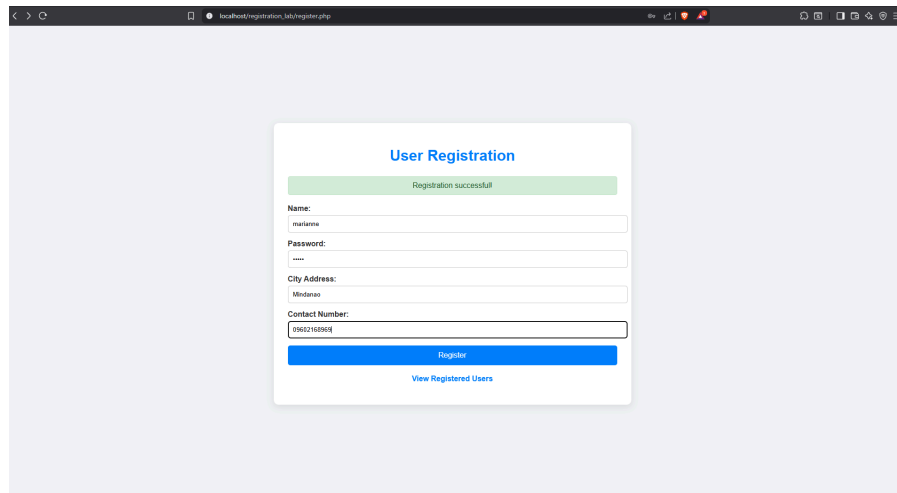
}
```

Screenshot of Output:

PHPMYADMIN SCREENSHOT



WEBSITE OUTPUT



User Registration

Registration successful!

Name:

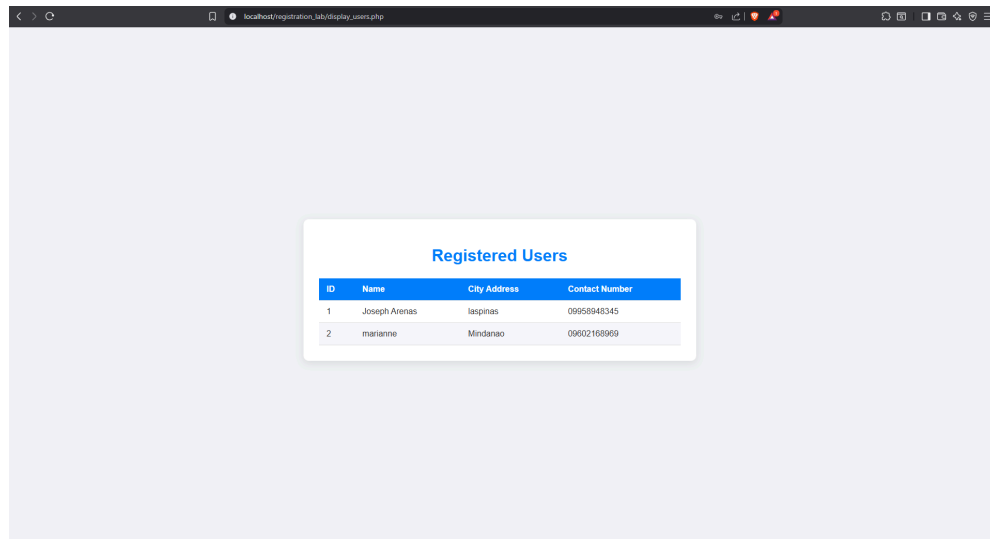
Password:

City Address:

Contact Number:

[Register](#)

[View Registered Users](#)



Registered Users

ID	Name	City Address	Contact Number
1	Joseph Arenas	Ispinas	09958948345
2	marianne	Mindanao	09602168969

Conclusion:

This laboratory exercise demonstrated how to build a basic user registration module by combining HTML, PHP, MySQL, and CSS. It highlighted essential skills such as handling form submissions, validating and securing user input, storing data in a database, and displaying it in a structured format. Overall, the activity provided a strong foundation for understanding the fundamentals of full-stack web development.