



POKHARA UNIVERSITY

LAB REPORT ON

Internet Technology II (4)

In partial fulfillment in the degree of

Bachelor of Computer Information System(BCSIT)

Shulabh Shrestha

P.U. Reg No. 2024-2-08-0747

P.U. Roll No. 24080115

Submitted to

Department of Computer Information System

Uniglobe College

New Baneshwar, Kathmandu

Nov 2025

GITHUB LINK: <https://github.com/Lumineskz/IT2-Assignments/>

1. Design a menu page (menu.php) with 3 links:

- ?page=home
- ?page=about
- ?page=contact

Using `$_GET['page']`, display the respective content dynamically (without multiple files).

2. Create a feedback form with fields: name, rating (1–5), and comment.

On submission:

- Show the submitted data.

Use `$_SERVER['REQUEST_METHOD']` to check if the form was posted.

Source Code:

```
<!-- menu.php -->

<?php

$page = isset($_GET['page']) ? $_GET['page'] : 'home';

?>

<!DOCTYPE html>

<html>

<head>

    <title>Menu Page</title>

</head>

<body style="font-family: Arial; margin: 40px;">

    <h2>Dynamic Menu Page</h2>

    <!-- Navigation Links -->

    <nav>

        <a href="?page=home">Home</a> |

        <a href="?page=about">About</a> |

        <a href="?page=contact">Contact</a>
```

```
</nav>

<hr>

<!-- Dynamic Page Content -->

<div>

    <?php
    if ($page == 'home') {
        echo "<h3>Welcome to the Home Page</h3><p>This is the home section.</p>";
    } elseif ($page == 'about') {
        echo "<h3>About Us</h3><p>This is the about section.</p>";
    } elseif ($page == 'contact') {
        echo "<h3>Contact Us</h3><p>This is the contact section.</p>";
    } else {
        echo "<h3>Page not found!</h3>";
    }
    ?>
</div>

<hr>

<h3>Feedback Form</h3>

<form method="post" action="">

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

    Rating (1–5):

    <input type="number" name="rating" min="1" max="5" required><br><br>

    Comment:<br>

    <textarea name="comment" rows="4" cols="40" required></textarea><br><br>

    <input type="submit" value="Submit Feedback">

</form>

<?php
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
```

```
$name = $_POST['name'];
$rating = $_POST['rating'];
$comment = $_POST['comment'];
echo "<hr><h3>Submitted Feedback:</h3>";
echo "Name: <strong>$name</strong><br>";
echo "Rating: <strong>$rating</strong><br>";
echo "Comment: <strong>$comment</strong><br><br>";
echo "<h4>Client & Server Info</h4>";
echo "Client IP: " . $_SERVER['REMOTE_ADDR'] . "<br>";
echo "Browser & OS: " . $_SERVER['HTTP_USER_AGENT'] . "<br>";
echo "Server Name: " . $_SERVER['SERVER_NAME'] . "<br>";
}
?>
</body>
</html>
```

Output:

Dynamic Menu Page

[Home](#) | [About](#) | [Contact](#)

Welcome to the Home Page

This is the home section.

Feedback Form

Name:

Rating (1-5):

Comment:

3. Build a PHP script that shows:

- The client's IP address (\$_SERVER['REMOTE_ADDR'])
- The browser and OS (\$_SERVER['HTTP_USER_AGENT'])
- The server name (\$_SERVER['SERVER_NAME'])

Source Code:

```
<?php

$clientIp = $_SERVER['REMOTE_ADDR'];

$userAgent = $_SERVER['HTTP_USER_AGENT'];

$serverName = $_SERVER['SERVER_NAME'];

$protocol = isset($_SERVER['HTTPS']) && $_SERVER['HTTPS'] === 'on' ? 'https' : 'http';
$serverUrl = $protocol . '://' . $serverName;

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Server and Client Environment Info</title>

</head>

<body>

    <h1>Environment Diagnostics</h1>

    <p>

        This page displays essential information about the client accessing this page and the server hosting it.

    </p>

    <hr>

    <h2>Client Information</h2>
```

```
<p>

    <strong>Client IP Address:</strong><br>

    <?php echo htmlspecialchars($clientIp); ?>

</p>

<p>

    <strong>Browser and OS (User Agent):</strong><br>

    <?php echo htmlspecialchars($userAgent); ?>

</p>

<hr>

<h2>Server Information</h2>

<p>

    <strong>Server Name:</strong><br>

    <?php echo htmlspecialchars($serverName); ?>

</p>

<div style="margin-top: 15px;">

    <p>Accessing this page via: <a href="<?php echo htmlspecialchars($serverUrl); ?>"><?php echo
htmlspecialchars($serverUrl); ?></a></p>

</div>

</body>

</html>
```

Output:

Environment Diagnostics

This page displays essential information about the client accessing this page and the server hosting it.

Client Information

Client IP Address:
::1

Browser and OS (User Agent):
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36

Server Information

Server Name:
localhost

Accessing this page via: <http://localhost>

4. Design a pizza order form that remembers the user's choices (preserve input) after form submission.

Inputs:

- **Size (radio): Small, Medium, Large**
- **Toppings (checkboxes): Cheese, Mushroom, Onion, Olive**
- **Crust Type (dropdown): Thin, Regular, Thick**

```
<?php
```

```
$errors = [];
```

```
$size = "";
```

```
$crust = "";
```

```
$toppings = [];
```

```
$submitted = false;
```

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {
```

```
    $submitted = true;
```

```
    // Retrieve inputs
```

```
    $size = $_POST["size"] ?? "";
```

```
    $crust = $_POST["crust"] ?? "";
```

```
    $toppings = $_POST["toppings"] ?? [];
```

```
    // Validation
```

```
    if (empty($size)) {
```

```
        $errors['size'] = "Please select a pizza size.";
```

```
    }
```

```
    if (empty($toppings)) {
```

```
        $errors['toppings'] = "Please select at least one topping.";
```

```
}
}
?>

<!DOCTYPE html>

<html>

<head>

  <title>Pizza Order</title>

  <style>

    .error { color: red; }

    body { font-family: Arial; margin: 20px; }

  </style>

</head>

<body>

  <h2>Pizza Order Form</h2>

  <form method="POST">

    <h3>Select Size:</h3>

    <label>

      <input type="radio" name="size" value="Small" <?=( $size == "Small" ) ? "checked" : "" ?>> Small

    </label><br>

    <label>

      <input type="radio" name="size" value="Medium" <?=( $size == "Medium" ) ? "checked" : "" ?>>
Medium

    </label><br>

    <label>

      <input type="radio" name="size" value="Large" <?=( $size == "Large" ) ? "checked" : "" ?>> Large
```


</label>

<?= \$errors['size'] ?? " ?>

<h3>Select Toppings:</h3>

<label>

<input type="checkbox" name="toppings[]" value="Cheese"

<?= in_array("Cheese", \$toppings) ? "checked" : "" ?>> Cheese

</label>

<label>

<input type="checkbox" name="toppings[]" value="Mushroom"

<?= in_array("Mushroom", \$toppings) ? "checked" : "" ?>> Mushroom

</label>

<label>

<input type="checkbox" name="toppings[]" value="Onion"

<?= in_array("Onion", \$toppings) ? "checked" : "" ?>> Onion

</label>

<label>

<input type="checkbox" name="toppings[]" value="Olive"

<?= in_array("Olive", \$toppings) ? "checked" : "" ?>> Olive

</label>

<?= \$errors['toppings'] ?? " ?>

<h3>Select Crust Type:</h3>

<select name="crust">

<option value="">-- Select Crust --</option>

<option value="Thin" <?= (\$crust == "Thin") ? "selected" : "" ?>>Thin</option>

<option value="Regular" <?= (\$crust == "Regular") ? "selected" : "" ?>>Regular</option>

```

        <option value="Thick" <?= ($crust == "Thick") ? "selected" : "" ?>>Thick</option>

</select><br><br>

<button type="submit">Order Pizza</button>

</form>

<?php if ($submitted && empty($errors)): ?>

    <h3>Your Order Summary</h3>

    <p><strong>Size:</strong> <?= $size ?></p>

    <p><strong>Crust:</strong> <?= $crust ?></p>

    <p><strong>Toppings:</strong> <?= implode(", ", $toppings) ?></p>

<?php endif; ?>

</body>

</html>

```

Output:

Pizza Order Form

Select Size:

- ☒ Small
☐ Medium
☐ Large

Select Toppings:

- ☒ Cheese
☐ Mushroom
☐ Onion
☒ Olive

Select Crust Type:

Regular ▼

Order Pizza

Your Order Summary

Size: Small

Crust: Regular

Toppings: Cheese, Olive

5. Create a Secure Registration Form File: register.php

Requirements:

Form Fields:

Full Name (text), Username (text), Email (email), Password (password), Confirm Password (password), Age (number), Submit button

```
<?php

// Initialize variables to store form data and errors

$fullName = $username = $email = $age = "";

$errors = array();


// Function to sanitize user input

function sanitize_input($data) {

    $data = trim($data);

    $data = stripslashes($data);

    $data = htmlspecialchars($data);

    return $data;

}


// Process the form submission

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    // --- 1. Sanitize and Validate Full Name ---

    $fullName = sanitize_input($_POST['fullName']);

    if (empty($fullName)) {

        $errors['fullName'] = "Full Name is required.";

    } elseif (strlen($fullName) < 3) {

        $errors['fullName'] = "Full Name must be at least 3 characters.";

    } elseif (!preg_match("/^[a-zA-Z]*$/", $fullName)) {
```

```
$errors['fullName'] = "Only letters and spaces allowed in Full Name.";
}

// --- 2. Sanitize and Validate Username ---

$username = sanitize_input($_POST['username']);
$usernamePattern = '/^[a-zA-Z0-9_]{5,15}$/';

if (empty($username)) {
    $errors['username'] = "Username is required.";
} elseif (!preg_match($usernamePattern, $username)) {
    $errors['username'] = "Username must be 5-15 characters and contain only alphanumeric
characters or underscore.";
}

// --- 3. Sanitize and Validate Email ---

$email = sanitize_input($_POST['email']);

if (empty($email)) {
    $errors['email'] = "Email is required.";
} elseif (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
    $errors['email'] = "Invalid email format.";
} elseif (strpos($email, ' ') !== false) {
    $errors['email'] = "Email must not contain spaces.";
}

// --- 4. Validate Password ---

$password = $_POST['password'];
$confirmPassword = $_POST['confirmPassword'];

$passwordPattern = '/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d){8,}$/';
```

```
if (empty($password)) {  
    $errors['password'] = "Password is required.";  
} elseif (!preg_match($passwordPattern, $password)) {  
    $errors['password'] = "Password must be min 8 characters and contain at least one uppercase,  
one lowercase, and one number.";  
}
```

```
// --- 5. Validate Confirm Password ---
```

```
if (empty($confirmPassword)) {  
    $errors['confirmPassword'] = "Confirm Password is required.";  
} elseif ($password !== $confirmPassword) {  
    $errors['confirmPassword'] = "Passwords do not match.";  
}
```

```
// --- 6. Sanitize and Validate Age ---
```

```
$age = sanitize_input($_POST['age']);  
if (empty($age)) {  
    $errors['age'] = "Age is required.";  
} elseif (!is_numeric($age)) {  
    $errors['age'] = "Age must be a number.";  
} elseif ($age < 18 || $age > 100) {  
    $errors['age'] = "Age must be between 18 and 100.";  
}
```

```
// --- 7. If no errors, process the registration ---
```

```
if (empty($errors)) {  
    // IMPORTANT: In a real-world scenario, you MUST hash the password before storing it.  
    // Example: $hashed_password = password_hash($password, PASSWORD_DEFAULT);
```

```
// This is where database insertion logic would go using prepared statements.
```

```
// For demonstration purposes:
```

```
echo "Registration Successful!";
```

```
// Clear variables after success
```

```
$fullName = $username = $email = $age = "";
```

```
} else {
```

```
    echo "Validation Errors Found";
```

```
}
```

```
}
```

```
?>
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <title>User Registration</title>
```

```
    <style>
```

```
        .error { color: red; font-size: 0.9em; margin-top: 5px; }
```

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

```
        label { display: block; font-weight: bold; }
```

```
    </style>
```

```
</head>
```

```
<body>
```

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

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

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

```
<label for="fullName">Full Name</label>
```

```
<input type="text" id="fullName" name="fullName" value="<?php echo  
htmlspecialchars($fullName); ?>" required>
```

```
<?php if (isset($errors['fullName'])) echo '<div class="error">' . $errors['fullName'] . '</div>'; ?>
```

```
</div>
```

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

```
<label for="username">Username</label>
```

```
<input type="text" id="username" name="username" value="<?php echo  
htmlspecialchars($username); ?>" required>
```

```
<?php if (isset($errors['username'])) echo '<div class="error">' . $errors['username'] .  
'</div>'; ?>
```

```
</div>
```

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

```
<label for="email">Email</label>
```

```
<input type="email" id="email" name="email" value="<?php echo  
htmlspecialchars($email); ?>" required>
```

```
<?php if (isset($errors['email'])) echo '<div class="error">' . $errors['email'] . '</div>'; ?>
```

```
</div>
```

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

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

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

<small>Min 8 characters, must contain at least one uppercase, one lowercase, and one
number.</small>

<?php if (isset($errors['password'])) echo '<div class="error">' . $errors['password'] .
'</div>'; ?>

</div>

<div class="form-group">

<label for="confirmPassword">Confirm Password</label>

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

<?php if (isset($errors['confirmPassword'])) echo '<div class="error">' .
$errors['confirmPassword'] . '</div>'; ?>

</div>

<div class="form-group">

<label for="age">Age</label>

<input type="number" id="age" name="age" value="<?php echo htmlspecialchars($age); ?>"
min="18" max="100" required>

<?php if (isset($errors['age'])) echo '<div class="error">' . $errors['age'] . '</div>'; ?>

</div>

<div class="form-group">

<input type="submit" value="Register">

</div>

</form>

</body>
```


</html>

Registration Successful!

User Registration

Full Name

Username

Email

Password

Min 8 characters, must contain at least one uppercase, one lowercase, and one number.

Confirm Password

Age

Register

6. Create a Simple Profile Picture Upload System

File: profile_upload.php

Requirements: Create a self-processing form with:

Form Fields: User Name (text input), Profile Picture (file input), Upload button

Validation Rules: Name is required (minimum 3 characters), File is required, Only image files allowed: JPG, JPEG, PNG, GIF, Maximum file size: 2MB, Check file upload error code

Processing: If validation passes:

- Create "uploads" folder if it doesn't exist
- Save file with original name
- Display success message
- Show the uploaded image (width: 200px)
- Display file information (name, size, type)

```
<?php
```

```
// Initialize variables
```

```
$errors = [];
```

```
$success = false;
```

```
$uploadedFilePath = "";
```

```
$fileInfo = [];
```

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {
```

```
    // -----
```

```
    // 1. Validate User Name
```

```
    // -----
```

```
    $name = trim($_POST['username']);
```

```
    if (empty($name)) {
```

```
        $errors[] = "Name is required";
```

```

} elseif (strlen($name) < 3) {

    $errors[] = "Name must be at least 3 characters long";

}

// -----

// 2. Validate File Upload

// -----

if (!isset($_FILES['profile_pic']) || $_FILES['profile_pic']['error'] == UPLOAD_ERR_NO_FILE) {

    $errors[] = "Profile picture is required";

} else {

    $file = $_FILES['profile_pic'];

    // Check upload errors

    if ($file['error'] != UPLOAD_ERR_OK) {

        $errors[] = "File upload error code: " . $file['error'];

    } else {

        $fileName = $file['name'];

        $fileTmp = $file['tmp_name'];

        $fileSize = $file['size'];

        $fileType = mime_content_type($fileTmp);

        // Allowed file types

        $allowedTypes = ['image/jpeg', 'image/jpg', 'image/png', 'image/gif'];

        if (!in_array($fileType, $allowedTypes)) {

            $errors[] = "Invalid file type. Only JPG, PNG, GIF allowed";

        }

    }

}

```

```

        // Check file size - max 2MB

        if ($fileSize > 2 * 1024 * 1024) {

            $errors[] = "File size exceeds 2MB limit";

        }

    }

}

// -----

// 3. If No Errors → Process Upload

// -----

if (empty($errors)) {

    // Create uploads folder if needed

    if (!is_dir("uploads")) {

        mkdir("uploads", 0777, true);

    }

    // Save file with original name

    $uploadedFilePath = "uploads/" . basename($fileName);

    if (move_uploaded_file($fileTmp, $uploadedFilePath)) {

        $success = true;

    }

    // Prepare file info

    $fileInfo = [

        "name" => $fileName,

```

```
        "size" => round($fileSize / 1024 / 1024, 2) . " MB",
        "type" => $fileType,
        "location" => $uploadedFilePath
    ];
} else {
    $errors[] = "Failed to save uploaded file.";
}
}
}
?>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
    <title>Profile Picture Upload</title>
```

```
</head>
```

```
<body>
```

```
<h2>Upload Profile Picture</h2>
```

```
<form action="" method="post" enctype="multipart/form-data">
```

```
    <label>User Name:</label><br>
```

```
    <input type="text" name="username" value=""><br><br>
```

```
    <label>Profile Picture:</label><br>
```

```
    <input type="file" name="profile_pic"><br><br>
```

<button type="submit">Upload</button>

</form>

<hr>

<?php if (!empty(\$errors)): ?>

<h3>Upload Errors:</h3>

<ul style="color:red;">

<?php foreach (\$errors as \$e): ?>

<?= \$e ?>

<?php endforeach; ?>

<?php endif; ?>

<?php if (\$success): ?>

<h2 style="color:green;">Profile Picture Uploaded Successfully!</h2>

<p>User Name: <?= htmlspecialchars(\$name) ?></p>

<h3>File Information:</h3>

File Name: <?= \$fileInfo['name'] ?>

File Size: <?= \$fileInfo['size'] ?>

File Type: <?= \$fileInfo['type'] ?>

Saved Location: <?= \$fileInfo['location'] ?>

<p></p>

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

```
</body>
```

```
</html>
```

Upload Profile Picture

User Name:

Profile Picture:

Choose File

No file chosen

Upload

Profile Picture Uploaded Successfully!

User Name: lumi

File Information:

- **File Name:** Screenshot 2025-11-03 012745.png
- **File Size:** 0.16 MB
- **File Type:** image/png
- **Saved Location:** uploads/Screenshot 2025-11-03 012745.png

