**Experiment 11**

**Title :** Upload various types of file from client side to server with validation

**Aim :** To make student understand how to upload a file on server using php

**Various types of file upload using PHP**

File uploads in PHP are a fundamental feature that enables users to send files from their local machines to a web server. This capability is commonly used in applications such as profile picture uploads, document submissions, and media uploads.

1. Setting Up the HTML Form

To facilitate file uploads, an HTML form must be configured correctly:

Use the method="POST" attribute to specify that the form data will be sent via POST.

Set the enctype="multipart/form-data" attribute, which is necessary for file uploads. This encoding type allows the form to send files along with other data.

<form action="upload.php" method="POST" enctype="multipart/form-data">

<input type="file" name="file" required>

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

</form>

2. Accessing Uploaded Files in PHP

PHP provides the $\_FILES superglobal array to manage uploaded files. This array contains information about the uploaded file, including:

$\_FILES['file']['name']: Original name of the uploaded file.

$\_FILES['file']['type']: MIME type of the uploaded file.

$\_FILES['file']['tmp\_name']: Temporary name of the file on the server.

$\_FILES['file']['error']: Error code associated with the upload (if any).

$\_FILES['file']['size']: Size of the uploaded file in bytes.

3. Handling File Uploads

When handling file uploads, follow these steps:

Check for Errors: Verify if the file was uploaded successfully using the error code.

Validate File Type: Ensure that the file is of an acceptable type (e.g., image, document) by checking its MIME type against a list of allowed types.

Check File Size: Validate that the file size does not exceed a predetermined limit to prevent excessively large uploads.

Move the File: Use the move\_uploaded\_file() function to transfer the file from the temporary directory to a permanent location on the server.

4. Security Considerations

File Type Validation: Always validate the MIME type and extension to avoid executing potentially harmful files.

File Size Limit: Implement limits on file sizes to prevent Denial-of-Service (DoS) attacks.

Use Unique Filenames: Avoid overwriting existing files by generating unique filenames (e.g., using timestamps or UUIDs).

Sanitize Input: Sanitize filenames and paths to prevent directory traversal attacks.

Access Control: Ensure that the upload directory is properly secured, ideally preventing direct access to uploaded files unless necessary.

Step 1: Create the HTML Form

Create a file named upload\_form.html with the following content

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>File Upload Form</title>

</head>

<body>

<h2>Upload a File</h2>

<form action="upload.php" method="POST" enctype="multipart/form-data">

<label for="file">Choose a file:</label>

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

<br><br>

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

</form>

</body>

</html>

Step 2: Create the PHP Script to Handle Uploads

Create a file named upload.php with the following content

<?php

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Check if the file was uploaded without errors

if (isset($\_FILES["file"]) && $\_FILES["file"]["error"] == 0) {

$fileName = $\_FILES["file"]["name"];

$fileType = $\_FILES["file"]["type"];

$fileSize = $\_FILES["file"]["size"];

// Specify allowed file types

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

// Validate file type

if (in\_array($fileType, $allowedTypes)) {

// Check file size (max 2MB)

if ($fileSize <= 2 \* 1024 \* 1024) {

// Specify the directory to save the uploaded file

$uploadDir = "uploads/";

// Create the uploads directory if it doesn't exist

if (!is\_dir($uploadDir)) {

mkdir($uploadDir, 0755, true);

}

// Move the uploaded file to the specified directory

$filePath = $uploadDir . basename($fileName);

if (move\_uploaded\_file($\_FILES["file"]["tmp\_name"], $filePath)) {

echo "File uploaded successfully: " . htmlspecialchars($fileName);

} else {

echo "Error moving the uploaded file.";

}

} else {

echo "File is too large. Maximum size is 2MB.";

}

} else {

echo "Invalid file type. Only JPEG, PNG, GIF, and PDF files are allowed.";

}

} else {

echo "Error uploading the file. Error code: " . $\_FILES["file"]["error"];

}

} else {

echo "Invalid request method.";

}

?>

**Conclusion:-**

Thus we have studied file uploading in PHP along with security checks