|  |  |
| --- | --- |
|  | Department of Information Technology, State Polytechnic of Malang  **Jobsheet-8: PHP - Form Upload, Cookies and Session**  **Web Design and Programming Courses**  Web Design and Programming Teaching Team  October 2024 |

Name : Evan Diantha Fafian

Class : SIB 2G

Absent : 09

NIM : 2341760163

**Topic**

* The concept of Form Upload with PHP and Jquery.
* Introduction to the concept of cookies.
* Introduction to session concepts.

**Objectives**

Students are expected to:

1. Students are able to create upload forms using PHP and jQuery.
2. Understand the basic concepts of cookies and its use.
3. Understand the basic concept of session and its use.

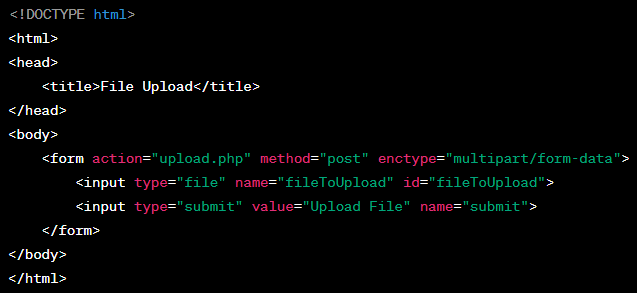
**Introduction**

**File Upload**

File upload in PHP is the process of uploading files from a user's computer to a web server. File uploads are useful in many cases such as when users want to upload a profile picture, document, or other media to a website. To do this, it can use the HTML element <input type="file> along with PHP to manage the upload process.

Here are the general steps to implement file uploads in PHP:

* Create an HTML form to upload a file:



* Create a PHP script (upload.php) to manage file uploads:



Above, we take some important steps:

* Create an HTML form with an element <input type="file> that allows users to select the file they want to upload.
* Set the form enctype attribute to "multipart/form-data" so that the file can be uploaded.
* When the user clicks the "Upload File" button, the form data will be sent to the upload.php.
* In upload.php, we check if the user has clicked the submit button ($\_POST["submit"]) and then specify the destination directory to save the file to.
* move\_uploaded\_file() is used to move files from a temp directory to a predefined destination directory.
* The corresponding message (success or failure) will be displayed to the user.

Make sure you have an uploads directory on your server, and give it the appropriate permissions so that PHP can store files there.

**Practical Section 1. Files**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file in the dasarWeb directory, naming it form\_upload.php. |
| 2 | Type into the form\_upload.php code file below. |
| 3 | Create a new file named upload.php that will be used for processing form\_upload.php. |
| 4 | Save the file, then open a browser and run localhost/dasarWeb/form\_upload.php.  Select a file and click the Submit button. Observe what happens and record your understanding. (Question No. 1)     * This HTML form allows users to upload files to the server, and PHP processes and stores the files in the destination directory, displaying a success or failure message. |
| 5 | Next, create a folder named uploads in the dasarWeb. Re-run localhost/dasarWeb/form\_upload.php.  Select a file and click the Submit button. Observe what happens and record your understanding. (Question No. 2)         * Once the uploads folder is created, the files uploaded via the form will be saved there, and a success or failure message is displayed depending on the folder permissions. |
| 6 | Change the contents of the upload.php file with the following code |
| 7 | Save the file, open a browser and run localhost/dasarWeb/form\_upload.php  Select a file with the .pdf extension or .docx. Click the Submit button.  Observe what happens and record your understanding.  (Question No. 3)       * The form will reject files with .pdf or .docx extensions, and display the message “The file is invalid or exceeds the maximum allowed size.” because those extensions are not included in the allowed list. |
| 8 | Next run localhost/dasarWeb/form\_upload.php again.  Select a file with the extension .jpg, .jpeg, .png, or .gif. Click the Submit button.  Observe what happens and record your understanding.  (Question No. 4)         * The form will accept files with allowed extensions and appropriate sizes, then save them in the uploads folder, displaying a success message if successful. |
| 9 | Add script from step 6 to display thumbnail image files with a width of 200 and height following the changes automatically after the image file is successfully uploaded.  Screen shoot the additional code. Explain your understanding after adding the program code.  (Question No. 5)         * Once the image file is successfully uploaded, additional code displays a thumbnail that is 200 pixels wide and auto-height to maintain the aspect ratio, providing immediate visual feedback to the user. |
| 10 | Next, change the contents of the upload.php file with the following code. |
| 11 | Save the file, open a browser and run localhost/dasarWeb/form\_upload.php  Select a file with an extension of .txt, .pdf, .doc, or .docx that is more than 5 MB in size. Click the Submit button. Observe what happens and record your understanding.  (Question No. 6)       * This happened because the uploaded file exceeded the set maximum size limit (5 MB), so the upload process failed, and the error message was displayed to the user as feedback. |
| 12 | Next run localhost/dasarWeb/form\_upload.php again.  Select a file with the extension .txt, .pdf, .doc, or .docx that is less than 3 MB in size. Click the Submit button. Observe what happens and record your understanding.  (Question No. 7)         * This indicates that the uploaded file meets the allowed extension and size criteria, so the file was successfully saved in the uploads folder and the upload process was successful. |

**Practical Section 2. Multi Upload File**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file named form\_multiupload.php. |
| 2 | Type the following code into form\_multiupload.php. |
| 3 | Create a new file named proses\_upload.php. Type the following code. |
| 4 | Save the file, open the browser and run localhost/dasarWeb/form\_multiupload.php. Select multiple files at once to upload. What do you understand from the script in the file? Record your understanding.  (Question No. 8)         * This script allows users to upload multiple document files at once to the server, check the existence of storage directories, and provide feedback on the success or failure of each file upload. |
| 5 | Change the code for multi upload of images.  Screen shoot the code changes and provide an explanation of the code.  (Question No. 9)         * This change adapts the previous code to allow image uploads, with validation to ensure only image files are allowed. Thus, users can upload multiple images at once with ease. |

**Practical Section 3. Upload Files with PHP and Jquery**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file named form\_upload\_ajax.php |
| 2 | Type the following code on the form\_upload\_ajax.php |
| 3 | Create a new file named upload.js. Write the following code. |
| 4 | Create a new file named upload\_ajax.php. Write the following code. |
| 5 | Save the file, then open a browser and run localhost/dasarWeb/form\_upload\_ajax.php.   * Upload a file in the form of an image.      * Uploading an image file: This should fail with the error message about allowed file extensions. * Upload a PDF file that is > 4 MB in size.        * Uploading a PDF file > 4MB: This should fail with the error message about file size. * Upload .docx files with a size of < 2 MB.          * Uploading a .docx file < 2MB: This should successfully upload and display the "File berhasil diunggah" message.   Observe what is happening and explain your understanding.  (Question No. 10) |
| 6 | Change the code to be able to do multi-upload image files.  Screenshot the code changes and explain the code.  (Question No. 11)         * This code allows users to upload multiple image files at once by validating the file extension and size. If the files are eligible, they will be moved to the specified directory, and the results of the upload process will be displayed on the page in real-time using AJAX. |

**Practical Section 4. Decorate Upload Files**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Modifiy the code of file form\_upload\_ajax.php in Practical Section 3 |
| 2 | Also modify the upload.js file as shown in the following code. |
| 3 | Create a new file named upload.css. Type the following code: |
| 4 | Save the file. Open a browser and run localhost/dasarWeb/form\_upload\_ajax.php.  What do you understand from the program code above? Record your understanding.  (Question No. 12)       * This program is designed to allow users to upload image files with a more interactive and responsive interface. Users can view the upload status directly, and the upload button is only activated when a file is selected. The program combines HTML, CSS, JavaScript, and PHP to provide a good user experience in the file upload process. |

**Introduction to *Cookies***

*A cookie* is a value that is sent and embedded *by a server* on the *client's computer. Cookies* are small files that are generally less than 100 bytes, so they won't affect *your browsing* speed. *Cookies* contain Description relating to the user. *Cookies* are useful for making it easier for users, such as "remembering" users every time they visit the same website.

The Description stored in *cookies* is simple data about user habits. A simple example is when a user visits a website that offers a view with English and Indonesian versions. If the user selects Indonesian, *the cookies* will automatically save the configuration. This configuration will be remembered, so that if the user visits the website again, the user does not need to select the language type again.

*Cookies* have an expiration date, meaning that any data stored on the user's computer could at some point be lost or destroyed. Based on their age*, cookies* are divided into two types, namely *session cookies* and *persistent cookies*. *Session cookies* are a type of *cookie* that is stored temporarily and will disappear when the user closes *the browser. These session cookies* are usually used in the "shopping cart" feature of online shopping sites. *Persistent cookies* are a type of *cookie* that is permanently stored on the computer until the user decides to delete it. *Persistent cookies* are generally used in the "*remember me*" feature when logging in.

PHP has functions that can be used to create and retrieve cookie values*.* Cookies must be declared before the page is displayed, which means they are written before the <html> tag. *Cookies* are created using the setcookie() function. Here is the basic syntax of the setcookie() function:

|  |
| --- |
| <?php  setcookie(name, value, expire);  ?> |

Description:

* name, is the name of  *the cookie* or variable that identifies the *cookie*
* value, contains the value stored in the *cookie*
* expire, is the period of time  *that cookies* will be stored on the computer

After knowing how to create *cookies,* then the value of the cookies that have been created can be retrieved using the $\_COOKIE variable. The following is the syntax for retrieving the value  *of cookies*:

|  |
| --- |
| <?php  $\_COOKIE[‘user’];  ?> |

**Practical Section 5. Creating *Cookies***

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file named cookiesCreate.php, then type the following code.  A close up of a text  Description automatically generated |
| 2 | Create a new file named cookiesCall.php, then type the following code.  A close-up of a sign  Description automatically generated |
| 3 | Open a *browser* and run the program code in step 2 by typing localhost/dasarWeb/cookiesCall.php |
| 4 | Observe and explain your observations  (Question No. 13)     * Since you haven't created the cookie yet, nothing will be displayed, or an undefined index error might occur because the user cookie doesn't exist yet. |
| 5 | Open a *browser* and run the program code step 1 by typing localhost/dasarWeb/cookiesCreate.php |
| 6 | Repeat step 3. |
| 7 | Observe and explain the results displayed  (Question No. 14)     * This will create a cookie named user with the value Polinema that will expire in one hour. |
| 8 | *Restart* your computer. |
| 9 | Once the computer is turned on, restart Apache on the laragon. |
| 10 | Open the same browser as before then repeat step 3. |
| 11 | Observe and explain the results displayed.  (Question No. 15)   * After restarting computer and Apache, the cookie should still exist if the same browser session is used, and you should still see Polinema displayed. However, if more than one hour has passed since the cookie was created, or if the browser session has been cleared, the cookie will expire, and the value will not be displayed. |

**Practical Section 6. Deleting the Value of *Cookies***

In this Practical Section, it will be discussed how to delete the value of *cookies.* If in the previous Practical Section the *cookies* were set with an *expiration* time()+3600, then to delete the cookie value is as follows:

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file with cookiesDel.php name, then type the following code.  A close up of a text  Description automatically generated |
| 2 | Open a *browser* and run the program code by typing localhost/dasarWeb/cookiesDel.php |
| 3 | Open a *browser* and run the program code from the part 5 Practical Section by typing  localhost/dasarWeb/cookiesCall.php |
| 4 | Observe and describe the results from steps 2 and 3, then draw conclusions.  (Question No. 16)     * This will set the user cookie with an expiration time in the past, effectively deleting it.      * This will attempt to access the user cookie. |

**Practical Section 7. Application of *Cookies* to the Shopping Cart Feature**

One example of the use of *cookies* is the "shopping cart" feature on the online store web application. The shopping cart contains the items that the user will buy. *Cookies* are used to remember the number of items selected by the user. Here is an example of the use  *of cookies* in the shopping cart feature:

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file with formBeli.html name, then type the following code.  A computer code with text  Description automatically generated |
| 2 | Create a new file named prosesBeli.php, then type the following code.  A computer code with black text  Description automatically generated |
| 3 | Create a new file named keranjangBelanja.php, then type the following code.  A screen shot of a computer code  Description automatically generated |
| 4 | Open *a browser* and run the program code step 3 by typing localhost/dasarWeb/keranjangBelanja.php |
| 5 | Observe and explain the results displayed.  (Question No. 17)     * Since no cookie is set yet, the page will either show no data or an error indicating the cookies do not exist. |
| 6 | Run the program code step 1 by typing localhost/dasarWeb/formBeli.html |
| 7 | Fill in the number of novels and textbooks you want to buy and then click the "submit" button. |
| 8 | Observe and explain the results displayed.  (Question No. 18)   * After submitting the form, the prosesBeli.php script sets the cookies for beliNovel and beliBuku with the respective values and redirects to keranjangBelanja.php. The keranjangBelanja.php will then display the number of novels and textbooks based on the set cookies. |
| 9 | Close *the browser* then reopen *the browser* then re-run the program code step 3 by typing localhost/dasarWeb/keranjangBelanja.php |
| 10 | Observe and explain the results displayed.  (Question No. 19)   * If the cookies haven't expired and the browser hasn't been cleared, the values for beliNovel and beliBuku should still be available and displayed. |

**Session Introduction**

*Session* is one of the facilities in PHP that is used to store data as a result of being stored in variables, so that the data can be accessed by the user as long as the *session*  variable is not emptied or deleted. *Sessions* are almost the same in principle as *cookies,* the difference is the position of the data storage*.* If *cookies* store data on the *client,* the session *data* will be stored on the server side*.* So *sessions* are relatively safer to use to store confidential value variables such as *usernames* and *passwords* at the time of login. Examples of using *sessions* include storing valid login Description in only one session and storing records of ordering goods in the *e-commerce*/online transaction system. The following table describes the comparison of *cookies* and *sessions*:

Comparison *of Cookies* and *Sessions*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Storing Description** | **Storage Location** | **Can be summoned** | **Global** | **Can be called up at another time** | **Browser Restricted** | **User editable** |
| *Cookies* | Yes | *Client* | Yes | Yes | Yes | Yes | Yes |
| *Session* | Yes | *Server* | Yes | Yes | Not | Not | Not |

To create *a session* in PHP the session\_start() function is used. Here is the basic syntax for starting *a session*:

|  |
| --- |
| <?php  session\_start();  ?> |

Once *the session* starts, the *session*  variable can be used. The variable in question is the PHP global variable $\_SESSION. Here's the syntax of using the $\_SESSION global variable:

|  |
| --- |
| <?php  $\_SESSION['nameVariable'];  ?> |

**Practical Section 8. Creating a *Session***

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file named sessionCreate.php, then type the following code.  A screen shot of a computer code  Description automatically generated |
| 2 | Create a new file named sessionCall.php, then type the following code.  A screenshot of a computer code  Description automatically generated |
| 3 | Open a *browser* and run the program code in step 2 by typing localhost/dasarWeb/sessionCall.php   * Before running sessionCreate.php, no session variables are set, so you might see errors or nothing displayed for the session variables. |
| 4 | Open a *browser* and run the program code step 1 by typing localhost/dasarWeb/sessionCreate.php   * This will start a session and set session variables favcolor to "green" and favanimal to "cat". |
| 5 | Repeat step 3 |
| 6 | Observe and explain the results displayed  (Question No. 20)   * This time, the page will display "Favorite color is green" and "Favorite animal is cat", confirming that the session variables were set and retrieved successfully. |

**Practical Section 9. Removing Session Values**

PHP provides a session\_destroy() function that can be used to delete *sessions*.

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file named sessionDel.php, then type the following code.  A white screen with black text  Description automatically generated |
| 2 | Open a *browser* and run the program code by typing localhost/dasarWeb/sessionDel.php   * After running sessionDel.php, a message will be displayed saying "All session variables are now removed, and the session is destroyed." |
| 3 | Open a *browser* and run the program code from the Practical Section 8 by typing localhost/dasarWeb/sessionCall.php   * When run sessionCall.php after destroying the session, the session variables favcolor and favanimal will no longer exist. As a result, you might see errors or nothing displayed since the session data has been removed. |
| 4 | Observe and explain the results displayed.  (Question No. 21)     * Using session\_unset() removes all session variables. * Using session\_destroy() ends the session and removes all session data. * After destroying the session, attempting to access previously set session variables will result in them being unavailable, confirming the session and its data have been cleared. |

**Practical Section 10. Implementation of *Session* on the Login Feature**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | Create a new file named sessionLoginForm.html, then type the following code |
| 2 | Create a file named sessionLoginProcess.php, then type the following code. |
| 3 | Create a file named homeSession.php, then type the following code. |
| 4 | Create a file named sessionLogout.php, then type the following code. |
| 5 | Open a *browser* and run the program code by typing localhost/dasarWeb/sessionLoginForm.html |
| 6 | Log in using your email username and password "0000". |
| 7 | Observe and explain the results displayed  (Question No. 22)     * The login will fail, and the message "Gagal login. Silahkan login lagi Halaman Login" will be displayed. |
| 8 | Re-run the program code by typing localhost/dasarWeb/sessionLoginForm.html  Log in using the username "admin" and password "1234". |
| 9 | Observe and explain the results displayed  (Question No. 23)     * The login will succeed, and the message "Anda berhasil login. Silahkan menuju Halaman Home" will be displayed. Clicking the link will take you to homeSession.php, which will display a welcome message with the username and a logout link. |
| 10 | Describe the sequence of the process from login to logout (also mention the order in which the files are processed)  (Question No. 24)  User visits sessionLoginForm.html:   * The login form is displayed to the user.   User submits the form:   * Action: Form data is sent to sessionLoginProcess.php.   sessionLoginProcess.php:   * Check Credentials: Compares submitted username and password. * Session Creation: If credentials match ("admin" and "1234"), a session is started, and session variables username and status are set. * Redirection: User is directed to homeSession.php. * Failed Login: If credentials do not match, an error message is displayed with a link back to the login form.   User visits homeSession.php:   * Session Check: Checks if the session status is 'login'. * Display: Shows a welcome message with the username if logged in. If not, prompts to log in.   User clicks on Logout link:   * Action: Directs to sessionLogout.php.   sessionLogout.php:   * Session Destroy: Ends the session and destroys session data. * Confirmation: Displays a message confirming the logout. |