

File System Explorer documentation

\$ Session variables

We have defined several session variables to develop the application.

- `$_SESSION["basePath"] = getcwd ()`
- `$_SESSION["currentPath"] = "root"`
- `$_SESSION["currentDirectories"] = array ()`
- `$_SESSION["searchFiles"] = array ()`
- `$_SESSION["matchedFiles"] = array ()`
- `$_SESSION["isSearching"] = false`
- `$_SESSION["searchText"] = " "`
- `$_SESSION["deleteSearch"] = false`

Steps:

File handling





Create the file system layout with some data. The data is:

- Type
- File name
- Size
- Creation & Modification date

To get this information we created a function that gets a **path** and a **file name** as parameters and **returns an associative array with all keys and values we needed to display from a file**. We used `filectime()` and `filemtime()` to get creation and modification dates and the resultant array looks like this:

```
$fileArray = array("icon" => $fIcon, "name" => $fName, "rawType" => $fMimeType,
"type" => $fType, "path" => $fPath, "size" => $fSize, "creation" => $fCreation,
"modification" => $fModification);
```

Image:

Type	Name	Size	Creation	Modification
	adios	64 B	29/06/21	29/06/21
	card.png	76.74 KB	01/07/21	29/06/21
	hola	224 B	01/07/21	01/07/21
	old-folder	64 B	01/07/21	29/06/21

Functionality

✓ Create folder

We use the session variable called `$_SESSION["currentDirectories"]` plus get the current root with `$_SESSION["currentPath"]`.

We handle the scenario to avoid duplicated folders and handle as well a recursive function to create "-copy" folder names.

Image:



224 B 01/07/21 01/07/21

Add new folder X

Folder name

Insert name

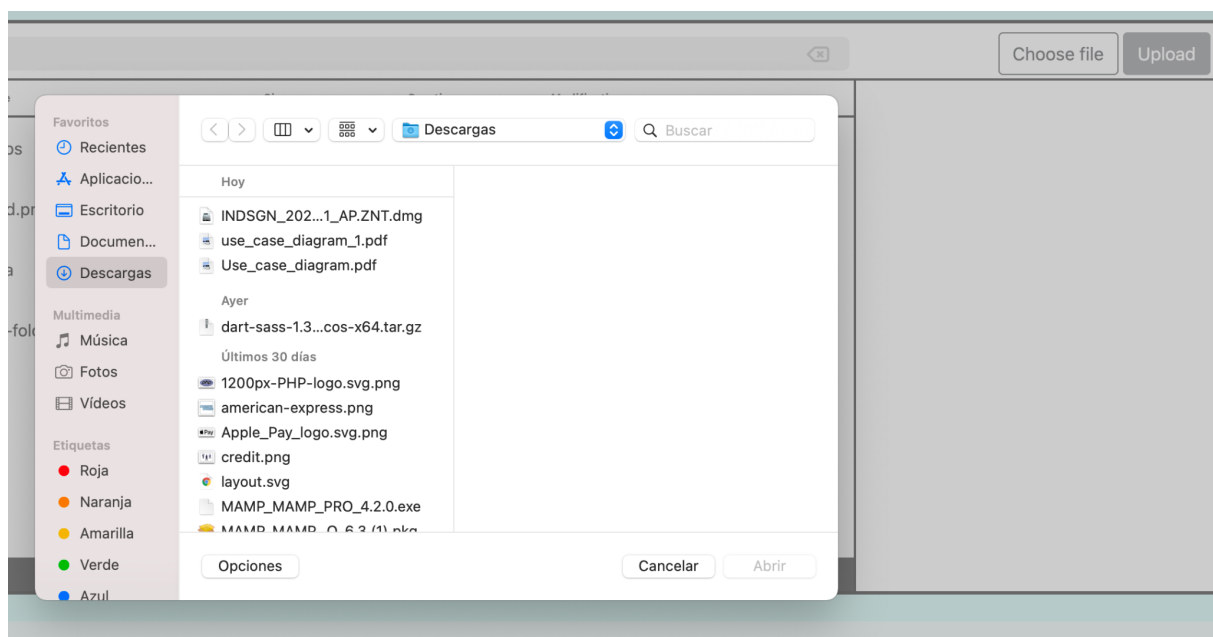
Close Add folder

✓ Upload folder

We needed to develop this functionality the session variable `$_SESSION["currentPath"]` and the superglobal variable `$_FILES` twice to target the correct directory.

Also, we had to handle only uploading if the file doesn't exist yet. Some more details, the user only can click the **upload button** when they have clicked the **Choose file button** first.

Image:

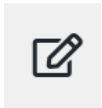


✓ Edit or rename

We use the session variable `$_SESSION["basePath"]`. In this functionality case have to pay attention in two key variables such as `$oldName` and `$newName`. These variables are pretty much important to run the **rename function**.

More important things in this functionality folder inside like how to handle the search bar functionality when a file or folder is renamed. In this case we need more session variables like `$_SESSION["isSearching"]` and `$_SESSION["searchText"]`.

Image:



New file name

File name

old-folder

Close

Rename

✓ Delete

To develop this functionality we use the session variable `$_SESSION["basePath"]` and two different variables to get some file info such as `$nameToDelete` and `$typeToDelete` where we obtain `filePath` and `fileType` throw `$_GET` method.

We handle two functions, first of all just creating a function to delete folders and files, and a second one a recursive function to delete all folders and files inside a folder.

Like edit or rename functionality we need to handle the search bar functionality when a file or folder is renamed. In this case we need more session variables like `$_SESSION["isSearching"]` and `$_SESSION["searchText"]`.

Image:



before:

📁	adios	64 B	29/06/21	29/06/21	
🖼️	card.png	76.74 KB	01/07/21	29/06/21	✎ 🗑️
📁	hola	224 B	01/07/21	01/07/21	
📁	old-folder	64 B	01/07/21	29/06/21	

After:

📁	adios	64 B	29/06/21	29/06/21	
📁	hola	224 B	01/07/21	01/07/21	✎ 🗑️
📁	old-folder	64 B	01/07/21	29/06/21	






✓ Search

To this functionality we need two session variables **\$_SESSION["currentPath"]** and **\$_SESSION["searchFiles"]**. Created a recursive function to search all files from the current path.


In another php file with the session variable **\$_SESSION["matchedFiles"]** and a **\$_POST variable** we manage how to search variables and a filtered array with all found files. Lastly, we handle how to reassign the default values to two session variables: **\$_SESSION["matchedFiles"]** and **\$_SESSION["searchText"]**

Image:

Before:

Search files				
Type	Name	Size	Creation	Modification
	Apple_Pay_logo.svg	16.07 KB	30/06/21	29/06/21
	Mastercard-Logo.png	20.96 KB	01/07/21	01/07/21
	javascript-dom-1-master.zip	1.64 MB	29/06/21	29/06/21
	new-folder	64 B	01/07/21	01/07/21
	sacsa.fit	46.95 KB	01/07/21	29/06/21

After:

java				
Type	Name	Size	Creation	Modification
	javascript-dom-1-master.zip	1.64 MB	29/06/21	29/06/21

More value to user

👉 Delete button in search bar

This is a kind of button that allows the user to delete the searching field when they are searching for some file name, file folder or exact extension. This functionality allows the users to reset the text if they fail typing in the search bar.

To develop this button we needed to use the session variable `$_SESSION["deleteSearch"]` within a `<a>` link element to redirect passing this variable value to **true** and reset the displayed files in the current path.






Image:



Before:

java					
Type	Name	Size	Creation	Modification	
	javascript-dom-1-master.zip	1.64 MB	29/06/21	29/06/21	

After:

Search files					
Type	Name	Size	Creation	Modification	
	Apple_Pay_logo.svg	16.07 KB	30/06/21	29/06/21	
	Mastercard-Logo.png	20.96 KB	01/07/21	01/07/21	
	javascript-dom-1-master.zip	1.64 MB	29/06/21	29/06/21	
	new-folder	64 B	01/07/21	01/07/21	
	sacsa.fit	46.95 KB	01/07/21	29/06/21	

👉 File preview and open

This functionality allows users to view a preview when they click to any file. This preview appears on the right side of the application. Right side contains some info about the file such as **name**, **path**, **type**, **size**, **creation date** and **modification date** as well.

Lastly, when the user clicks a button named **preview** will see the file in another tab. In summary the user will see the file content.

To develop file preview and open functionality we needed another php file that contains information about the file as we said before. After that, we used a **\$_GET variable** with **fileName** and **filePath** and display a preview icon and de main information with several echo's.

Image:



Preview

Name	Mastercard-Logo.png
Path	root/hola/Mastercard-Logo.png
Type	png
Size	20.96 KB
Creation	01/07/21
Modification	01/07/21

👉 Navigation

In order to navigate through all directories and path, the application allows us to do so using an always updating `$_SESSION["currentPath"]` to make a `scandir()` of the current path we are in.

Also, the user has both a **sidebar with a tree structure** and a **bottom path with breadcrumbs**. All of them consist of `<a>` pointing at the `updatePath.php` file that gets URL variables and updates `$_SESSION["currentPath"]` to call `scandir()` again.

Bottom navigation allows the user to navigate from the central block of the application, a more visually and clearly way to do it so the user can know where it is all the time. This functionality doesn't exist in Google Drive or Dropbox.

To develop the **sidebar structure** we both create an `<a>` pointing at the parent directory and an `<a>` for each child directory pointing at their respective paths.

To develop root navigation we needed to use the session variable `$_SESSION["currentPath"]` and two extra variables called `$totalPath` and `$pathCount`. After all, we create a foreach loop to display the information. First of all, display the first item in a single item path, second one, first item, third one, between items and finally the last item.

Image of the sidebar navigation:

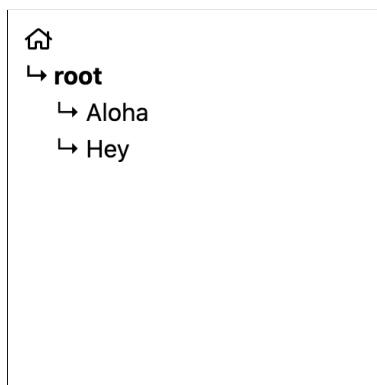


Image of bottom path navigation:



👉 Input name when the edit button is clicked and validation

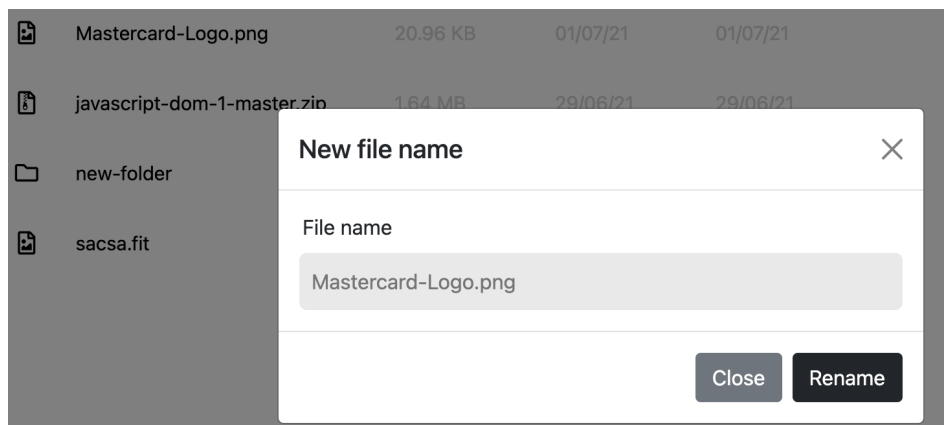
It's A simple functionality but allows the users to know the file name that they want to edit or rename.

The users cannot rename the current file with the same name, so, we handle this situation with a validation. When the old file name matches with the new file name, the app displays a warning alert to inform the user that they can't use the same name.

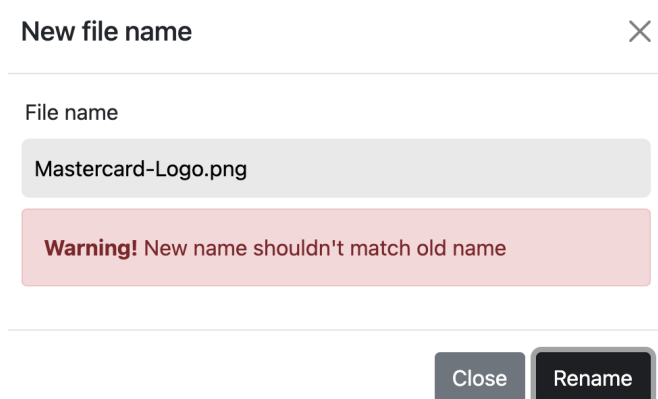
To develop both functionalities we use **Jquery** language. To the first functionality we use a single line of code that matches the placeholder attribute value.

To the second one, validation input, we create a Jquery function that matches the old name with the new name and we display the information, and save the new file name.

Image, first functionality:



Image, second functionality:



👉 Drag and drop

This functionality allows users to drag and drop upload files with a single movement. Also, you can click inside the box to choose a file to upload. This box to drag and drop is situated on the bottom side of the application.

To develop this functionality we help us to a **Dropzone** library, we create a div and choose a route to another php file that handles the uploading.

Image:

