# CUSTOM CHANGES SUMMARY

This document summarizes all custom changes applied to the plugin in this workspace.

## 1) PREVENT EMPTY YEAR/MONTH FOLDERS

Problem: Uploads created empty folders like ‘2026/02‘ during imports.

Fix: Avoid creating year/month folders unless a file is actually copied into uploads.

Where:
- ‘Plugin::handle_import_file()‘ in class.add-from-server.php

Notes:
- ‘wp_upload_dir( $time, false )‘ is used when only reading upload paths.
- Actual folder creation happens only when copying files into uploads.

---

## 2) DATE LOGIC HARDENED TO AVOID INVALID PATHS

Problem: Files were ending up in invalid folders such as ‘1770/89‘ or ‘1640/95‘.

Fixes:
- Base ‘$time‘ on file modification time when possible.
- Clamp invalid/extreme timestamps (before 1970 or too far in future) back to current time.
- If a ‘YYYY/MM‘ segment exists in the source path, only use it when the year/month is valid.

Where:
- ‘Plugin::handle_import_file()‘ in class.add-from-server.php

---

## 3) FORCED IMPORTS INTO A PLUGIN-SPECIFIC SUBFOLDER

Goal: All imports should go into a single plugin folder, not year/month directories.

Implementation:
- Added helper ‘Plugin::get_import_uploads_dir()‘ to build a prefixed uploads subdir.
- Default prefix: ‘uploads/add-from-server-reloaded/‘.
- All copied/moved imports now go into this prefixed folder.

Where:
- ‘Plugin::get_import_uploads_dir()‘ in class.add-from-server.php

- ‘Plugin::handle_import_file()‘ uses the helper for destination paths.

Filter:
- ‘afsrreloaded_upload_subdir‘ controls the prefix (set to empty string to disable).

---

4) MOVE FILES ALREADY INSIDE UPLOADS INTO THE PLUGIN FOLDER

Goal: Even files that were already in uploads should end up in the plugin subfolder.

Implementation:
- If a file is already in uploads but not in the plugin subfolder, it is moved (rename or copy+unlink fallback).

Where:
- ‘Plugin::handle_import_file()‘ in class.add-from-server.php

---

5) FOLDER IMPORT SKIPS RESTRICTED FILES AND SHOWS A SUMMARY

Goal: When importing a folder, restricted files should be skipped and one summary message shown (not one error per file).

Implementation:
- Folder scan collects restricted files and omits them from the import list.
- One summary notice shows the number of skipped files and all restricted extensions.

Where:
- ‘Plugin::handle_imports()‘ in class.add-from-server.php
- ‘Plugin::get_files_from_folder()‘ in class.add-from-server.php
- New helper ‘Plugin::get_restricted_extensions()‘ and updated ‘Plugin::is_restricted_file()‘.

Message Example:
- â 16 files were skipped for security reasons. Some files in the selected folders were not imported because their file types are not allowed: PHP, PHTML, PHPS, PHT, PHAR, EXE, SH, BAT, CMD.â

---

6) RESTRICTED EXTENSIONS LIST CENTRALIZED

Goal: Ensure the notice lists all restricted types consistently.

Implementation:

- Added ‘get_restricted_extensions()‘ to centralize the list.
- Summary message uses this list.

Current restricted extensions:
‘php‘, ‘phtml‘, ‘phps‘, ‘pht‘, ‘phar‘, ‘exe‘, ‘sh‘, ‘bat‘, ‘cmd‘

---

7) DEFAULT PREFIX NAME UPDATED

Change: Default uploads subdir prefix changed from ‘/afsrreloaded‘ to ‘/add-from-server-reloaded‘.

Where:
- ‘Plugin::get_import_uploads_dir()‘ in class.add-from-server.php

---

8) OUTPUT FORMATTING FOR FOLDER VS FILE SELECTION

Folder import output:
- Shows imported count.
- Shows â {FolderName} folder uploaded.â
- Shows duplicates count: â X files already exist in Media Library.â
- Shows skipped count and restricted extensions list.

Specific file selection output:
- Lists each duplicate as â filename: File already exists. View in Media Libraryâ .
- Lists each restricted file as â filename: This file was not imported due to security restrictions.â

---

FILES CHANGED

- class.add-from-server.php