**Cài đặt TinyMCE**

* Bước 1: Cài đặt thông qua package manager

npm install tinymce

OR

yarn add tinymce

* Bước 2: Sử dụng webpack mix

mix.js([

'resources/js/app.js',

'node\_modules/tinymce/tinymce.min.js',

], 'public/js/app.js')

.sass('resources/sass/app.scss', 'public/css');

mix.copyDirectory('node\_modules/tinymce/plugins', 'public/js/tinymce/plugins');

mix.copyDirectory('node\_modules/tinymce/themes', 'public/js/tinymce/themes');

mix.copyDirectory('node\_modules/tinymce/icons', 'public/js/tinymce/icons');

mix.copyDirectory('node\_modules/tinymce/skins', 'public/js/tinymce/skins');

Ở đây ta mix file node\_modules/tinymce/tinymce.min.js vào chung trong file public/js/app.js và copy các plugins, themes, icons, skins ra ngoài thư mục public để tiny có thể gọi đến.

Bước 3: Tạo view và route  
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Timymce</title>

</head>

<body>

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

@csrf

<textarea id="timymce" name="tinymce"></textarea>

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

</form>

<script src="{{ mix('js/app.js') }}"></script>

<script>

tinymce.baseURL = "js/tinymce";

tinymce.init({

selector: "textarea#timymce",

toolbar: "insertfile undo redo | styleselect | bold italic |

alignleft aligncenter alignright alignjustify | bullist numlist outdent indent",

});

</script>

</body>

</html>

Route::get('/tinymce', function () {;

return view('tinymce');

});

* Giao diện của tinymce sau khi tích hợp vào textarea

# Laravel filemanager

**Cài đặt**

* Yêu cầu:
  + php >= 5.4
  + exif extension
  + fileinfo extension
  + GD Library >=2.0 or Imagick PHP extension >=6.5.7
  + Laravel 5
* Tiến hành cài đặt
* Bước 1: Cài đặt package unisharp/laravel-filemanager bằng composer

composer require unisharp/laravel-filemanager

* Bước 2: Đưa các config và assets của package ra ngoài

php artisan vendor:publish --tag=lfm\_config

php artisan vendor:publish --tag=lfm\_public

Bước 3: Clear cache

php artisan route:clear

php artisan config:clear

Bước 4: Sysmlink tài nguyên từ storage ra ngoài public để laravel filemanager có thể truy cập đến

php artisan storage:link

Bước 5: Tạo các route cần thiết cho laravel filemanager

Route::group(['prefix' => 'laravel-filemanager', 'middleware' => ['web', 'auth']], function () {

\UniSharp\LaravelFilemanager\Lfm::routes();

});

* Thử vào trang demo nào {BASE\_URL}/laravel-filemanager/demo

**File config lfm**

<?php

return [

'use\_package\_routes' => true,

'allow\_private\_folder' => true,

'private\_folder\_name' => UniSharp\LaravelFilemanager\Handlers\ConfigHandler::class,

'allow\_shared\_folder' => false,

'shared\_folder\_name' => 'shares',

'folder\_categories' => [

'file' => [

'folder\_name' => 'files',

'startup\_view' => 'grid',

'max\_size' => 50000, // size in KB

'valid\_mime' => [

'image/jpeg',

'image/pjpeg',

'image/png',

'image/gif',

'image/svg+xml',

],

],

'image' => [

'folder\_name' => 'photos',

'startup\_view' => 'list',

'max\_size' => 50000, // size in KB

'valid\_mime' => [

'image/jpeg',

'image/pjpeg',

'image/png',

'image/gif',

'image/svg+xml',

'application/pdf',

'text/plain',

],

],

],

'paginator' => [

'perPage' => 30,

],

'disk' => 'public',

'rename\_file' => false,

'alphanumeric\_filename' => false,

'alphanumeric\_directory' => false,

'should\_validate\_size' => false,

'should\_validate\_mime' => false,

'over\_write\_on\_duplicate' => false,

'should\_create\_thumbnails' => false,

'thumb\_folder\_name' => 'thumbs',

'raster\_mimetypes' => [

'image/jpeg',

'image/pjpeg',

'image/png',

],

'thumb\_img\_width' => 200, // px

'thumb\_img\_height' => 200, // px

'file\_type\_array' => [

'pdf' => 'Adobe Acrobat',

'doc' => 'Microsoft Word',

'docx' => 'Microsoft Word',

'xls' => 'Microsoft Excel',

'xlsx' => 'Microsoft Excel',

'zip' => 'Archive',

'gif' => 'GIF Image',

'jpg' => 'JPEG Image',

'jpeg' => 'JPEG Image',

'png' => 'PNG Image',

'ppt' => 'Microsoft PowerPoint',

'pptx' => 'Microsoft PowerPoint',

],

'php\_ini\_overrides' => [

'memory\_limit' => '256M',

],

];

* Một số cấu hình cần lưu ý:
  + paginator: Hiển thị số lượng mỗi trang.
  + disk: Nơi lưu trữ ảnh (với giá trị public: Sẽ lưu dưới local, s3: Sẽ lưu trữ trên cloud dựa vào config trong filemanager.php)
  + should\_validate\_size: validate file upload.
  + should\_validate\_mime: validate minetype.
  + raster\_mimetypes: Định nghĩa minetype được upload.
* Các config còn lại các bạn có thể tự tìm hiểu...

# Tích hợp laravel filemanager vào tinymce

* Bước 1: Sử dụng plugins image, imagetool và toolbar image
  + Cấu hình tinymce sẽ như sau:

tinymce.init({

selector: "textarea#timymce",

plugins: [

"image imagetool",

],

toolbar: "insertfile undo redo | styleselect | bold italic |

lignleft aligncenter alignright alignjustify | bullist numlist outdent indent |

image",

});

* Bước 2: Sử dụng function file\_picker\_callback(callback, value, meta) để custom lại dialog chọn image file click vào biểu tượng upload
  + callback: Một callback được gọi, khi bạn có tệp nó mong đợi giá trị mới cho trường làm đối số đầu tiên và tùy chọn thông tin meta cho các trường khác trong dialog làm đối số thứ hai.
  + value: Giá trị hiện tại của trường bị tác động.
  + meta: Đối tượng chứa giá trị của các trường khác trong dialog được chỉ định.
  + Tiny chỉ cung cấp cho cung ta cái hook, còn việc triển khai như thế nào thì mình lại tự code.

tinymce.init({

selector: "textarea#timymce",

plugins: [

"image imagetool",

],

toolbar: "insertfile undo redo | styleselect | bold italic |

alignleft aligncenter alignright alignjustify | bullist numlist outdent indent |

image",

file\_picker\_callback: function (callback, value, meta) {

let x = window.innerWidth || document.documentElement.clientWidth || document.getElementsByTagName('body')[0].clientWidth;

let y = window.innerHeight|| document.documentElement.clientHeight|| document.getElementsByTagName('body')[0].clientHeight;

let type = 'image' === meta.filetype ? 'Images' : 'Files',

url = '/laravel-filemanager?editor=tinymce5&type=' + type;

tinymce.activeEditor.windowManager.openUrl({

url : url,

title : 'Filemanager',

width : x \* 0.8,

height : y \* 0.8,

onMessage: (api, message) => {

callback(message.content);

}

});

}

});

* Đầu tiên file\_picker\_callback sẽ lấy with, hieght của màn hình hiện tại, để tính toán kích thước của dialog khi upload.
* Tiếp theo sẽ kiểm tra file type và sinh ra một url.
* Open một dialog cho phép chọn ảnh.
  + url: Giá trị vừa được sinh ra ở trên.
  + title: Tiêu đề của dialog.
  + width: Chiều rộng của dialog.
  + height: Chiều cao của dialog.
  + onMessage: Khi chọn ảnh trong filemanager xong thì sẽ trả về link ảnh cho dialog của tinymce.

# Laravel File Manager Tutorial Step by Step

Step 1: Install alexusmai/laravel-file-manager Package

in first step, we need install alexusmai/laravel-file-manager composer package so let's use bellow command to install:

composer **require** alexusmai/laravel-file-manager

after installing successfully, we need to publish configuration file using bellow command:

php artisan vendor:publish --tag=fm-config

Now they created file-manager.php file in config folder. you can add or remove disk list from "diskList" key as bellow:

**config/file-manager.php**

....

'diskList' => ['local'],

....

now we need to public asset files with following command:

php artisan vendor:publish --tag=fm-assets

Step 2: Create Route

In this is step we need to create some routes for filemanager view.

**routes/web.php**

<?php

**use** Illuminate\Support\Facades\Route;

**use** App\Http\Controllers\FileManagerController;

/\*

|--------------------------------------------------------------------------

| Web Routes

|--------------------------------------------------------------------------

|

| Here is where you can register web routes for your application. These

| routes are loaded by the RouteServiceProvider within a group which

| contains the "web" middleware group. Now create something great!

|

\*/

Route::**get**('filemanager', [FileManagerController::**class**, 'index']);

Read Also: [Laravel 8 Send Mail using Gmail SMTP Server](https://www.itsolutionstuff.com/post/laravel-8-send-mail-using-gmail-smtp-serverexample.html)

Step 3: Create Controller

in this step, we need to create FileManagerController and add following code on that file:

**app/Http/Controllers/FileManagerController.php**

<?php

**namespace** App\Http\Controllers;

**use** Illuminate\Http\Request;

**class** FileManagerController **extends** Controller

{

/\*\*

\* Display a listing of the resource.

\*

\* @return \Illuminate\Http\Response

\*/

**public** **function** index(Request $request)

{

**return** view('filemanager');

}

}

Step 4: Create Blade Files

here, we need to create blade files for filemanager. so let's create one by one files:

here will code for navbar links:

**resources/views/filemanager.blade.php**

<!DOCTYPE html>

**<html** **lang**="{{ app()->getLocale() }}">

**<head>**

**<meta** **charset**="utf-8"**>**

**<meta** **name**="viewport" **content**="width=device-width, initial-scale=1, shrink-to-fit=no"**>**

<!-- CSRF Token -->

**<meta** **name**="csrf-token" **content**="{{ csrf\_token() }}"**>**

**<title>**{{ config('app.name', 'File Manager') }}**</title>**

<!-- Styles -->

**<link** **rel**="stylesheet" **href**="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/css/bootstrap.min.css"**>**

**<link** **rel**="stylesheet" **href**="https://use.fontawesome.com/releases/v5.15.2/css/all.css"**>**

**<link** **href**="{{ asset('vendor/file-manager/css/file-manager.css') }}" **rel**="stylesheet"**>**

**<link** **href**="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"**>**

**</head>**

**<body>**

**<div** **class**="container"**>**

**<h2>**Laravel File Manager Tutorial Example - ItSolutionStuff.Com**</h2>**

**<div** **class**="row"**>**

**<div** **class**="col-md-12" **id**="fm-main-block"**>**

**<div** **id**="fm"**></div>**

**</div>**

**</div>**

**</div>**

<!-- File manager -->

**<script** **src**="{{ asset('vendor/file-manager/js/file-manager.js') }}"**></script>**

**<script>**

document.addEventListener('DOMContentLoaded', **function**() {

document.getElementById('fm-main-block').setAttribute('style', 'height:' + window.innerHeight + 'px');

fm.$store.commit('fm/setFileCallBack', **function**(fileUrl) {

window.opener.fmSetLink(fileUrl);

window.close();

});

});

**</script>**

**</body>**

**</html>**

Now we are ready to run our example. so run bellow command so quick run:

php artisan serve

Add to TinyMCE 5 configuration

tinymce.init({

selector: '#my-textarea',

// ...

file\_picker\_callback (callback, value, meta) {

let x = window.innerWidth || document.documentElement.clientWidth || document.getElementsByTagName('body')[0].clientWidth

let y = window.innerHeight|| document.documentElement.clientHeight|| document.getElementsByTagName('body')[0].clientHeight

tinymce.activeEditor.windowManager.openUrl({

url : '/file-manager/tinymce5',

title : 'Laravel File manager',

width : x \* 0.8,

height : y \* 0.8,

onMessage: (api, message) => {

callback(message.content, { text: message.text })

}

})

}

});

### Standalone button

Add button

<div class="input-group">

<input type="text" id="image\_label" class="form-control" name="image"

aria-label="Image" aria-describedby="button-image">

<div class="input-group-append">

<button class="btn btn-outline-secondary" type="button" id="button-image">Select</button>

</div>

</div>

and js script

document.addEventListener("DOMContentLoaded", function() {

document.getElementById('button-image').addEventListener('click', (event) => {

event.preventDefault();

window.open('/file-manager/fm-button', 'fm', 'width=1400,height=800');

});

});

// set file link

function fmSetLink($url) {

document.getElementById('image\_label').value = $url;

}

Ghi chú: file manager

lavarel-filemanager:

php artisan storage => create storage folder in public folder and auto link

don't use imagetool in lavarel-filemanager

APP\_URL trong file .evn để là APP\_URL=http://127.0.0.1:8000

file app.php trong thư mục config để là 'url' => env('APP\_URL', 'http://127.0.0.1:8000'),

file web.php trong thư mục route thêm 2 route sau:

Route::get('/filemanager', function () {;

return view('filemanager');

});

Route::group(['prefix' => 'laravel-filemanager'], function () {

\UniSharp\LaravelFilemanager\Lfm::routes();

});

**Cài đặt imagick chop hp**

Check imagick đã cài hay chưa

**if** ( ! extension\_loaded( 'imagick' ) ) {

**echo** 'Imagick not installed';

}

Việc đầu tiên bạn cần làm trước khi tiến hành cài đặt Imagick đó là bạn mở XAMPP lên, sau đó bạn kiểm tra phpinfo xem các thông số: Compiler, Architecture, Thread Safety. Tùy thuộc vào các thông số này mà nội dung bài viết bạn sẽ thực hiện theo có thể khác đi.

### 1. Tải và cài đặt ImageMagick cho hệ điều hành Windows

<https://imagemagick.org/script/download.php>

Bạn chọn tập tin tương thích với các thông số như ở lúc đầu trước khi tiến hành cài đặt

### 2. Tải Imagick cho ngôn ngữ PHP

Truy cập vào <https://pecl.php.net/package/imagick>

và tải thư viện tập tin DLL mới nhất về máy tính. Bạn nhấn chuột vào biểu tượng lá cờ Windows có chữ DLL, một trang mới hiện ra, bạn chọn gói phù hợp với phiên bản PHP và nền tảng kiến trúc bạn đang sử dụng

Bạn tải về, giải nén ra và chỉ sao chép tập tin php\_imagick.dll bỏ vào đường dẫn thư mục php/ext của XAMPP trên máy tính của bạn. Tiếp đến mở tập tin php.ini lên và thêm dòng extension=php\_imagick.dll vào khu vực dưới Dynamic Extensions.

### 3. Tải thư viện cần thiết Imagick binaries

Vào link <https://windows.php.net/downloads/pecl/deps/>

bạn tìm trên trình duyệt với từ khóa ImageMagick bằng cách nhấn Ctrl+F. Bạn chọn tập tin phù hợp với trình biên dịch đang sử dụng trên máy của bạn

Bạn tải về, giải nén và vào thư mục bin, tìm kiếm tất cả các tập tin .dll bằng cách tìm kiếm trên File Explorer của hệ điều hành Windows từ khóa \*.dll. Bạn sao chép toàn bộ tập tin .dll (ngoại trừ tập tin ImageMagickObject.dll) và bỏ vào thư mục apache/bin của XAMPP

146 tập tin .dll (có tiền tố CORE\_ và IM\_MOD\_).

Sau khi hoàn thành mọi thứ, bạn khởi động lại Apache trên bảng điều khiển XAMPP bằng cách nhấn Stop và Start trở lại. Nếu không gặp thông báo lỗi gì thì bạn đã tiến hành kích hoạt Imagick thành công trên XAMPP với phiên bản PHP 7.1.1.