ЧТО ДЕЛАЕМ:Проектируем и внедряем удобную и масштабируемую структуру папок для фото объявлений, чтобы удобно хранить/выдавать медиа, не упираться в лимиты ФС и легко переезжать на S3/CDN.🎯 ПЛАН:1) Выбрать схему директорий2) Добавить генератор пути (не ломая текущее хранилище)3) Сохранять файлы через диски Laravel (storage/app/public) и раздавать через /storage4) Подготовить миграцию для метаданных (если нужно)5) Показать пример сохранения и выдачи изображений

**1) Рекомендуемая схема директорий**

Чтобы:

* не было десятков тысяч файлов в одной папке,
* можно было быстро чистить по пользователю/объявлению,
* легко уехать на S3,

предлагаю такую иерархию:

* Базовый путь: public диск (симлинк на /public/storage)
* Паттерн: users/{userIdShard}/ads/{adIdShard}/{YYYY}/{MM}/{uuid}.{ext}

Где:

* {userIdShard} и {adIdShard} — шардинг по ID, чтобы не скапливать всё в одной папке (например, 000/123/456 для id=123456).
* {YYYY}/{MM} — разбивка по месяцам.
* {uuid}.{ext} — уникальное имя файла.

Пример:

* Оригинал: storage/app/public/users/000/001/234/ads/000/045/678/2025/08/4faa9b92-7f90-4d44-9a6b-1d6b3e8d7c6d.jpg
* Превью (если делаем версии): .../thumbs/4faa9b92-...\_400x400.jpg

Если не хотите шардинг — можно упростить:

* users/{userId}/ads/{adId}/{YYYY}/{MM}/{uuid}.{ext}

Но я рекомендую шардинг для будущего роста.

**2) Генератор путей (удобно и безопасно)**

Создадим утилиту, которую можно вызывать из сервисов (например, из AdMediaService), не трогая текущую логику сохранения.💻 КОД:Создайте файл app/Infrastructure/Media/PathGenerator.php:

php

Apply



<?php

namespace App\Infrastructure\Media;

use Illuminate\Support\Str;

use Carbon\Carbon;

class PathGenerator

{

    // Разбиваем число по 3 символа: 123456 -> 000/123/456

    protected static function shard(int $id): string

    {

        $padded = str\_pad((string) $id, 9, '0', STR\_PAD\_LEFT);

        return implode('/', str\_split($padded, 3));

    }

    public static function adPhotoPath(int $userId, int $adId, string $extension = 'jpg', ?Carbon $date = null): string

    {

        $date = $date ?: Carbon::now();

        $userShard = self::shard($userId);

        $adShard = self::shard($adId);

        $uuid = (string) Str::uuid();

        return sprintf(

            'users/%s/ads/%s/%s/%s/%s.%s',

            $userShard,           // users/000/001/234

            $adShard,             // ads/000/045/678

            $date->format('Y'),   // 2025

            $date->format('m'),   // 08

            $uuid,                // 4faa9b92-...

            ltrim($extension, '.') // jpg

        );

    }

    public static function adThumbPath(int $userId, int $adId, string $suffix, string $extension = 'jpg', ?Carbon $date = null): string

    {

        $date = $date ?: Carbon::now();

        $userShard = self::shard($userId);

        $adShard = self::shard($adId);

        $uuid = (string) Str::uuid();

        return sprintf(

            'users/%s/ads/%s/%s/%s/thumbs/%s\_%s.%s',

            $userShard,

            $adShard,

            $date->format('Y'),

            $date->format('m'),

            $uuid,

            $suffix, // например 400x400

            ltrim($extension, '.')

        );

    }

}

**3) Сохранение файла через Laravel Filesystem**

Пример использования в любом сервисе (например, в AdMediaService):

php

Apply



use Illuminate\Http\UploadedFile;

use Illuminate\Support\Facades\Storage;

use App\Infrastructure\Media\PathGenerator;

public function storeAdPhoto(UploadedFile $file, int $userId, int $adId): string

{

    $ext = $file->getClientOriginalExtension() ?: 'jpg';

    $path = PathGenerator::adPhotoPath($userId, $adId, $ext);

    // сохраняем на диск public

    Storage::disk('public')->put($path, file\_get\_contents($file->getRealPath()));

    // вернём относительный путь (для базы) или URL

    return $path; // например: users/.../2025/08/uuid.jpg

}

Выдача в шаблоне/фронте:

* URL: /storage/{path} → asset('storage/'.$path)

**4) База данных (если нужно хранить метаданные)**

Если у вас нет таблицы медиаданных для объявлений — добавьте:

php

Apply



// database/migrations/xxxx\_xx\_xx\_xxxxxx\_create\_ad\_media\_table.php

Schema::create('ad\_media', function (Blueprint $table) {

    $table->id();

    $table->unsignedBigInteger('ad\_id')->index();

    $table->unsignedBigInteger('user\_id')->index();

    $table->string('disk', 50)->default('public');

    $table->string('path', 512);     // users/.../uuid.jpg

    $table->string('mime', 100)->nullable();

    $table->unsignedInteger('size')->nullable();

    $table->unsignedInteger('width')->nullable();

    $table->unsignedInteger('height')->nullable();

    $table->boolean('is\_main')->default(false);

    $table->unsignedInteger('order')->default(0);

    $table->timestamps();

    $table->foreign('ad\_id')->references('id')->on('ads')->cascadeOnDelete();

});

Плюсы:

* можно быстро сменить диск на s3 (поле disk)
* легко достать главное фото, сортировку, превью и т.д.

**5) Удаление и чистка**

* При удалении объявления — удаляйте файлы + записи ad\_media (Observer/Job).
* Для одного объявления легко подчистить: просто удалите папку users/.../ads/.../YYYY/MM (только если это изолировано под это объявление) и записи в БД.

**Важные рекомендации**

* Для производительности обязательно делайте шардинг или разбиение по дате — не храните десятки тысяч файлов в одной директории.
* На проде лучше S3/MinIO + CDN. Структура путей останется такой же, просто сменится disk.
* Для превью/версий изображений лучше генерировать на загрузке (например, thumbs/400x400, 800x800) в ту же ветку директорий.
* Старайтесь хранить в БД только относительный path и disk — URL стройте на лету (Storage::disk($disk)->url($path)).

🔧 КОМАНДЫ:

* Создать симлинк для раздачи из public:

text

Apply



php artisan storage:link

* Опционально установить обработчик изображений (превью):

text

Apply



composer require intervention/image

✅ ПРОВЕРКА:1) Загрузите фото объявления через форму2) Убедитесь, что файл попал в storage/app/public/users/...3) В базе (если используете таблицу ad\_media) появилась запись с disk=public и path4) На фронте отображается через asset('storage/'.$path) или Storage::disk('public')->url($path)