

PRAKTIKUM PEMROGRAMAN FRAMEWORK

MODUL 9 Laravel Factory & File Storage



Disusun Oleh:
Purnama Anaking, S.Kom., M.Kom.

**PROGRAM STUDI S1 SISTEM INFORMASI
FAKULTAS TEKNOLOGI INFORMASI DAN BISNIS
INSTITUT TEKNOLOGI TELKOM SURABAYA
2023**

DAFTAR ISI

| | |
|---|----|
| DAFTAR ISI | 2 |
| 1. Membuat dan Mendefinisikan Factory | 3 |
| 2. Menjalankan Factory via Seeder | 4 |
| 3. Meletakkan File pada Local Disk | 4 |
| 4. Meletakkan File pada Public Disk | 4 |
| 5. Menampilkan Isi File | 5 |
| 6. Mendownload File | 5 |
| 7. Menampilkan URL, Path dan Size dari File | 6 |
| 8. Menyimpan File via Form | 6 |
| 9. Menghapus File pada Storage | 7 |
| 10. Menerapkan Fitur Upload pada Web Data Master Employee | 7 |
| 11. Latihan | 12 |
| 12. Tugas | 12 |

MODUL 9

LARAVEL FACTORY & FILE STORAGE

Pada praktikum kali ini kita akan melanjutkan belajar tentang Laravel Factory dan File Storage. Mahasiswa akan mencoba menerapkan fitur Laravel Factory untuk men-generate data dummy ke database dan menerapkan Laravel File Storage untuk mengelola asset file pada project laravel yang dimiliki.

1. Membuat dan Mendefinisikan Factory

- Buat file Factory untuk tabel **positions** dengan command artisan sebagai berikut:

```
php artisan make:factory PositionFactory
```

- Definiskan value setiap kolom yang dimiliki tabel **positions** dengan bantuan faker untuk men-generate data dummy. Sesuaikan isi function **definition()** seperti kode program di bawah ini.

```
public function definition(): array
{
    return [
        'code' => fake()->stateAbbr(),
        'name' => fake()->jobTitle(),
        'description' => fake()->sentence(),
    ];
}
```

- Buat file Factory untuk tabel **employees** dengan command artisan sebagai berikut:

```
php artisan make:factory EmployeeFactory
```

- Definiskan value setiap kolom yang dimiliki tabel **employees** dengan bantuan faker untuk men-generate data dummy. Sesuaikan isi function **definition()** seperti kode program di bawah ini.

```
public function definition(): array
{
    return [
        'firstname' => fake()->firstName(),
        'lastname' => fake()->lastName(),
        'email' => fake()->email(),
        'age' => fake()->numberBetween(25, 50),
        'position_id' => Position::factory()
    ];
}
```

2. Menjalankan Factory via Seeder

- Buat file **PositionSeeder.php** kemudian **generate 5 data dummy** untuk tabel **positions** dengan menuliskan kode program seperti di bawah ini.

```
Position::factory()->count(5)->create();
```

- Buat file **EmployeeSeeder.php** kemudian **generate 10 data dummy** untuk tabel **employees** dengan menuliskan kode program seperti di bawah ini.

```
Employee::factory()->count(10)->create();
```

- Jalankan seeder yang telah disesuaikan dengan command di bawah ini.

```
php artisan db:seed
```

3. Meletakkan File pada Local Disk

- Buka file **routes/web.php** kemudian buat Route baru dengan kode program seperti di bawah ini. Kemudian akses Route tersebut via Browser.

```
Route::get('/local-disk', function() {  
    Storage::disk('local')->put('local-example.txt', 'This is local example  
content');  
    return asset('storage/local-example.txt');  
});
```

- Jika berhasil akan terbuat file dengan nama **local-example.txt** di folder **/storage/app** pada project laravel anda.

4. Meletakkan File pada Public Disk

- Buat **Symbolic Link** yang menghubungkan antara direktori **/storage/app/public** dengan direktori **/public/storage** via command di bawah ini:

```
php artisan storage:link
```

- Buka file **routes/web.php** kemudian buat Route baru dengan kode program seperti di bawah ini. Kemudian akses Route tersebut via Browser.

```
Route::get('/public-disk', function() {  
    Storage::disk('public')->put('public-example.txt', 'This is public example  
content');  
    return asset('storage/public-example.txt');  
});
```

- Jika berhasil akan terbuat file dengan nama **public-example.txt** di folder **/storage/app/public** pada project laravel anda.
- Anda dapat mengakses file yang sudah terbuat tersebut via browser dengan URL:

```
http://localhost:8000/storage/public-example.txt
```

- URL di atas mengakses file melalui folder **/public/storage** yang memiliki “**symbolic link**” ke direktori **/storage/app/public**.

5. Menampilkan Isi File

- Menampilkan isi file **local** dengan cara buat Route baru dengan kode program di bawah ini.

```
Route::get('/retrieve-local-file', function() {  
    if (Storage::disk('local')->exists('local-example.txt')) {  
        $contents = Storage::disk('local')->get('local-example.txt');  
    } else {  
        $contents = 'File does not exist';  
    }  
  
    return $contents;  
});
```

- Menampilkan isi file **public** dengan cara buat Route baru dengan kode program di bawah ini.

```
Route::get('/retrieve-public-file', function() {  
    if (Storage::disk('public')->exists('public-example.txt')) {  
        $contents = Storage::disk('public')->get('public-example.txt');  
    } else {  
        $contents = 'File does not exist';  
    }  
  
    return $contents;  
});
```

- Akses route-route tersebut via browser lalu analisis apa yang ditampilkan pada browser.

6. Mendownload File

- Mendownload file **local** dengan cara buat Route baru dengan kode program di bawah ini.

```
Route::get('/download-local-file', function() {  
    return Storage::download('local-example.txt', 'local file');  
});
```

- Mendownload file **public** dengan cara buat Route baru dengan kode program di bawah ini.

```
Route::get('/download-public-file', function() {  
    return Storage::download('public/public-example.txt', 'public file');  
});
```

- Akses route-route tersebut via browser lalu analisis apa yang ditampilkan pada browser.

7. Menampilkan URL, Path dan Size dari File

- Buat route baru dengan kode program di bawah ini.

```
Route::get('/file-url', function() {
    // Just prepend "/storage" to the given path and return a relative URL
    $url = Storage::url('local-example.txt');
    return $url;
});

Route::get('/file-size', function() {
    $size = Storage::size('local-example.txt');
    return $size;
});

Route::get('/file-path', function() {
    $path = Storage::path('local-example.txt');
    return $path;
});
```

- Akses route-route tersebut via browser lalu analisis apa yang ditampilkan pada browser.

8. Menyimpan File via Form

- Buat route baru dengan kode program di bawah ini.

```
Route::get('/upload-example', function() {
    return view('upload_example');
});

Route::post('/upload-example', function(Request $request) {
    $path = $request->file('avatar')->store('public');
    return $path;
})->name('upload-example');
```

- Buat file view dengan nama **upload_example.blade.php**. Tulis kode program seperti di bawah ini.

```
<!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>Upload Example</title>
</head>
<body>
    <form action="{{ route('upload-example') }}" method="POST"
    enctype="multipart/form-data">
        @csrf
```

```

        <input type="file" name="avatar">
        <button type="submit">Upload</button>
    </form>
</body>
</html>

```

- Akses URL **/upload-example** via browser, pilih salah satu file, lalu klik button **Upload**.

Choose File

No file chosen

Upload

- Jika berhasil maka file yang diupload akan tersimpan pada direktori **/storage/app/public**.

9. Menghapus File pada Storage

- Buat route baru dengan kode program di bawah ini.

```

Route::get('/delete-local-file', function(Request $request) {
    Storage::disk('local')->delete('local-example.txt');
    return 'Deleted';
});

Route::get('/delete-public-file', function(Request $request) {
    Storage::disk('public')->delete('public-example.txt');
    return 'Deleted';
});

```

- Akses route-route tersebut via browser lalu analisis apa yang terjadi.
- Jika berhasil maka file yang bersangkutan akan terhapus pada project laravel anda.

10. Menerapkan Fitur Upload pada Web Data Master Employee

- Buat file migration baru untuk menambahkan kolom yang digunakan menyimpan data nama file **original** dan yang **terenkripsi**. Tulis command di bawah ini:

```
php artisan make:migration add_filename_column_into_employees_table
```

- Buka file migration yang baru saja ter-generate. Tulis kode program di bawah ini untuk menambahkan 2 kolom baru dengan nama **“original_filename”** dan **“encrypted_filename”**.

```

/**
 * Run the migrations.
 */
public function up(): void
{
    Schema::table('employees', function (Blueprint $table) {
        $table->string('original_filename')->after('position_id')->nullable();
    });
}

```

```

$table->string('encrypted_filename')->after('original_filename')->nullable();
    });
}

/**
 * Reverse the migrations.
 */
public function down(): void
{
    Schema::table('employees', function (Blueprint $table) {
        $table->dropColumn('encrypted_filename');
        $table->dropColumn('original_filename');
    });
}

```

- Jalankan “**php artisan migrate**” untuk meng-eksekusi file migration yang baru saja kita buat. Jika berhasil maka akan muncul 2 kolom baru pada tabel **employees** di database anda.
- Buka file **/views/employee/create.blade.php**, sesuaikan fitur **Form Create Employee** dengan atribut **enctype** pada tag **<form>**.

```

<form action="{{ route('employees.store') }}" method="POST"
enctype="multipart/form-data">


```

- Tambahkan kode program di bawah ini setelah bagian yang menampilkan pilihan “Position” pada form.

```

<div class="col-md-12 mb-3">
    <label for="cv" class="form-label">Curriculum Vitae (CV)</label>
    <input type="file" class="form-control" name="cv" id="cv">
</div>

```



Create Employee

First Name

Last Name

Email

Age

Position

MD - Sociology Teacher
▼

Curriculum Vitae (CV)

Choose File

No file chosen

⬅ Cancel

✔ Save

- Buka file **EmployeeController.php** kemudian sesuaikan function **store()** seperti di bawah ini.

```
public function store(Request $request)
{
    $messages = [
        'required' => ':Attribute harus diisi.',
        'email' => 'Isi :attribute dengan format yang benar',
        'numeric' => 'Isi :attribute dengan angka'
    ];

    $validator = Validator::make($request->all(), [
        'firstName' => 'required',
        'lastName' => 'required',
        'email' => 'required|email',
        'age' => 'required|numeric',
    ], $messages);

    if ($validator->fails()) {
        return redirect()->back()->withErrors($validator)->withInput();
    }

    // Get File
    $file = $request->file('cv');

    if ($file != null) {
        $originalFilename = $file->getClientOriginalName();
        $encryptedFilename = $file->hashName();

        // Store File
        $file->store('public/files');
    }

    // ELOQUENT
    $employee = New Employee;
    $employee->firstname = $request->firstName;
    $employee->lastname = $request->lastName;
    $employee->email = $request->email;
    $employee->age = $request->age;
    $employee->position_id = $request->position;

    if ($file != null) {
        $employee->original_filename = $originalFilename;
        $employee->encrypted_filename = $encryptedFilename;
    }

    $employee->save();

    return redirect()->route('employees.index');
}
```

- Jalankan fitur **Form Create Employee** via browser, jika berhasil maka data employee akan masuk ke database dan file CV akan tersimpan di direktori **/storage/app/public/files** pada project laravel anda.

- Buat route baru pada file **/routes/web/php** dengan menambahkan kode program seperti di bawah ini untuk operasi **download file**.

```
Route::get('download-file/{employeeId}', [EmployeeController::class,
'downloadFile'])->name('employees.downloadFile');
```

- Buka file **/views/employee/show.blade.php**, sesuaikan kode program seperti di bawah ini.

```
@extends('layouts.app')

@section('content')
    <div class="container-sm my-5">
        <div class="row justify-content-center">
            <div class="p-5 bg-light rounded-3 col-xl-4 border">
                <div class="mb-3 text-center">
                    <i class="bi-person-circle fs-1"></i>
                    <h4>Detail Employee</h4>
                </div>
                <hr>
                <div class="row">
                    <div class="col-md-4 mb-3">
                        <label for="firstName" class="form-label">First
Name</label>

                        <h5>{{ $employee->firstname }}</h5>
                    </div>
                    <div class="col-md-4 mb-3">
                        <label for="lastName" class="form-label">Last
Name</label>

                        <h5>{{ $employee->lastname }}</h5>
                    </div>
                    <div class="col-md-4 mb-3">
                        <label for="age" class="form-label">Age</label>
                        <h5>{{ $employee->age }}</h5>
                    </div>
                    <div class="col-md-12 mb-3">
                        <label for="email" class="form-label">Email</label>
                        <h5>{{ $employee->email }}</h5>
                    </div>
                    <div class="col-md-12 mb-3">
                        <label for="age" class="form-label">Position</label>
                        <h5>{{ $employee->position->name }}</h5>
                    </div>
                    <div class="col-md-12 mb-3">
                        <label for="age" class="form-label">Curriculum Vitae
(CV)</label>

                        @if ($employee->original_filename)
                            <h5>{{ $employee->original_filename }}</h5>
                            <a href="{ route('employees.downloadFile',
['employeeId' => $employee->id]) }}" class="btn btn-primary btn-sm mt-2">
                                <i class="bi bi-download me-1"></i> Download CV
                            </a>
                        @else
                            <h5>Tidak ada</h5>
                        @endif
                    </div>
                </div>
            </div>
        </div>
    </div>
```

```

        </div>
    </div>
    <hr>
    <div class="row">
        <div class="col-md-12 d-grid">
            <a href="{{ route('employees.index') }}" class="btn
btn-outline-dark btn-lg mt-3"><i class="bi-arrow-left-circle me-2"></i> Back</a>
        </div>
    </div>
</div>
</div>
</div>
@endsection

```

- Buka file **EmployeeController.php** lalu tambahkan function **downloadFile()** seperti di bawah ini.


```

public function downloadFile($employeeId)
{
    $employee = Employee::find($employeeId);
    $encryptedFilename = 'public/files/'.$employee->encrypted_filename;
    $downloadFilename =
Str::lower($employee->firstname.'_'.$employee->lastname.'_cv.pdf');

    if(Storage::exists($encryptedFilename)) {
        return Storage::download($encryptedFilename, $downloadFilename);
    }
}

```

- Jika berhasil maka ketika anda mengakses halaman employee detail, maka akan tampilan button **“download”** CV dari employee. Dan CV employee dapat di-download dengan melakukan klik pada button tersebut.



Detail Employee

| | | |
|------------|-----------|-----|
| First Name | Last Name | Age |
| Purnama | Anaking | 30 |

Email
purnama@gmail.com

Position
Sociology Teacher

Curriculum Vitae (CV)
11 - PUR - PBO - Interface & Abstract Class (Modul Praktikum).pdf

[Download CV](#)

⬅ Back

11. Latihan

- Lanjutkan **Fitur Edit Employee**
 - Edit data employee di database
 - Ubah file CV nya dengan cara: file CV yang sebelumnya, dihapus dari project laravel anda dan diganti dengan file CV yang baru.

The image displays two versions of the 'Edit Employee' form. The left form shows the initial state where the CV field is empty ('Tidak ada'). The right form shows the state after a file upload, with a 'Download' button appearing next to the CV field. Both forms include input fields for personal and professional details, and 'Cancel'/'Edit' action buttons.

- Lanjutkan **Fitur Hapus Employee**
 - Hapus data employee dari database
 - Hapus file CV nya dari project laravel anda

12. Tugas

1. **Praktekan** seluruh poin **praktikum dan latihan** yang ada di atas secara **INDIVIDU**. Buat pada **1 repository** Github saja. Infokan URL Repository tersebut pada laporan praktikum.
2. Dokumentasikan seluruhnya (**screenshot output**) dalam bentuk Laporan Praktikum.
3. Kumpulkan Laporan Praktikum (**.pdf**) yang telah dibuat di dalam praktikum via **E-Learning** paling lambat sebelum jadwal praktikum minggu depan.